Lógica Informática (Tecnologías Informáticas). 2 de diciembre de 2022

Relación 6: De Resolución a Prolog

Ejercicio 124. Decidir si el siguiente conjunto de fórmulas se puede reescribir de manera equivalente como un conjunto de cláusulas de Horn:

$$S = \{p \to q,\ \neg s \wedge r \to q,\ \neg q,\ q \leftrightarrow r \wedge s\}$$

Ejercicio 125. Escribe en Prolog (regla, hecho o pregunta) las siguientes frases:

1. María lee el libro de programación lógica del autor peter lucas.
2. A cualquiera le gusta ir de compras si es rico.
3. ¿A quién le gusta ir de compras?
4. Pedro odia cualquier ciudad grande y concurrida.

Ejercicio 126. Consideremos el siguiente programa Prolog:

```
robot(wall_e).
robot(otto).
robot(c3po).
robot(r2d2).

can_walk(c3po).
can_drive(r2d2).
can_drive(wall_e).

same_story(c3po,r2d2).
same_story(wall_e,otto).
same_story(X,Y):- robot(X), robot(Y), same_story(Y,X).

can_move(X) :- can_walk(X).
can_move(X) :- can_drive(X).
```

Construye las consultas Prolog correspondientes a las siguientes preguntas:

1. ¿Cuáles son los robots de los que hablamos?
2. ¿Qué robots pueden tanto caminar como conducir?
3. ¿Qué pares de robots pueden moverse y forman parte de la misma historia?
4. ¿Qué robots comparten historia con un robot que camina?

Ejercicio 127. Consideremos la siguiente base de datos en Prolog:

```
a(a1,1).
a(A,2).
a(a3,N).

b(1,b1).
b(2,B).
b(N,b3).

c(X,Y) :- a(X,N), b(N,Y).

d(X,Y) :- a(X,N), b(Y,N).
d(X,Y) :- a(N,X), b(N,Y).
```

Predice las respuestas a las siguientes consultas, y después compruébalas con Prolog, describiendo la deducción:

1. `?- a(X,2).`
2. `?- b(X,kalamazoo).`
3. `?- c(X,b3).`
4. `?- c(X,Y).`
5. `?- d(X,Y).`
6. `?- d(X,Y),c(X,Y).`

Ejercicio 128. Consideremos el programa Prolog:

```
suma(0,Y,Y).                          % R1
suma(s(X),Y,s(Z)) :- suma(X,Y,Z).     % R2
suma(X,s(Y),s(Z)) :- suma(X,Y,Z).     % R2
```

1. Describe el árbol de deducción del programa para la pregunta

   `?- suma(0,s(0),X).`

2. Describe el árbol de deducción del programa para la pregunta

   `?- suma(s(0),s(0),X).`

3. Obtén, a partir del árbol del apartado anterior, una refutación de la respuesta obtenida.

Ejercicio 129. Consideremos el siguiente programa Prolog para obtener el país al que pertenece un estado:

```
located_in(X,usa) :- located_in(X,georgia).          % Regla 1
located_in(X,usa) :- located_in(X,texas).            % Regla 2
located_in(X,canada) :- located_in(X,ontario).       % Regla 3
located_in(X,north_america) :- located_in(X,usa).    % Regla 4
located_in(X,north_america) :- located_in(X,canada). % Regla 5
```

y los hechos:

```
located_in(atlanta,georgia). % Hecho 1
located_in(houston,texas).   % Hecho 2
located_in(austin,texas).    % Hecho 3
located_in(toronto,ontario). % Hecho 4
```

1. Describe el árbol de deducción para la pregunta

   ```
   ?- located_in(toronto,north_america).
   ```

2. Obtén, a partir del árbol del apartado anterior, una refutación de la respuesta obtenida.

Ejercicio 130. Consideremos el siguiente programa Prolog:

```
male(phil).
female(liz).
parent(phil, chas).
parent(liz, chas).
male(harry).
parent(chas,harry).
grandmother(GM, C):- mother(GM, P), parent(P, C).
mother(M,C):- female(M), parent(M,C).
```

1. Describe el árbol de deducción para la pregunta `?-mother(liz,chas).`
2. Describe el árbol de deducción para la pregunta `grandmother(liz,harry).`
3. Obtén, a partir del árbol del apartado anterior, una refutación de la respuesta obtenida.

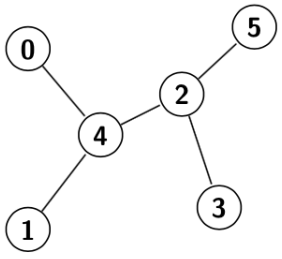
Ejercicio 131. Amplía el programa del apartado anterior añadiendo reglas para las siguientes relaciones familiares (añada más personas si es necesario, para que pueda comprobar sus resultados):

1. hermano(X, Y) donde X es el hermano de Y
2. hermana(X, Y) donde X es la hermana de Y
3. hijo(X, Y) donde X es el hijo de Y
4. hija(X, Y) donde X es la hija de Y
5. casado(X, Y) donde X está casado con Y
6. ancestro(X, Y) donde X es el ancestro de Y

1. ¿Qué problemas encuentras y cuál es la naturaleza de los problemas? ¿Qué soluciones (si las hay) puede sugerir?
2. Supongamos que disponemos el predicado `diferentes(X,Y)` para denotar que X es diferente a Y ¿Qué hechos habría que añadir? ¿Cuáles de las definiciones anteriores no resueltas se pueden obtener?
3. En Prolog existe el predicado `X =\= Y` para poner en la cola de la regla que X e Y son distintos. Arregla las reglas del apartado 1 que no eran correctas para que calculen bien los resultados.

Ejercicio 132. En este ejercicio se trabajará con representaciones de grafos. Con el predicado `arista(X,Y)` para representar que existe una arista del nodo X al Y, se pide:

1. Representa el grafo:



2. Diseña un programa que sirva para preguntar sobre el predicado `conectado(X,Y)`, que es cierto si existe un camino de x a Y
3. Describe el arbol de deducción de tu programa para la pregunta `conectado(0,1).`
4. ¿Qué pregunta le podríamos hacer a tu programa para, dado un grafo, nos determine si es acíclico?

Ejercicio 133. Dada la siguiente definición de número natural:

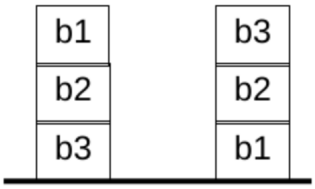
```
natural(0).
natural(s(X)) :- natural(X).
```

Escriba un programa Prolog para los siguientes predicados:

1. `menor_que(X,Y)` que se cumple si y sólo si X es menor que Y.
2. `even(X)` que se cumple si y sólo si X es par.
3. `sum(X,Y,Z)` que se cumple si y sólo si Z es la suma de X e Y.
4. `product(X,Y,Z)` se cumple si y sólo si Z es el producto de X e Y
5. `factorial(X,Y)` se cumple si y sólo si Y es el factorial de X.
6. `exp(N,X,Y)` se cumple si y sólo si $Y = X^N$.

Ejercicio 134. Un mundo de bloques puede describirse mediante una colección de hechos en `on(Bloque1,Bloque2)`, que es verdadera si $Bloque1$ está inmediatamente encima de $Bloque2$.

1. Representa el mundo



2. Defina el predicado `encima(X,Y)` que exprese que el bloque X está apilado por encima de Y.
3. Describe el arbol de deducción para la pregunta `?-encima(b3,b1).`
4. ¿Cuáles son las respuestas para `?-encima(X,Y).`

Ejercicio 135. Consideremos el siguiente programa, que intenta obtener las parejas de casados en cualquier orden:

```
married(X,Y) :- married(Y,X).
married(abrahaxn,sarah).
```

1. Demostrar que el árbol de deducción es infinito para la pregunta `?- married(abrahaxn,sarah).`
2. Demostrar que el siguiente programa siempre para cualquier pregunta cerrada:

```
are_married(X,Y) :- married(X,Y).
are_married(X,Y) :- married(Y,X).
```

Ejercicio 136. Consideremos el programa Prolog que define el elemento neutro y el inverso en una estructura (representando con `operacion(X,Y,Z)` la ecuación $X * Y = Z$:

```
operacion(X,e,X).  % Elemento neutro
operacion(e,X,X).  % Elemento neutro

operacion(X,Y,e) :- inverso(X,Y).
operacion(Y,X,e) :- inverso(X,Y).

 inverso(X,Y) :- operacion(X,Y,e), operacion(Y,X,e). % "definición" del inverso
```

1. Demuestra que `operacion(e,e,e)` es cierto, obteniendo una rama éxito del árbol de deducción ¿Es un árbol infinito?
2. Demuestra que `inverso(e,e)` es cierto, obteniendo una rama éxito del árbol de deducción ¿Es un árbol infinito?

Ejercicio 137. En Prolog existen dos predicados predefinidos para la falsedad y verdad, `true,false`. Consideremos el siguiente programa Prolog:

```
foo(a).
foo(b).
foo(X) :- foo(X).
```

Usando un intérprete de Prolog, considere las siguientes preguntas. ¿Qué respuesta(s) producirán? ¿Terminarán? Interpreta los resultados

1. `?-foo(Y).`
2. `?-foo(Y), true.`
3. `?-foo(Y), false.`
4. `?-true, foo(Y).`
5. `?-false, foo(Y).`
6. Si ponemos len primer lugar la regla ¿Qué respuesta produce para `?-foo(Y).`?