5.2 Introducción a Prolog

Lenguaje Prolog

- Su nombre viene de ***Programación en Lógica***, creado a comienzos de los ´70:
  - Robert Kowalski (Edimburgo): lado teórico
  - Maarten van Emden (Edimburgo): demostración práctica
  - Alain Colmerauer (Marsella): Implementación
- Popularidad se debe a David Warren (Edimburgo), que a mediados de los ´70 realizó implementación eficiente.

Características de Prolog

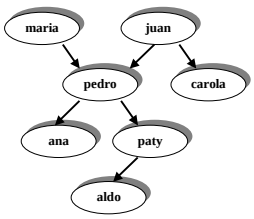
- Basado en Lógica y programación declarativa
- Produce estilo de programación orientado a ***metas***
- No se especifica ***cómo*** debe hacerse, sino ***qué*** debe lograrse (alto nivel)
- El programador se concentra más en el conocimiento que en los algoritmos
  - ¿Qué es conocido? (hechos y relaciones)
  - ¿Qué preguntar? (cómo resolverlo)

Aplicaciones de Prolog

- Pruebas Matemáticas
  - Demostración de teoremas
- Inteligencia Artificial
  - Sistemas Expertos
- Consultas a base de datos
  - Permite inferir relaciones no especificadas a priori

Hechos en Prolog: Ejemplo

```
padre(maria, pedro).
padre(juan, pedro).
padre(juan, carola).
padre(pedro, ana).
padre(pedro, paty).
padre(paty, aldo).
```



Consultas en Prolog

?- padre(pedro, ana).
=> yes

?- padre(ana, paty).
=> no

?- padre(X, carola).
=> X = juan

?- padre(pedro, X).
=> X = ana ;
=> X = paty ;
=> no

Consulta: Ejemplo 1

Preguntar por el abuelo de **aldo**:

$\exists X, Y : (X \text{ es padre de } Y) \cap (Y \text{ es padre de aldo})$

que se expresa en Prolog como:

```
?- padre(X, Y), padre(Y, aldo).
=>      X = pedro
        Y = paty
```


Consulta: Ejemplo 2

Preguntar por los nietos de **juan**:

$\exists X, Y : (\text{juan es padre de } X) \cap (X \text{ es padre de } Y)$

que se expresa en Prolog como:

```
?- padre(juan, X), padre(X, Y).
=>      X = pedro
        Y = ana ;

=>      X = pedro
        Y = paty
```


Consulta: Ejemplo 3

Preguntar si **ana** y **paty** tienen un padre en común:

$\exists X : (X \text{ es padre de ana}) \cap (X \text{ es padre de patricio})$

que se expresa en Prolog como:

```
?- padre(X, ana), padre(X, paty).
=> X = pedro
```

Otros Hechos

Agregar cláusulas sobre el sexo de las personas (relaciones unarias):

```
femenino(maria).
masculino(juan).
masculino(pedro).
femenino(carola).
femenino(ana).
femenino(paty).
masculino(aldo).
```


Alternativa de definición de hechos

Podría haberse definido también con una relación binaria:

```
sexo(maria, femenino).
sexo(juan, masculino).
sexo(pedro, masculino).
sexo(carola, femenino).
sexo(ana, femenino).
sexo(paty, femenino).
sexo(aldo, masculino).
```

¡A continuación usaremos la forma unaria!

Reglas en Prolog

- La relación:
  a ⊂ b
- se expresa en Prolog como:
  a :- b.
- Una cláusula de este tipo se denomina **regla**, que tiene la siguiente estructura:
  - la **cabeza** (parte izquierda de :-) es la **conclusión**
  - la proposición definida en el **cuerpo** (parte derecha de :-)

Resolución Simple

- La relación ***hijo de*** corresponde a:
  ∀X, Y : (Y es hijo de X) ⊂ (X es padre de Y)
- que se expresa en Prolog como:
  hijo(X, Y) :- padre(Y, X).

- **Ejemplo:** la meta siguiente es evaluada como:

La meta:	`hijo(paty, pedro)`
se convierte en submeta	`padre(pedro, paty)`
Se busca este hecho:	`yes`

Ejemplo de Reglas

Se puede definir ahora varias nuevas reglas como:

```
papa(X, Y)      :- padre(X, Y), masculino(X).
mama(X, Y)      :- padre(X, Y), femenino(X).
abuelo(X, Y)    :- padre(X, Z), padre(Z, Y).
hermana(X, Y)   :- padre(Z, X), padre(Z, Y), femenino(X).
```


Ejemplo de Consulta

```
?- hermana(ana, paty).
   => yes

?- hermana(X, paty).
   => X = ana ;
   => X = paty
```

oops ... paty es hermana de ella misma

¡Falta excluir este caso:

```
hermana(X, Y)   :- diferente(X, Y), padre(Z, X),
                   padre(Z, Y), femenino(X).
```

Observaciones

- Programas Prolog se extienden simplemente agregando más cláusulas
- Cláusulas son de tres tipos: ***hechos***, ***reglas*** y ***consultas***
- **Reglas** declaran cosas que cuya verdad depende de otras condiciones
- Por medio de **consultas** el usuario puede solicitar al programas que establezca qué cosas son verdad
- Una **cláusula** tiene una **cabeza** y un **cuerpo**. El cuerpo son metas separadas por comas (***conjunción***)
  - **Hechos** son cláusulas que no tienen cuerpo
  - **Preguntas** sólo tienen cuerpo
  - **Reglas** tienen cabeza y cuerpo
- Una evaluación puede sustituir una **variable** X por otro objeto (se dice que X se ***instancia***)
- Variables se cuantifican universalmente (∀)

Reglas Recursivas

La relación **antepasado** se define sobre la base de una regla de descendencia directa y otra regla de descendencia indirecta:

∀X, Z : (X es un antepasado de Z), si
{X es padre de Z } ∨
{ ∃Y: (X es padre de Y) ∧ (Y es antepasado de Z) }

Lo que en Prolog se expresa como :

```
antepasado(X, Z) :- padre(X, Z).          % descendiente directo
antepasado(X, Z) :- padre(X, Y), antepasado(Y, Z). % descendiente ind.
```


Ejemplo de Consulta

```
% Consultar por los descendientes de maria

?- antepasado (maria, X)
   => X = pedro ;
   => X = ana ;
   => X = paty ;
   => X = aldo
```

Resolución de Consulta