Universidad Politécnica de San Luis Potosí
Academia de Computación
Inteligencia Artificial I
Proyecto 1.

Dr. Omar Montaño
Urbano Villalón núm. 500, Col Ladrillera, C.P. 78363
Universidad Politécnica de San Luis Potosí
San Luis Potosí, SLP
omar.montano@upslp.edu.mx

Objetivos

Introducir al alumno al área de Búsqueda Informada.

1. Búsqueda Informada

Considere el problema del 8-puzzle visto en clase. En este juego, las casillas numeradas del 1 al 8 se mueven en una malla de 3 x 3. Cualquier casilla adyacente a la posición vacía puede moverse a la posición vacía. El juego consiste en encontrar una secuencia de movimientos para llegar a una configuración objetivo o meta. Por ejemplo, en la figura 1 se muestran tres configuraciones del juego: la configuración b) se puede alcanzar a partir de a) moviendo la casilla 5 a la izquierda; la configuración c) se puede alcanzar de b) moviendo la casilla 6 para arriba. La configuración c) es la configuración meta.

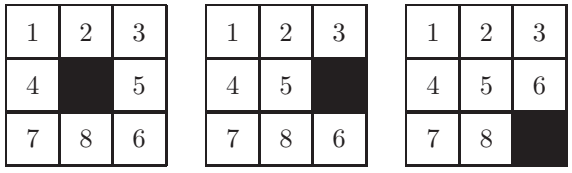

Figura 1: Tres configuraciones distintas en el 8-puzzle.

Considere las dos heurísticas discutidas en clase.

1. Numero de casillas mal colocadas: número de casillas mal colocadas (excluyendo a la posición vacía) con respecto a la configuración meta.
2. Distancia Manhattan: distancia Manhattan total de todas las casillas (excluyendo la posición vacía) con respecto a la configuración meta. Recuerde que la distacia Manhattan entre dos casillas es la suma de las distancias horizontales y verticales que existe entre las dos casillas.

Este proyecto consiste en que escribas dos programas que resuelvan el problema del 8-puzzle **usando** A^*.

1. El primer programa resolverá el problema del 8-puzzle usando la heurística 1.
2. El segundo programa resolverá el problema del 8-puzzle usando la heurística 2.

Los dos programas deben usar como función de costo $g(n)$ el número de movimientos realizados para obtener el estado n. Los dos programas deben leer la configuración inicial desde la entrada estándar y resolver el problema (si es que existe solución).

1.1. Formato de archivo de entrada

Tu programa debe ser capaz de leer una entrada (de la entrada estandar) que contenga 9 números (cero representando a la posición vacía), por ejemplo:

```
123405786
```

representa a la configuración a) de la figura 1.

1.2. Formato de salida

Tu programa debe desplegar (en la salida estandar) siempre que una solución al problema exista, cuantos pasos tomó el encontrar la solución, el factor de ramificación promedio y cual es la solución. Una salida de ejemplo puede ser:

```
Pasos: 2
Factor de ramificacion promedio: 1
Solucion:
R
D
```

El factor de ramificación promedio esta dado por la siguiente equación:

$$N = 1 + b + b^2 + \ldots + b^k \quad (1)$$

donde N es el número de nodos expandidos, k es el número de pasos que tomó resolver el problema (la profundidad de la solución) y b es el factor de ramificación promedio. Si multiplicamos la ecuación (1) por b obtenemos.

$$bN = b + b^2 + \ldots + b^k + b^{k+1} \quad (2)$$

Si restamos (1) de (2) e igualamos a 0 obtenemos:

$$bN - b = b^{k+1} - 1$$

e igualando a 0 obtenemos:

$$b^{k+1} + b(1 - N) - 1 = 0 \quad (3)$$

La ecuación (3) puede ser usada para aproximar el valor de b usando métodos numéricos (por ejemplo, Newton-Raphson).

El método de Newton-Raphson consiste en iterar la siguiente formula hasta alcanzar cierta precisión (por ejemplo, $|x_{i+1} - x_i| < 0{,}00001$).

$$x_{i+1} = x_i - \frac{f(x_i)}{f'(x_i)}$$

donde f es alguna función de la variable que queremos aproximar y f' es su derivada (con respecto a la variable). Para este proyecto la equación (3) es la función f. Por tanto, la siguiente ecuación puede ser usada para aproximar los valores de b.

$$b_{i+1} = b_i - \frac{b_i^{k+1} + b_i(1 - N) - 1}{(k+1)b^k - N}$$

El siguiente algoritmo implementa dicha aproximación del factor de ramificación promedio b.

```
    Newton-Raphson(N, K, b)
1     f := b^{K+1} + x(1 − N) − 1
2     f' := (K + 1)b^K − N
2     b' := b − f/f'
3     if |b' − b| < 0,00001 then
4      return b'
6    else return Newton-Raphson(N, K, b')
```

Por último, la solución debe estar compuesta de todos los movimientos necesarios para llegar a la configuración meta. Las etiquetas `U`, `D`, `R` y `L` representan los movimientos arriba, abajo, derecha e izquierda respectivamente de la **casilla vacía**.

1.3. Que entregar

Notar que no se evaluarán programas que no cumplan con las especificaciones descritas anteriormente.

Las cosas que debe entregar son las siguientes:

1. El código fuente de los dos programas. Tu código debe ser escrito en C/C++, Java, Prolog o cualquier otro lenguaje de programación y debe ser muy bien comentado. Trate de usar en la medida de lo posible un compilador compatible con un sistema Linux.

2. Un archivo LEAME que contenga las instrucciones necesarias para generar un ejecutable (o para ejecutar su programa en algún intérprete) y los integrantes del equipo.

3. Subir el archivo `matricula.zip` (por ejemplo, `017842.zip`) que contenga una carpeta llamada `matricula` (`017842`) a la plataforma Blackboard en el enlace de este proyecto 1. La carpeta debe contener el código fuente y el archivo LEAME.

2. Observaciones

La fecha límite para entregar este proyecto es el día 19/02/2014. ¡Por ningún motivo se aceptarán trabajos fuera de tiempo!

Referencias

[1] Stuart Russell & Peter Norving. *Inteligencia Artificial: Un Enfoque Moderno.* Prentice-Hall, Inc. 2004.