Paradigmas de la Programación
Práctico 8: Programación Lógica

Laura Alonso Alemany Cristian Cardellino Ezequiel Orbe

4 de junio de 2015

1. Cuáles de los siguientes pares de términos unifican? En el caso de que unifiquen, indique cómo se instancian las variables.

   a) `pan = pan`
   b) `'Pan' = pan`
   c) `'pan' = pan`
   d) `Pan = pan`
   e) `pan = salchicha`
   f) `comida(pan) = pan`
   g) `comida(pan) = X`
   h) `comida(X) = comida(pan)`
   i) `comida(pan,X) = comida(Y,salchicha)`
   j) `comida(pan,X,cerveza) = comida(Y,salchicha,X)`
   k) `comida(pan,X,cerveza) = comida(Y,hamburguesa)`
   l) `comida(X) = X`
   m) `meal(comida(pan),bebida(cerveza)) = meal(X,Y)`
   n) `meal(comida(pan),X) = meal(X,bebida(cerveza))`

2. A partir de la siguiente base de datos:

```
elfo_domestico(dobby).
bruja(hermione).
bruja('McGonagall').
bruja(rita_skeeter).
puede_hacer_magia(X):- elfo_domestico(X).
puede_hacer_magia(X):- hechicero(X).
puede_hacer_magia(X):- bruja(X).
```

   Cuáles de las siguientes consultas se satisfacen? Con qué instanciaciones de variables?

```
?- puede_hacer_magia(elfo_domestico).
?- hechicero(harry).
?- puede_hacer_magia(hechicero).
?- puede_hacer_magia('McGonagall').
?- puede_hacer_magia(Hermione).
```

   Dibuje el árbol de búsqueda para la consulta `puede_hacer_magia(Hermione).`

3. La búsqueda en Prolog es exhaustiva, mientras que en otros lenguajes se ejecuta la sentencia controlada por el primer patrón que hace pattern matching. Sin embargo, existe una forma en prolog para cortar una búsqueda exhaustiva, usando el operador `cut`, como se ve en el siguiente ejemplo:

```
max(X,Y,Y)  :-  X  =<  Y,!.
max(X,Y,X)  :-  X>Y.
```

   Escriba un pseudocódigo en imperativo (usando `if ... then ... else ...`) o funcional (usando guardas) con la misma semántica.

4. Defina en Prolog el problema de recomendar "amigos" en una red social, especificado como sigue: para un usuario P con un conjunto de amigos $CA = [A_1, A_2, ...A_n]$, los cuales a su vez tienen cada uno asociado un conjunto de amigos C_1, C_2... C_n le recomendaremos a P todos los miembros de la unión de todos los conjuntos de amigos de sus propios amigos $CCA = C_1 \cup C_2 ... \cup C_n$ que NO están en la lista de sus propios amigos.

   Pueden usar la siguiente base de conocimiento.

```
amigo(pedro,maría).     amigo(pedro,juan).
amigo(maría,pedro).     amigo(maría,clara).
amigo(maría,romina).    amigo(juan,pedro).
amigo(juan,clara).      amigo(clara,maría).
amigo(clara,juan).      amigo(romina,maría).
```

5. Defina en Prolog un sistema de ayuda a una central de turnos. Este sistema dirá si se le puede asignar un turno a un paciente con un determinado médico para un determinado día, siguiendo las siguientes premisas:

   - el paciente no tiene ningún turno asignado para el mismo día a la misma hora.
   - el paciente no tiene ningún turno asignado con ningún especialista de la misma especialidad para la que pide turno.
   - el médico para el que pide turno no tiene turno asignado con ningún otro paciente para el mismo día a la misma hora.

   Se puede definir con una sola regla!

   La base de conocimiento puede ser como sigue:

```
turno(celia,rivas,(6,30,8)).
turno(celia,zilvetti,(7,14,11)).
turno(tomás,rivas,(7,11,10)).
turno(tomás,pérez,(8,11,10)).
turno(tomás,schuster,(9,11,10)).
turno(lidia,zilvetti,(7,14,10)).
turno(lidia,schuster,(9,11,11)).
turno(esteban,rivas,(7,1,9)).

especialidad(rivas,oftalmologia).
especialidad(smith,oftalmologia).
especialidad(zilvetti,ginecología).
especialidad(román,ginecología).
especialidad(pérez,endocrinología).
especialidad(schuster,clínico).
```