INF-253 Lenguajes de Programación
Tarea 5: Prolog

Profesor: José Luis Martí, Jorge Díaz Matte, Rodrigo Salas Fuentes
Ayudante Cátedras: Gonzalo Severín Muñoz, Jhossep Martínez Velásquez
Ayudante Coordinador: Álvaro Rojas Valenzuela
Ayudante Tareas: Ricardo Barriga Vera, Bryan González Ramírez, Bastián Salomón Ávalos, Cristian Tapia Llantén, Cristóbal Tirado Morales

14 de junio de 2024

1. Objetivo

En esta tarea existen múltiples problemas a resolver usando lenguaje de programación Lógica. Específicamente Prolog. Se debe usar específicamente SWI-Prolog que se puede encontrar aquí:
https://www.swi-prolog.org/download/stable

2. Problemas a Resolver:

2.1. El rango del Ejército

Un general encargo de un ejército necesita una información importante para ganar la guerra. Él te da una lista desordenada de los puntos de defensa de todos los soldados, y te pide que, mediantes operaciones lógicas, obtengas el rango de la defensa del ejército.

Para ello tienes que implementar el predicado `obtenerRango(L, R)`, donde `L` es una lista desordenada de los puntos de defensa de cada soldado. Y `R` toma el valor del rango.

El rango de una lista de números S corresponde a: $max\{S_i\} - min\{S_i\}$.

Ejemplo:

```
?- obtenerRango([5, 2, 3, 1, 9], R).
R = 8.

?- obtenerRango([5], R).
R = 0.

?- obtenerRango([3, 2, 1], 2).
```

```
true.

?- obtenerRango([5, 10, 2, 1], 10).
false.

?- obtenerRango([], R).
R = 0.
```

Nota: Obtener el rango de una lista vacía es igual a 0.

2.2. Distancia Manhattan

La Distancia Manhattan entre dos puntos $P_1(x_1, y_1)$ y $P_2(x_2, y_2)$ es igual a:

$$\text{distManhattan}(P_1, P_2) = |x_1 - x_2| + |y_1 - y_2|$$

De un punto fijo P y de una lista L de puntos definidos con el functor `punto(X, Y)`, obtén el mínimo de las Distancias Manhattan entre el punto P y los puntos de la lista L, es decir:

$$minDistManhattan = min\{distManhattan(P, L_i)\}$$

Para ello tienes que implementar el predicado `minDistManhattan(punto(A, B), L, R)`, donde A y B son las coordenadas del punto P, L es la lista de puntos descrita anteriormente, y R calza con el mínimo de las Distancias Manhattan entre el punto P y la lista de puntos L.

Ejemplo:

```
?- minDistManhattan(punto(3, 4), [punto(6, 7), punto(9, 10), punto(-2, 4)],
     R).
R = 5.

?- minDistManhattan(punto(1, 10), [punto(2, 3), punto(-2, -4)], 8).
true.

minDistManhattan(punto(1, 3), [], R).
R = inf
```

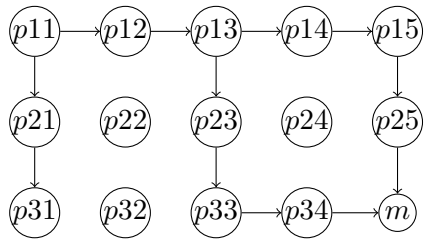
Nota: El mínimo de las Distancias Manhattan de una lista vacía es infinita.

2.3. El Laberinto

Un laberinto compuesto de caminos (.) y muros (*) se puede representar fácilmente como un grafo dirigido. Además, hay un camino marcado como '`m`' representado como la meta. Pero en este laberinto tienen una restricción, solo se puede mover hacia la derecha y hacia abajo.

Por ejemplo, el siguiente laberinto de tamaño `3x5`, donde cada camino y muro está enumerado de la forma p_{ij}, con $i = \{1, 2, 3\}$ y $j = \{1, 2, 3, 4, 5\}$. Su grafo equivalente sería:

```
. . . . .
. * . * .
. * . . m
```



Tu objetivo es construir reglas lógicas para decir si desde un punto cualquiera se puede llegar a la meta.

Al comienzo del programa se crearán las reglas camino($p_{i,j}, p_{u,v}$), donde $p_{i,j}$ tienen un camino a $p_{u,v}$, para construir un laberinto de prueba. E implementa el predicado `verificar(S)`, donde S es un nodo cualquiera, y debe retornar true si se puede llegar a la meta, o false lo contrario.

```
camino(p11, p12).
camino(p12, p13).
camino(p13, p14).
camino(p14, p15).
camino(p15, p25).
camino(p25, m).

camino(p11, p21).
camino(p21, p31).

camino(p13, p23).
camino(p23, p33).
camino(p33, p34).
camino(p34, m).
```

Las consultas ejemplos serían:

```
?- verificar(p11).
true.

?- verificar(p21).
false.

?- verificar(m).
true.
```

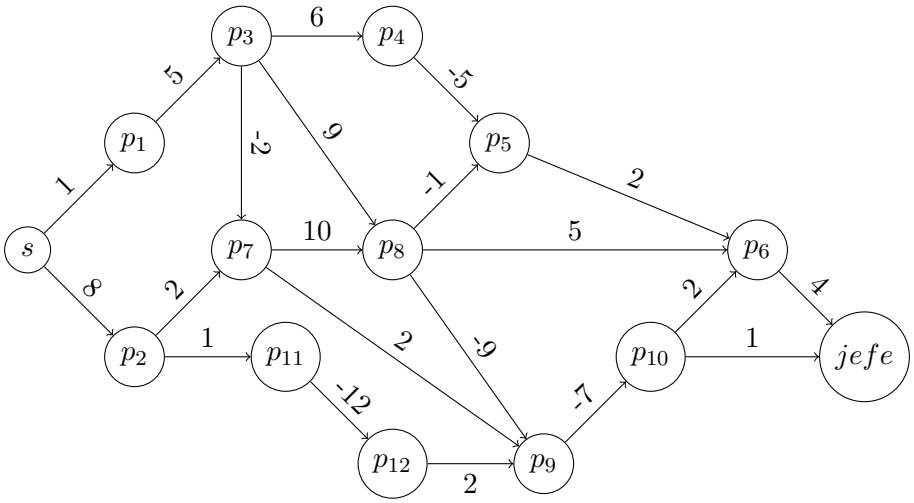
Nota: Se va a probar con diferentes laberintos remplazando los predicados `camino()`, siempre va a existir un nodo meta '`m`'.

2.4. El Camino De Fuerzas y el Búho Gigante

Jaime inicia su aventura para derrotar al temible Búho gigante. Durante su viaje se encuentra con un mapa mágico. Este le muestra todos los caminos e intersecciones para llegar al jefe. Y además, le muestra todos los puntos de fuerzas que puede ganar y perder al viajar de una intersección a otra. Jaime necesita que encuentres todos los caminos posibles para llegar al Búho gigante, pero con dos requerimientos:

1. En ningún momento, la fuerza de Jaime sea 0 o menos.
2. La fuerza final recolectada es la necesaria para derrotar al Búho.

El mapa mágico muestra el siguiente grafo dirigido:



Para ayudar a Jaime, primero crear las relaciones necesarias para construir el mapa mágico. Y luego implemente un predicado llamado `caminosDeFuerzas(I, O, R)`, donde `I` es la fuerza inicial de Jaime, `O` es la fuerza necesaria para derrotar al jefe, y R toma las listas de los caminos de fuerzas posibles para llegar desde `s` hasta el `jefe`.

Ejemplos:

```
?- caminosDeFuerzas(8,15,R).
R = [p1, p3, p4, p5, p6, jefe] ;
R = [p1, p3, p7, p8, p6, jefe] ;
R = [p1, p3, p7, p8, p5, p6, jefe] ;
R = [p1, p3, p8, p6, jefe] ;
R = [p1, p3, p8, p5, p6, jefe] ;
R = [p2, p7, p8, p9, p10, p6, jefe] ;
R = [p2, p7, p8, p6, jefe] ;
R = [p2, p7, p8, p5, p6, jefe] ;
R = [p2, p7, p9, p10, p6, jefe] ;
```

```
false.

?- caminosDeFuerzas(10, 15, [p2, p7, p9, p10, jefe]).
true.

?- caminosDeFuerzas(15,23,R).
R = [p1, p3, p4, p5, p6, jefe] ;
R = [p1, p3, p7, p8, p6, jefe] ;
R = [p1, p3, p7, p8, p5, p6, jefe] ;
R = [p1, p3, p8, p6, jefe] ;
R = [p1, p3, p8, p5, p6, jefe] ;
R = [p2, p7, p8, p9, p10, p6, jefe] ;
R = [p2, p7, p8, p6, jefe] ;
R = [p2, p7, p8, p5, p6, jefe] ;
R = [p2, p7, p9, p10, p6, jefe] ;
false.
```

Nota: La fuerza total obtenida en al finalizar el viaje tiene que ser mayor o igual a la fuerza del jefe.
Importante: No se aceptarán respuestas ad-hoc, es decir, enlistar mediantes reglas todos los caminos válidos para una cierta combinación de valores I y O. Significaría puntaje 0 en la pregunta.

3. Sobre la Entrega

- Se deberá entregar un archivo por cada problema solicitado, con el siguiente formato de nombres:
  - `p1.pl, p2.pl, p3.pl, p4.pl`
- Los códigos deben venir comentados, explicando clara y brevemente lo que realiza. Se deja libertad al formato del comentario.
- Puede crear otros predicados auxiliares que le permitan resolver el problema.
- Cuidado con el orden y la indentación de su tarea, llevará descuento de los más 20 puntos.
- Se debe trabajar de forma individual obligatoriamente.
- **La entrega debe realizarse en un archivo comprimido en tar.gz y debe llevar el nombre: Tarea5LP_RolAlumno.tar.gz**.
- **El archivo README.txt debe contener nombre y rol del alumno, e instrucciones detalladas para la correcta utilización**.

- La entrega será vía aula y el plazo máximo de entrega es hasta el **24 de junio a las 23:55 hora aula**.
- Por cada día de atraso se descontarán 25 puntos (10 puntos dentro la primera hora).
- Las copias serán evaluadas con nota 0 y se informarán a las respectivas autoridades.

4. Clasificación

4.1. Entrega

Para la calificación de su tarea, debe realizar una entrega con requerimientos mínimos que otorgarán 30 pts base, luego se entregará puntaje dependiendo de los otros requerimientos que llegue a cumplir.

4.1.1. Entrega Mínima

La entrega mínima deberá tener la implementación del predicado que realiza lo pedido en **El Rango del Ejército**, resolviendo correctamente los casos de prueba.

4.1.2. Entrega

Luego de cumplir con la Entrega Mínima, puede obtener más puntaje cumpliendo con los siguientes puntos:

- Implementación de funciones para la correcta resolución de los problemas (Total 70 pts)
  - Implementa **Distancia Manhatta** (20 pts)
  - Implementa **El Laberinto Posible o Imposible** (20 pts)
  - Implementa **El Camino De Fuerzas y el Búho Gigante** (30 pts)
    - Creación de reglas para crear el mapa. (5 pts)
    - Identifica todos los caminos de fuerzas posibles desde s hasta el $jefe$. (25 pts)

Para todos los predicados, existe puntaje parcial acorde a los casos de pruebas que resuelve.

4.2. Descuentos

- Falta de comentarios (-5 pts c/u, Max 20 pts)
- Falta de README (-20 pts)
- Falta de alguna información obligatoria en el README (-5 pts c/u)
- Falta de orden (entre -5 y -20 pts)
- Día de atraso (-20 pts por día, -10 dentro de la primera hora)
- Mal nombre en algún archivo entregado (-5 pts c/u)