

Prolog

Evaluador aritmético ‘is’ y Operadores Relacionales

Conceptos de Inteligencia Artificial
Sistemas Inteligentes Artificiales

Departamento de Ciencias e Ingeniería de la Computación
Universidad Nacional del Sur

Unificación – Predicado =

- Recordemos que el predicado predefinido $=/2$ intenta **unificar** los operandos a su izquierda y a su derecha.

Ejemplos:

$$X = 2$$

$$3 = 7 - 4$$

- La respuesta a una consulta utilizando este operador tiene éxito si es posible realizar tal **unificación**.

Unificación – Predicado =

Aridad de un
predicado
(cantidad de
argumentos)

- Recordemos que el predicado predefinido $=/2$ intenta unificar los operandos a su izquierda y a su derecha.

Ejemplos:

$$X = 2$$

$$3 = 7 - 4$$

- La respuesta a una consulta utilizando este operador tiene éxito si es posible realizar tal unificación.

Unificación – Predicado =

- Recordemos que el predicado predefinido $=/2$ intenta **unificar** los operandos a su izquierda y a su derecha.

Ejemplos:

$$X = 2$$

$$3 = 7 - 4$$

- La respuesta a una consulta utilizando este operador tiene éxito si es posible realizar tal **unificación**.
- ¿Cuál es la respuesta ante la siguiente consulta?

?- $f(X) = f(a)$.

¿ $f(X)$ y $f(a)$ unifican?

Unificación – Predicado =

- Recordemos que el predicado predefinido $=/2$ intenta **unificar** los operandos a su izquierda y a su derecha.

Ejemplos:

$$X = 2$$

$$3 = 7 - 4$$

- La respuesta a una consulta utilizando este operador tiene éxito si es posible realizar tal **unificación**.
- ¿Cuál es la respuesta ante la siguiente consulta?

?- $f(X) = f(a)$.

$X = a$

¿ $f(X)$ y $f(a)$ unifican?

Sí, cuando $X = a$.

Unificación – Predicado \=

- El predicado predefinido `\= /2` tiene éxito cuando no es posible unificar sus operandos.

`\=` tiene éxito cuando `=` falla y viceversa

Unificación – Predicado \=

- El predicado predefinido `\= /2`tiene éxito cuando no es posible unificar sus operandos.

`\=` tiene éxito cuando `=` falla y viceversa

- Ejemplos:

?- a `\=` a.

?- a `\=` b.

?- X `\=` a.

Unificación – Predicado \=

- El predicado predefinido `\= /2` tiene éxito cuando no es posible unificar sus operandos.

`\=` tiene éxito cuando `=` falla y viceversa

- Ejemplos:

?- a `\=` a.
false

?- a `\=` b.
true

?- X `\=` a.
false



La respuesta a las consultas utilizando `\=` es únicamente true o false!

Expresiones Aritméticas

- En Prolog las **expresiones aritméticas** se representan mediante **estructuras**.
- A continuación se muestra una **BNF** describiendo la **sintaxis** de las **expresiones aritméticas**:

```
E ::= Numero | +(E1, E2) | -(E1, E2) | *(E1, E2) | ...
```

Expresiones Aritméticas

- En Prolog las **expresiones aritméticas** se representan mediante estructuras.
- A continuación se muestra una **BNF** describiendo la **sintaxis** de las **expresiones aritméticas**:

```
E ::= Numero | +(E1, E2) | -(E1, E2) | *(E1, E2) | ...
```

- Ejemplos:

$+(2,3)$
 $*(5,+(3,2))$
 $-(*(5,8),10)$

Para **comodidad** del programador, Prolog admite el uso de **notación infija**

$2 + 3$
 $5 * (3 + 2)$
 $5 * 8 - 10$

Expresiones Aritméticas

- Cuando una **expresión aritmética** es utilizada en Prolog no siempre es evaluada.

Programa Prolog

```
p(2+2).
```

```
?- W = 1+2
```

```
?- 3 = 1+2
```

```
?- p(X).
```

```
?- p(4).
```

Expresiones Aritméticas

- Cuando una **expresión aritmética** es utilizada en Prolog no siempre es evaluada.

Programa Prolog

```
p(2+2).
```

```
?- W = 1+2
```

```
W = 1+2
```

```
?- 3 = 1+2
```

```
false
```

```
?- p(X).
```

```
X = 2+2
```

```
?- p(4).
```

```
false
```

Expresiones Aritméticas

- Cuando una **expresión aritmética** es utilizada en Prolog no siempre es evaluada.

Programa Prolog

```
p(2+2).
```

La **unificación tuvo éxito!** El intérprete simplemente retorna la sustitución aplicada

```
?- W = 1+2
```

```
W = 1+2
```

```
?- 3 = 1+2
```

```
false
```

```
?- p(X).
```

```
X = 2+2
```

```
?- p(4).
```

```
false
```

IMPORTANTE: El operador de unificación **NO evalúa expresiones aritméticas**

Evaluador Aritmético is /2 y Operadores Relacionales

- Dados dos términos T y E, donde E representa una expresión aritmética, la consulta:

```
?- is(T, E).
```

permite verificar si T unifica con el resultado de evaluar E

Evaluador Aritmético is /2 y Operadores Relacionales

- Dados dos términos T y E, donde E representa una expresión aritmética, la consulta:

```
?- is(T, E).
```

Para comodidad del programador,
Prolog admite el uso de notación infija:
T is E

permite verificar si T unifica con el resultado de evaluar E

Evaluador Aritmético is /2 y Operadores Relacionales

- Dados dos términos T y E, donde E representa una expresión aritmética, la consulta:

?- is(T, E).

Para comodidad del programador, Prolog admite el uso de notación infija:
T is E

permite verificar si T unifica con el resultado de evaluar E

?- T is E.

Primero evalúa E y luego intenta unificar T con el resultado de la evaluación

Evaluador Aritmético is /2 y Operadores Relacionales

- El evaluador aritmético **is** y los **operadores relationales** imponen **restricciones** sobre sus argumentos.

T is E

E debe ser una **expresión aritmética** y estar **completamente instanciada**

Evaluador Aritmético *is* /2 y Operadores Relacionales

- Las expresiones aritméticas son evaluadas cuando se utiliza el predicado *is* o algún operador relacional.

Programa Prolog

```
densidad(X, Y):- poblacion(X, P),  
                 area(X, A),  
                 Y is P/A.
```

```
poblacion(baires, 1000).  
area(baires, 10).  
poblacion(york, 500).  
area(york, mucha).
```

?- densidad(baires,DB).

?- densidad(york,DY).

?- densidad(baires,20).

?- densidad(baires,100).

Evaluador Aritmético *is* y Operadores Relacionales

- Las expresiones aritméticas son evaluadas cuando se utiliza el predicado *is* o algún operador relacional.

Programa Prolog

```
densidad(X, Y):- poblacion(X, P),  
                 area(X, A),  
                 Y is P/A.
```

```
poblacion(baires, 1000).  
area(baires, 10).  
poblacion(york, 500).  
area(york, mucha).
```

?- *densidad(baires,DB)*.

DB = 100

?- *densidad(york,DY)*.

ERROR!

?- *densidad(baires,20)*.

false

?- *densidad(baires,100)*.

true

Evaluador Aritmético is /2 y Operadores Relacionales

- Las expresiones aritméticas son evaluadas cuando se utiliza el predicado *is* o algún operador relacional.

Programa Prolog

```
densidad(X, Y):- poblacion(X, P),  
    area(X, A),  
    Y is P/A.
```

```
poblacion(baires, 1000).  
area(baires, 10).  
poblacion(york, 500).  
area(york, mucha).
```

?- densidad(baires,DB).

DB = 100

?- densidad(york,DY).

ERROR!

El error ocurre porque la evaluación aritmética no puede realizarse!

?- densidad(baires,100).
true

Evaluador Aritmético *is/2* y Operadores Relacionales

- Las expresiones aritméticas son evaluadas cuando se utiliza el predicado *is* o algún operador relacional.

```
fuePresidente(P,A):- presidente(P, Ini,Fin),  
                      A >= Ini,  
                      A < Fin.
```

Notación infija
de operadores
relacionales

Evaluador Aritmético *is/2* y Operadores Relacionales

- Las expresiones aritméticas son evaluadas cuando se utiliza el predicado *is* o algún operador relacional.

```
fuePresidente(P,A):- presidente(P, Ini, Fin),
```

Las variables **A**, **Ini** y **Fin** deben estar
instanciadas con expresiones aritméticas

$A \geq \text{Ini}$,
 $A < \text{Fin}$.

Notación infija
de operadores
relacionales

Evaluador Aritmético *is* /2 y Operadores Relacionales

- Las expresiones aritméticas son evaluadas cuando se utiliza el predicado *is* o algún operador relacional.

fuePresidente(P,A):- presidente(P, Ini, Fin),
A >= Ini,
A < Fin.

Las variables **A**, **Ini** y **Fin** deben estar instanciadas con expresiones aritméticas

Notación infija de operadores relacionales

El predicado **is** y los operadores relacionales provocan la evaluación de sus **argumentos**, los cuales deben ser expresiones aritméticas:

- **Predicado is:** Evalúa expresión aritmética del **segundo argumento**
- **Operador Relacional:** Evalúa expresión aritmética de **los dos argumentos**

Evaluador Aritmético is /2 y Operadores Relacionales

■ Ejemplos:

?- 5 is 2+3.

?- 8 is 2+3.

?- X is 4.0*7.

?- R is 2+X.

?- X = 1+2, Y is X.

?- A is a.

?- 2+7 >= 3.

?- 4 < -2.

?- 4*3 =:= 12.

?- X =\= 5.

Evaluador Aritmético is /2 y Operadores Relacionales

■ Ejemplos:

?- 5 is 2+3.
true

?- 8 is 2+3.
false

?- X is 4.0*7.
X = 28.0

?- R is 2+X.
ERROR

?- X = 1+2, Y is X.
X = 1+2
Y = 3

?- A is a.
ERROR

?- 2+7 >= 3.
true

?- 4 < -2.
false

?- 4*3 =:= 12.
true

?- X =\= 5.
ERROR

Evaluador Aritmético is /2 y Operadores Relacionales

■ Ejemplos:

?- 5 is 2+3.
true

?- 8 is 2+3.
false

?- X is 4.0*7.
X = 28.0

?- R is 2+X.
ERROR

?- X = 1+2, Y is X.
X = 1+2
Y = 3

?- A is a.
ERROR

Variable X no
instanciada

'a' no es una expresión
aritmética

?- 2+7 >= 3.
true

?- 4 < -2.
false

?- 4*3 =:= 12.
true

?- X =\= 5.
ERROR

Variable X no
instanciada