

Paradigmas de Programación

Programación Lógica: Lógica de Predicados

Universidad Tecnológica Nacional
Facultad Regional Resistencia



Contenidos

- Introducción
 - Problema
 - Sistemas de Producción
 - Búsqueda
- Programación con Lógica Proposicional
 - Sintaxis
 - Consecuencia lógica
 - Resolución
 - Refutación y Deducción
 - Árbol de Resolución
 - Negación
 - Resolución SLD
- ***Programación con Lógica de Predicados***
 - ***Sintaxis***
 - ***Cláusulas***
 - ***Unificación***
 - ***Resolución***
- ***ProLog***
 - ***Ejemplos***



Introducción

- La principal debilidad de la lógica proposicional es su limitada habilidad de expresar conocimiento.
- Existen sentencias complejas que pierden mucho de su significado cuando se las representa mediante lógica proposicional.
- La lógica de predicados tiene acceso a los elementos constitutivos de cada proposición.
 - Ejemplo:
 - Proposición: `Lassie es un mamífero = q`
 - Predicado: `Mamifero (lassie)`



Introducción 2

- Las sentencias expresan **relaciones entre objetos**, así como también **cualidades y atributos de tales objetos**.
- Tales *cualidades, relaciones o atributos*, se denominan ***predicados***. Los *objetos* se conocen como ***argumentos*** o ***términos*** del predicado.
- Pueden evaluarse también como V o F.
- La veracidad depende de sus términos: un predicado puede ser V para un conjunto de términos y F para otros.
 - Ejemplo:
 - `color (yerba, verde)` V
 - `color (yerba, azul)` F



Introducción 3: Ejemplo

```
estrella (sol)
orbita (mercurio, sol)
orbita (venus, sol)
orbita (tierra, sol)
orbita (luna, tierra)
orbita (phobos, marte)
```

Al construir los predicados se asume que su veracidad está basada en su relación con el mundo real.

Ahora bien:

```
parte_de (argentina, europa)
```

Tales predicados, establecidos y asumidos como lógicamente verdaderos se denominan **axiomas**, y no requieren de justificación para establecer su verdad.



Constantes y Variables

```
estrella (sol)
orbita (mercurio, sol)
orbita (venus, sol)
orbita (tierra, sol)
orbita (luna, tierra)
orbita (phobos, marte)
```

En estos ejemplos los argumentos son constantes pero podrían ser variables. Por ejemplo en la consulta:

```
orbita (X, sol)?
```

Estaríamos preguntando cuales son los cuerpos que orbitan alrededor del sol?

Obtendríamos tres respuestas: X=mercurio, X=venus, X=tierra.



Sintaxis

En el Cálculo de Predicados se usan varios tipos de símbolos:

- Un conjunto de elementos llamado átomos.
 - *Caracteres en minúsculas y números: raquel, a, b, 3200.*
- Un vocabulario V de variables
 - *Se utilizarán letras MAYÚSCULAS: X, Y, Z.*
- Un vocabulario F de símbolos funcionales (functores)
 - *Se utilizarán letras MAYÚSCULAS: EMPLEADO (jaime, 32)*
- Un vocabulario P de símbolos predicativos.
 - *Se utilizarán letras minúsculas: cartero (juan)*



Definiciones

- Un término es:
 - Un átomo
 - Una variable
 - Un símbolo funcional seguido de una sucesión de términos (aridad).
- Un Predicado es:
 - Un símbolo predicativo seguido de una sucesión de términos (aridad).
- Una regla o cláusula es alguna de las siguientes alternativas:
 - A
 - $\neg B_1 \vee \dots \vee \neg B_n \vee A$
 - $\neg B_1 \vee \dots \vee \neg B_n$

Donde A, B_1, \dots, B_n son predicados.

Cláusulas de Horn

$\forall x_1, \dots, x_j \quad (A)$ **Tipo I**

Afirmación del predicado A

$\forall x_1, \dots, x_j \quad (B_1 \wedge \dots \wedge B_m \Rightarrow A)$ **Tipo II**

Si $B_1 \dots B_m$ son predicados ciertos, entonces A también lo es.

$\forall x_1, \dots, x_j \quad (\neg B_1 \vee \dots \vee \neg B_m)$ **Tipo III**

$\neg \exists x_1, \dots, x_j \quad (B_1 \wedge \dots \wedge B_m)$ **Tipo III**

Existen valores de los argumentos (x_1, \dots, x_j) tales que con dichos valores los predicados B_1 y \dots y B_m son ciertos?



Unificación: *Sustitución*

- El proceso de unificación soluciona el problema de cómo resolver dos predicados que tengan el mismo símbolo predicativo pero sus argumentos no coinciden.

Sustitución

Una sustitución es un conjunto de asignaciones del tipo $X := t$

dónde X es una variable y t es un término.

- *No puede existir más de una asignación a la misma variable.*
- *Una sustitución tiene alcance clausular.*

$\{ X := \text{juan}, Y := \text{noel} \}$
 $\{ W := Z, R := \text{EMPLEADO}(T, 3200) \}$
 $\{ Q := [], R := [X, Z] \}$



Sustitución: ***Aplicación***

Dada una sustitución θ y un predicado P , la aplicación de θ a P , produce un nuevo predicado que se denota $P\theta$, y que corresponde al predicado inicial P donde toda variable asignada en θ se cambia por el término correspondiente, y las otras variables permanecen sin cambios.

Ejemplo:

Dados: $p(X, Z) \wedge p(Z, Y) \Rightarrow a(X, Y)$ y $\theta = \{X:=\text{juan}, Y:=\text{pedro}\}$

Resulta:

$p(\text{juan}, Z) \wedge p(Z, \text{pedro}) \Rightarrow a(\text{juan}, \text{pedro})$



Unificador

Dadas dos expresiones del lenguaje definido (por ejemplo dos predicados) E_1 y E_2 . Se llama unificador, a una sustitución θ tal que cumple que:

$$E_1\theta = E_2\theta$$

Es decir que la aplicación de la sustitución a ambas expresiones da la misma expresión.



Unificador: *Ejemplos*

Encontrar un unificador para
padre (Z, diego) y padre (jorge, diego)
 $\theta = \{Z := \text{jorge}\}$

Encontrar un unificador para tio (X, diego) y tio (W, diego)
La sustitución $\theta = \{X := W\}$ es un unificador.
La sustitución $\theta = \{W := X\}$ es un unificador.
La sustitución $\theta = \{X := \text{guille}, W := \text{guille}\}$ es un unificador.

Encontrar un unificador para r (Z, diego) y r (diego, jorge)
No existe unificador para ambos.

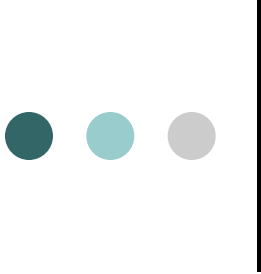


MGU: Unificador más general

Es el unificador que necesita asignaciones menos específicas de términos a variables.

Dadas dos sustituciones θ_1 y θ_2 y ambas son unificadores de las expresiones E_1 y E_2 , se dice que θ_1 es más general que θ_2 si existe una sustitución θ_3 tal que:

$$E_1 \theta_1 \theta_3 = E_2 \theta_2$$



Algoritmo de unificación de Robinson: Elementos I

○ Primer par de discordancia:

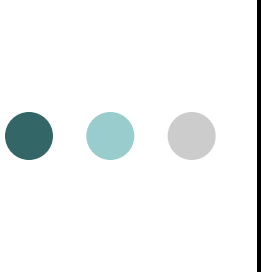
- En el símbolo funcional
- En el número de argumentos
- En alguno de sus argumentos

● Ej:

- $p(a, f(b, c), g(X))$
- $p(a, f(X, Y), g(h(Z)))$

- $p(f(a, b), c)$
- $p(g(X, Y))$

- $p(g(a, b, c), X)$
- $p(f(X, a), Z)$



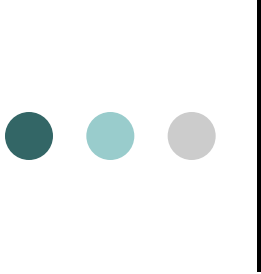
Algoritmo de unificación de Robinson: Elementos II

- Comprobación de Apariciones (occur check)
 - Determinar cuando una variable aparece dentro de un término.
 - Ej:
 - X aparece en $p(a, f(b, X))$
 - X aparece en $q(X, g(X, Y))$
 - X no aparece en $p(a, Z, g(a, b))$



Algoritmo de unificación de Robinson I

- Sean E y F dos términos que queremos unificar.
 - Sea $\theta_0 = \{ \}$
 - Sea $E_0 = E \theta_0$
 - Sea $F_0 = F \theta_0$
 - Sea $k = 0$
-
- **Paso 1:** si $E_k = F_k$ entonces las cláusulas E y F son unificables y un unificador de máxima generalidad es:
 - $\theta = \cup \theta_i$ donde $0 \leq i \leq k$
 - **Paso 2:** si $E_k \neq F_k$, se busca el primer par de discordancia entre E_k y F_k . Sea este D_k .



Algoritmo de unificación de Robinson II

- **Paso 3:** Si D_k posee una variable y un término (puede ser otra variable) pasamos al siguiente paso, en caso contrario los términos no son unificables y finaliza el proceso.
- **Paso 4:** Si la variable aparece en el término, se produce un *occur check* y los términos no unifican; finaliza el proceso. Si esto no ocurre pasamos al siguiente paso.
- **Paso 5:** Construimos una nueva sustitución que vincule la variable con el término de D_k . Sea esta sustitución θ_{k+1} .
Construimos ahora dos nuevos términos:
$$E_{k+1} = E_k \theta_{k+1}$$
$$F_{k+1} = F_k \theta_{k+1}$$
$$k = k + 1; \text{ Volvemos al paso 1.}$$

Algoritmo de unificación de Robinson. Ejemplo I

○ Ejemplo 1

- Sean los términos $p(a, X)$ y $p(X, Y)$
- Sea $\theta_0 = \{ \}$
- Sea $E = p(a, X)$
- Sea $F = p(X, Y)$
- Por lo tanto:
- Sea $E_0 = p(a, X)$
- Sea $F_0 = p(X, Y)$

Iteración 1

Paso 1: $E_0 \neq F_0$

Paso 2: $D_0 = \{a, X\}$

Paso 3: ir al paso 4

Paso 4: Ir al paso 5

Paso 5: $\theta_1 = \{X := a\}$

$E_1 = E_0 \theta_1 = p(a, a)$

$F_1 = F_0 \theta_1 = p(a, Y)$

Iteración 2

Paso 1: $E_1 \neq F_1$

Paso 2: $D_1 = \{a, Y\}$

Paso 3: ir al paso 4

Paso 4: Ir al paso 5

Paso 5: $\theta_2 = \{Y := a\}$

$E_2 = E_1 \theta_2 = p(a, a)$

$F_2 = F_1 \theta_2 = p(a, a)$

Iteración 3

Paso 1: $E_2 = F_2$

$\theta = \theta_2 \cup \theta_1 \cup \theta_0$

$\theta = \{X := a, Y := a\}$

Algoritmo de unificación de Robinson. Ejemplo II

○ Ejemplo 2

- Sean los términos

$$p(a, F(X, b), Y) \quad y \\ p(X, F(G(y), Z), T)$$

- Sea $\theta_0 = \{ \}$
- Sea $E = E_0 = p(a, F(X, b), Y)$
- Sea $F = F_0 = p(X, F(G(Y), Z), T)$

Iteración 1

Paso 1: $E_0 \neq F_0$

Paso 2: $D_0 = \{a, X\}$

Paso 3: ir al paso 4

Paso 4: Ir al paso 5

Paso 5: $\theta_1 = \{X := a\}$

$$E_1 = E_0 \theta_1 = p(a, F(a, b), Y)$$

$$F_1 = F_0 \theta_1 = p(a, F(G(Y), Z), T)$$

Iteración 2

Paso 1: $E_1 \neq F_1$

Paso 2: $D_1 = \{a, G(Y)\}$

Paso 3: Átomo y Símbolo Funcional.

No unifican.



Regla de Resolución

Supongamos que t_1, \dots, t_n y s_1, \dots, s_n son términos tales que t_i y s_i son unificables con un MGU θ (Most General Unifier) para $1 \leq i \leq n$ y C_1 y C_2 con cláusulas.

Definición:

$$C_1 \vee R(t_1, \dots, t_n), C_2 \vee \neg R(s_1, \dots, s_n) \vdash C_1\theta \vee C_2\theta$$

dónde R es un símbolo predicativo.

Regla de Resolución

Demostración

Tomemos una cláusula C1 de tipo (3) y C2 de tipo (2):

$$C1 = \neg Q_1 \vee \dots \vee \neg Q_n$$

$$C2 = \neg P_1 \vee \dots \vee \neg P_m \vee S$$

Podemos tener dos cláusulas:

$$C1 \vee R(t_1, \dots, t_n)$$

$$C2 \vee \neg R(s_1, \dots, s_n)$$

Podemos escribirlas como:

$$Q_1 \wedge \dots \wedge Q_n \Rightarrow R(t_1, \dots, t_n)$$

$$R(s_1, \dots, s_n) \wedge \dots \wedge P_1 \wedge P_m \Rightarrow S$$

Como t_1, \dots, t_n y s_1, \dots, s_n son unificables con un MGU θ , lo aplicamos a cada una de las dos cláusulas anteriores, para obtener:

$$Q_1\theta \wedge \dots \wedge Q_n\theta \wedge P_1\theta \wedge \dots \wedge P_m\theta \Rightarrow S\theta$$

$$\neg Q_1\theta \vee \dots \vee \neg Q_n\theta \vee \neg P_1\theta \vee \dots \vee \neg P_m\theta \vee S\theta$$

$$C1\theta \vee C2\theta$$

Ejemplo 1

1. padre (jorge, diego)
2. hermano (ricardo, jorge)
3. tio (X, Y) :- padre (Z, Y), hermano (X, Z)

Quien es el tío de Diego?

1. padre (jorge, diego)
2. hermano (ricardo, jorge)
3. tio (X, Y) \vee \neg padre (Z, Y) \vee \neg hermano (X, Z)
4. \neg tio (W, diego)
5. \neg padre (Z, diego) \vee \neg hermano (W, Z)
6. \neg hermano (W, jorge)
7. \square

$\theta = \{Y := \text{diego}, X := W\}$

$\theta = \{Z := \text{jorge}\}$

$\theta = \{W := \text{ricardo}\}$

Ejemplo 2

Dado el siguiente
programa.
(en notación Prolog)

1. $s(X) \text{ :- } q(Y), r(X, Y)$
2. $q(X) \text{ :- } p(X)$
3. $p(b)$
4. $r(a, b)$

Consulta: $s(a)$

1. $s(X) \vee \neg q(Y) \vee \neg r(X, Y)$

2. $q(X) \vee \neg p(X)$

3. $p(b)$

4. $r(a, b)$

5. $\neg s(a)$

Hipótesis agregada

6. $\neg q(Y) \vee \neg r(a, Y)$

Res por 1 y 5, $\theta = \{X := a\}$

7. $\neg p(Y) \vee \neg r(a, Y)$

Res por 2 y 6, $\theta = \{X := Y\}$

8. $\neg r(a, b)$

Res por 3 y 7, $\theta = \{Y := b\}$

9. \square

Res por 4 y 8.



Resolución SLD

Selection Rule - Linear Resolution – Definite Clauses

Para construir el árbol hay que determinar dos reglas:

- **Regla de selección:** determina qué fórmula atómica del objetivo utilizar a la hora de generar un nuevo nivel en el árbol SLD.
 - Izquierda a Derecha
 - Derecha a Izquierda
 - Aleatoriamente...
- **La regla de búsqueda:** determina cómo se seleccionan las cláusulas que unifican con una fórmula atómica dada.
 - Arriba hacia abajo
 - Abajo hacia arriba
 - Aleatoriamente
 - Empezando por los hechos
 - Empezando por las reglas que genere un unificador más pequeño

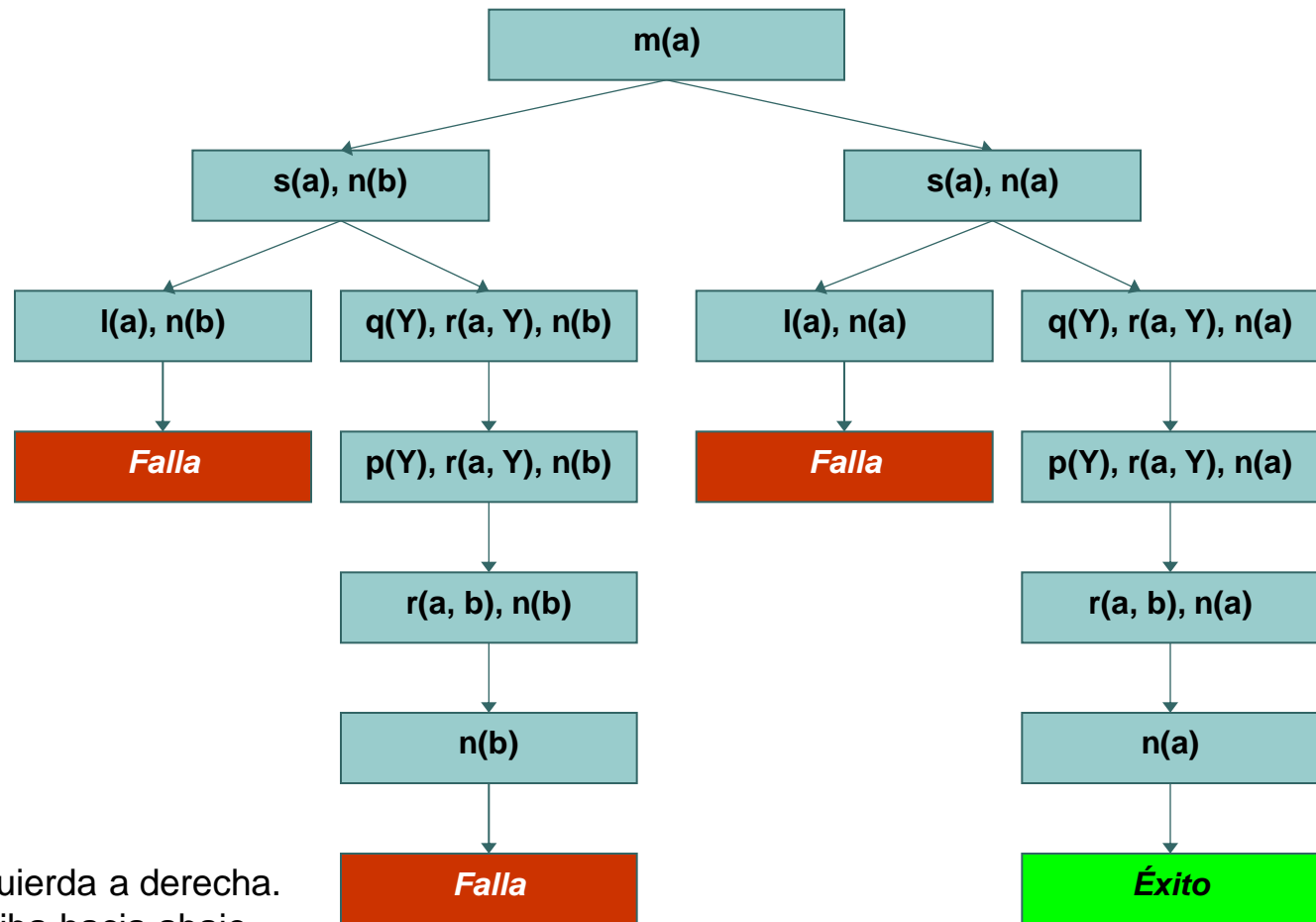
Ejemplo 3: SLD

Dado el siguiente programa:

$m(X) \text{ :- } s(X), n(b)$
 $m(X) \text{ :- } s(X), n(X)$
 $s(X) \text{ :- } l(X)$
 $s(X) \text{ :- } q(Y), r(X, Y)$
 $q(X) \text{ :- } p(X)$
 $p(b)$
 $r(a, b)$
 $l(c)$
 $n(a)$

Consulta: $m(a)$

Regla de Selección: de izquierda a derecha.
Regla de Búsqueda: de arriba hacia abajo.





SLDDraw

- prog1.pl Ejemplo anterior
- prog.pl Ejemplo de Parentesco
- prog2.pl Concatenación de Listas.
- prog3.pl Miembro de una lista.
- prog5.pl Recursión. Oficina.
- prog6.pl Regalo.



Manos a la obra 1

```
amigo(matias,X) :- chica(X), gusta(matias,X).  
amigo(matias,X) :- gusta(X,futbol).  
amigo(matias,X) :- gusta(X,rock).  
chica(paula).  
chica(julia).  
gusta(carlos,futbol).  
gusta(diego,rock).  
gusta(matias,julia).
```

- Quienes son las amigas de Matias?



Manos a la obra 2

padre(mario, jorge).

padre(jorge, emilio).

padre(jorge, pedro).

padre(pedro, juan).

ancestro(X,Y):-padre(X,Y).

ancestro(X,Y):-padre(Z,Y), ancestro(X,Z).

- Quiénes son los ancestros de Juan?