Programación Lógica

Clase 9 . Introducción al paradigma lógico

PARADIGMAS DE PROGRAMACION *UTN - La Plata*

Temario

1. Características del Paradigma
2. Base de conocimiento
3. Consultas
4. Motor de inferencias
5. Tipos de reglas
6. Usos y aplicación
7. Prolog
8. Actividades

1. Características del paradigma

- En el paradigma lógico se trabaja en forma descriptiva: el programa NO indica cómo resolver el problema sino establece relaciones entre las entidades del mismo, las cuales describen ***qué*** hacer. Por esta razón se lo encuadra en la programación declarativa.

- Un programa lógico NO tiene un algoritmo que indique los pasos a seguir para llegar al resultado, sino que está formado por **expresiones lógicas que declaran o describen la solución** y es el sistema interno quien proporciona la secuencia de control en que se utilizan esas expresiones.

1. Características del paradigma

Tiene como característica principal el uso de reglas lógicas o predicados de 1º orden para inferir o derivar conclusiones a partir de datos.

Conociendo los datos y las condiciones del problema, la ejecución de un programa consiste en demostrar un objetivo a partir de las relaciones declaradas.

En este paradigma, un algoritmo lógico se construye:

Especificando **una base de conocimiento**

sobre la que se realizan **consultas**

y aplicando un **mecanismo de inferencia o deducción** sobre dicha base, **se infieren** conclusiones.

2. Base de conocimiento

- Se construye con **predicados de 1° orden llamados Cláusulas de Horn**.

- Cada cláusula permite deducir conclusiones a partir del conjunto de datos del problema.

- Una cláusula de Horn tiene la siguiente forma:

$$c := p1, p2, p3, p4, \ldots\ldots pk$$

donde **c** es una conclusión o consecuente y las $\mathbf{p_i}$ son las premisas o antecedentes.

Se dice que **c** es verdadera si son verdaderas cada una de las premisas $\mathbf{p_i}$.

2. Base de conocimiento

- Está formada por hechos y reglas.

- Los hechos representan enunciados o predicados siempre verdaderos que muestran relaciones entre los datos del problema. (axiomas)

- Las reglas muestran relaciones entre los datos, los hechos y otras reglas, y permiten deducir o demostrar objetivos.

- Hechos → hombre(juan).

  hombre(tomás).

  mujer(ana).

  progenitor(ana, juan).

- Reglas → mortal(X):- humano(X).

  cabeza — cuerpo

  que significa: X es mortal si X es humano

  Si X es humano entonces X es mortal.

  madre(X,Y):- mujer(X), progenitor(X,Y).

3. Consultas

- Una consulta es un objetivo a ser demostrado por comparación o pattern matching contra la base de conocimiento.

- Se trata de aparear la consulta con algún hecho o con la cabeza de una regla.

- La consulta puede ser de dos tipos:

  - validación (cuya respuesta es Si o No)

  - una búsqueda (que retorna los valores que hacen verdadero al objetivo).

3. Consultas

- Si puede demostrar la consulta u objetivo, responde:
  - **Yes** (si es una validación)
  - o **el/los valor/es** que satisface/n la consulta (en caso de búsqueda).

- Si no puede demostrar el objetivo, responde **No**, que tiene dos significados:
  - que es falso
  - o que no puede deducirlo con la información contenida en la base de conocimiento

3. Consultas

- Ejemplo:

?- hombre(juan)

Yes

?- hombre(luis)

No

?hombre(X)

X=juan;

X=tomás;

No

BC

hombre(juan).

hombre(tomás).

mujer(ana).

mujer(pilar).

progenitor(ana, juan).

Observación:
Si colocamos un ; después de un resultado, sigue mostrando otros resultados posibles hasta agotarlos todos.
Si colocamos un . el proceso de búsqueda se corta.

3. Consultas

- Si la consulta posee variables, las mismas se ***asocian o ligan*** con los valores de los hechos o con los argumentos de las cabezas de las reglas. Esto se denomina ***unificación.***
- No existe el concepto de asignación de valor a las variables. Tampoco el concepto de variable como celda de memoria, sino como variables matemáticas. Una vez que fueron ligadas a un valor, dicho valor no cambia. No hay efectos laterales.(propiedad de transparencia referencial)
- Al aparearse las variables de una consulta con la cabeza de una regla, las variables se ligan y luego se aplica la regla: se usa su definición y debe demostrarse cada sub-objetivo de la misma. Este proceso de demostración se denomina ***resolución***.

3. Consultas

- Ejemplo:

?-hombre(X)

X=juan;

X=tomás;

No

?- madre(ana,Z)

Z=juan;

No

BC

hombre(juan).

hombre(tomás).

mujer(ana).

mujer(pilar).

progenitor(ana, juan).

madre(X,Y):- mujer(X), progenitor(X,Y).

Se aparea la consulta con la cabeza de la regla:
liga X→ ana y Z→ Y, reemplaza en el cuerpo de la regla y la aplica.
mujer(ana),progenitor(ana,Z) y ahora tiene 2 subobjetivos nuevos a demostrar.

3. Consultas

Se pueden hacer 3 tipos de consultas:

- Sin variables
  - ?gusta(maria, juan) a maria le gusta juan Validación
- Con variables
  - ?gusta(maria, X) quién le gusta a maría Búsqueda
- Con variable anónima: para saber si existe algún objeto que haga verdadera la consulta
  - ?gusta(maria,_) hay alguien que le gusta a maría. La respuesta es Si o No

3. Consultas: reversibilidad el predicado

- Se pueden introducir valores en la consulta esperando que el motor de inferencias encuentre las variables de salida que apareen o unifiquen con la consulta.

```
?- progenitor(ana, X)
X=juan;
No
```

De quién es progenitor ana?

BC
mujer(ana).
hombre(juan).
progenitor(ana, juan).

- Recíprocamente también puedo introducir un valor de salida y esperar que el motor encuentre qué valores de entrada son adecuados para satisfacer esa salida.

```
?- progenitor(Y, juan)
Y=ana;
No
```

quién es el progenitor de juan?

4. Motor de inferencias

- Es quien controla la ejecución del programa lógico a partir de una consulta.

- Recorre la base de datos desde el principio y permite ***derivar o deducir*** un objetivo en base a dos mecanismos: ***recursión y backtracking***.

- Gracias al backtracking, si falla 1 camino de demostración, retrocede a un punto previo (deshace la última sustitución) y reinicia el proceso de demostración. Cuando agotó todos los caminos posibles, termina la demostración.

4. Motor de inferencias

eshombre(juan).

eshombre(pedro).

eshombre(luis).

quiere(juan,maria).

quiere(juan, elena).

quiere(pedro, raquel).

quiere(pedro, belen).

?-eshombre(X),quiere(X,Y)

X=juan Y=maría;
X=juan Y=elena;
X=pedro Y=raquel;
X=pedro Y=belen;
No

Proceso de resolución: evalúa de izq a derecha.

Toma 1er objetivo: eshombre(X):

Liga X=juan y reemplaza en el segundo objetivo

?quiere(juan,Y) busca todos los posibles valores de Y

X=juan Y=maría;

X=juan Y=elena; No hay más alternativas posibles

Por backtracking retrocede al objetivo previo: eshombre(X)

Liga X=pedro y reemplaza en el segundo objetivo

?quiere(pedro,Y)

X=pedro Y=raquel;

X=pedro Y=belen; No hay más alternativas posibles

Por backtracking retrocede al objetivo previo: eshombre(X)

Liga X=Luis y reemplaza en el segundo objetivo

?quiere(Luis,Y)

No encuentra ninguna cláusula en la BC que satisfaga el segundo objetivo, termina el proceso y muestra No.

5. Tipos de reglas

- Simples
  - notrabaja(belen).
  - bueno(pedro).
  - cuida(belen, pedro):- notrabaja(belen), bueno(pedro).

hechos

- Con variables
  - humano(juan).
  - humano(ana).
  - mortal(X):- humano(X).

hechos

- Recursivas
  - antepasado(X,Y):- progenitor(X,Y).
  - antepasado(X,Y):- antepasado(X,Z), progenitor(Z,Y).

Cómo programar en este paradigma

- A) Modelar el problema mediante una Base de Conocimiento formada por hechos que muestren las relaciones de los datos de dicho problema.

- B) Formular luego reglas que se cumplan en el mundo del problema modelado.

- C) Formular consultas al sistema para que demuestre o deduzca la respuesta a partir de la base de conocimiento usando resolución.

6. Usos y aplicación

- Se aplica a:

  Inteligencia Artificial

  Sistemas expertos

  Robótica

  Procesamiento de Lenguaje Natural

  Compiladores

  Bases de datos deductivas

  Acceso a bases de datos desde páginas web

7. PROLOG (PROgrammation in LOGique)

- Lenguaje interpretado, que consta de un compilador y un intérprete, un Shell (utilidad para probar programas) y una biblioteca
- Sintaxis:
  - Constantes y predicados comienzan con minúscula
  - Variables comienzan en mayúscula
  - Hechos y predicados terminan en punto
  - Se usa ; en una consulta para ver más resultados
  - El <enter> o el . finaliza la consulta
  - Los comentarios se empiezan con %

7. PROLOG (PROgrammation in LOGique)

- Predicados predefinidos:
  - true
  - fail
  - var(X) devuelve V si X es una variable sin instanciar
  - nonvar(X) devuelve T si X tiene valor asignado
  - integer(X) devuelve V si X es una variable entera
  - float(X) devuelve V si X es una variable real
  - number(X) devuelve V si X es una variable numérica
  - write(X) imprime el valor de X
  - read(X) ingresa un valor a X
  - == compara si son iguales
  - \== compara si son distintos

  Operadores: + - / * X mod Y

7. PROLOG (PROgrammation in LOGique)

Ejemplos de reglas:

- suma(X,Y,Z):- number(X), number(Y), Z is X+Y.

El operador **is** sirve para asignar valor o para comparar, dependiendo su uso.

- par(X):- integer(X), 0 is X mod 2.

  Esta regla devuelve true si 0 es el resultado de hacer X mod 2

- impar(X):- integer(X),1 is X mod 2.

- igual(X,Y):- X==Y.

- cuad(X,Y):- nonvar(X), number(X), Y is X*X.

7. PROLOG (PROgrammation in LOGique)

Ejemplos de reglas:

```
mayor(X,Y,Z):- X>Y, Z is X.
mayor(X,Y,Z):- X<Y, Z is Y.
mayor(X,Y,Z):- Z is 0.
```

Otra forma de hacerlo:

```
mayor(X,Y,X):- X>Y.
mayor(X,Y,Y):- X<Y.
mayor(X,Y,0):- X==Y.
```

```
posit(X):- X>0.
posit(X):-X<0,false.
posit(X):-X==0,false.
```

```
posit(X,Z):- X>0, Z is false.  %se usa la hipótesis de mundo cerrado
```

7. PROLOG (PROgrammation in LOGique)

Ejemplo de hechos y reglas:

%hechos

- gusta(martin,negro).
- gusta(diana,verde).
- gusta(lucas,azul).
- gusta(cecilia,blanco).
- gusta(soledad,violeta).
- color(pantalón,negro).
- color(pantalón,azul).
- color(camisa,verde).
- color(bufanda,violeta).
- color(saco,azul).
- %reglas

% X usa la prenda Y si le gusta el color de dicha prenda

- usa(X,Y) :- gusta(X,Z), color(Y,Z).

Consultas:

?-usa(lucas,R)

?- usa(martin,camisa)

?- usa(X,bufanda)

Cómo se llama la propiedad aplicada en la última consulta?

7. PROLOG (PROgrammation in LOGique)

Ejemplo de hechos y reglas:

%hechos

- padreDe(juan, maría). % Juan es padre de María
- padreDe(pablo, juan). % Pablo es padre de Juan
- padreDe(pablo, marcela). % Pablo es padre de Marcela
- padreDe(carlos, débora). % Carlos es padre de Débora

- %reglas

% A es hijo de B si B es padre de A

- hijode(A,B) :- padreDe(B,A).

8. Actividades

Defina un programa lógico para cada uno de los siguientes casos:

Usando la BC de la diapositiva previa:

1) A y B son hermanos si el padre de A es también el padre de B y si A y B no son la misma persona.

2) Dada la siguiente base de conocimiento:

```
mujer(pilar).
mujer(belen).
mujer(lucia).
mujer(ana).
mujer(maria).
hombre(tomas).
hombre(pedro).
hombre(jose).
progenitor(belen,pedro).
progenitor(ana,belen).
```

Definir el predicado madre(X,Y) sabiendo que X es madre de Y si es mujer y es progenitor de Y.

8. Actividades

3) En base a el ejercicio anterior responder las siguientes consultas:

?-madre(belen,pedro)
?-madre(X,belen)
?-madre(maría,Y)
?-madre(ana,_)
?-madre(X,Y)