

Introducción a la Programación en Lógica

Lógica para Ciencias de la Computación

Primer Cuatrimestre de 2008

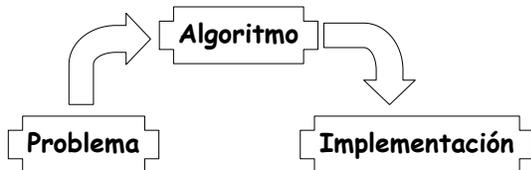
– Material Adicional –

Paradigmas de Programación

- Principales paradigmas:
 - Imperativo
 - Lógico
 - Funcional
 - Orientado a Objetos
- En esta asignatura introduciremos el paradigma lógico y el funcional.

Paradigma Imperativo

Proceso de resolución de un problema en el paradigma imperativo:



Paradigma Lógico

Proceso de resolución de un problema en el paradigma lógico:



Imperativo vs. Lógico

- En el paradigma imperativo:
 1. Comprender el problema y definir una solución viable.
 2. Codificar la solución recién encontrada.
- En el paradigma lógico:
 1. Expresar el conocimiento acerca del problema en un lenguaje apropiado.
 2. Dejar que el intérprete encuentre una solución al problema.

Lenguajes de Programación del Paradigma Imperativo

- Principales características:
 - Variables
 - Asignación
 - Secuencia
 - Repetición
 - Condicional

En el paradigma lógico, muchas de estas características están ausentes!

Prolog

- Prolog es el representante más conocido del paradigma lógico.
- El problema a ser resuelto se expresa mediante un conjunto de relaciones entre objetos.
- Un programa lógico permite:
 - Definir relaciones entre objetos.
 - Verificar si ciertos objetos están relacionados entre sí

Sintaxis Informal

nombres de relaciones
(también llamadas
predicados) y objetos.



secuencias de caracteres
comenzando con
minúscula.

- Prolog cuenta con variables. Las variables denotan objetos sin especificar.

nombres de variables



secuencias de caracteres
comenzando con **mayúscula**.

Sintaxis Informal

Existen 3 tipos construcciones en PROLOG (llamadas cláusulas) :

- **Hechos** } Permiten definir relaciones entre objetos.
- **Reglas** }
- **Consultas** } Permiten verificar si ciertos objetos están relacionados entre sí

Hechos

- Permiten establecer que una determinada tupla de objetos están relacionados bajo una relación en particular.
- Sintaxis:

$r(\text{obj}_1, \text{obj}_2, \dots, \text{obj}_n).$

"la tupla $(\text{obj}_1, \text{obj}_2, \dots, \text{obj}_n)$ pertenece a la relación r "

Ejemplos de Hechos

`padre_de(homero, bart).`

`padre_de(homero, lisa).`

`amigo_de(homero, barny).`

`amigo_de(flanders, X).`

Reglas

- Permiten establecer que una determinada tupla de objetos están relacionados bajo una relación en particular, pero en términos de otras relaciones.
- Sintaxis:

$r(\text{obj}_1, \dots, \text{obj}_n) :- r_1(\dots), \dots, r_m(\dots).$

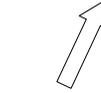
Ejemplos de Reglas

```
hermanos(bart, lisa):-  
  padre_de(homero, bart),  
  padre_de(homero, lisa).
```

```
hermanos(X,Y):-  
  padre_de(P, X),  
  padre_de(P, Y).
```

Terminología

$r(obj_1, \dots, obj_n) :- r_1(\dots), \dots, r_m(\dots).$



**Cabeza
(Head)**



**Cuello
(Neck)**



**Cuerpo
(Body)**

Programas Prolog

- Un programa **Prolog** se compone de hechos y reglas.

Ej:

padre_de(homero, bart).

padre_de(homero, lisa).

padre_de(abraham, homero).

hermanos(X,Y):- padre_de(P, X), padre_de(P, Y).

abuelo(X,Y):- padre_de(X,Z), padre_de(Z,Y).

definición del predicado
padre_de/2

definición del predicado
hermanos/2

definición del predicado
abuelo/2

Consultas

- Permiten verificar si una determinada tupla de objetos están relacionados bajo una relación en particular de acuerdo a un dado programa PROLOG.

- Sintaxis:

$?- r(obj_1, obj_2, \dots, obj_n).$

Ejemplos de Consultas

padre_de(homero, bart).

Programa

padre_de(homero, lisa).

amigo_de(homero, moe).

hermanos(X,Y):- padre_de(P, X), padre_de(P, Y).

$?- \text{padre_de}(\text{homero}, \text{bart}).$

yes

Ejemplos de Consultas

padre_de(homero, bart).

Programa

padre_de(homero, lisa).

amigo_de(homero, moe).

hermanos(X,Y):- padre_de(P, X), padre_de(P, Y).

$?- \text{amigo_de}(\text{homero}, \text{burns}).$

no

Ejemplos de Consultas

padre_de(homero, bart).

Programa

padre_de(homero, lisa).

amigo_de(homero, moe).

hermanos(X,Y):- padre_de(P, X), padre_de(P, Y).

?- amigo_de(homero, X).

X = moe

Ejemplos de Consultas

padre_de(homero, bart).

Programa

padre_de(homero, lisa).

amigo_de(homero, moe).

hermanos(X,Y):- padre_de(P, X), padre_de(P, Y).

?- hermanos(bart, lisa).

yes

FIN