Inteligencia Artificial
Taller de Búsqueda. (No informada e Informada)

1. [Búsqueda no Informada]. El siguiente rompecabezas es un juego para niños en el cual las piezas solo pueden unirse en línea recta cambiándolas de lugar a través de tres operaciones permitidas por el juego. El objetivo es lograr una configuración como la que se presenta a continuación:

   El estado inicial puede ser cualquier combinación posible de las cuatro piezas. Las operaciones permitidas son:

   - **D:** intercambiar dos piezas a la derecha
   - **C:** intercambiar dos piezas en el centro
   - **I:** intercambiar dos piezas en la izquierda

   Tomando un estado inicial al azar y considerando las operaciones permitidas:

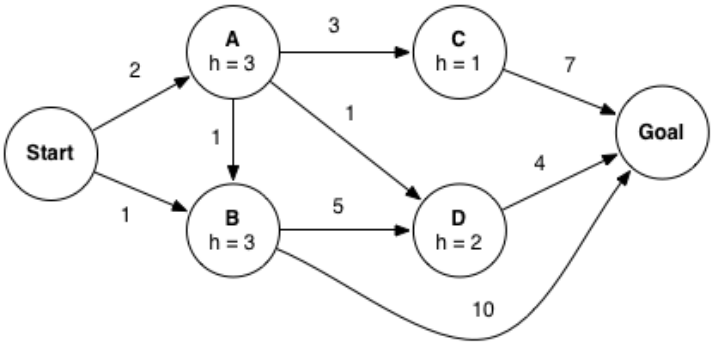
   A. Defina el juego como un problema de búsqueda
   B. Construya el grafo del espacio de estados
   C. ¿Cuál es el tamaño del espacio de estados para el juego?
   D. Construya el árbol de búsqueda y encuentre una solución al problema. ¿existe más de una solución?
   E. Construya un programa en Python que permita encontrar una solución para cualquier estado inicial. Recuerde que una solución será definida como la secuencia de acciones que nos llevará del estado inicial al estado objetivo.
   F. Genere una representación completa del árbol de búsqueda de forma automática (considere usar NetworX)

Sugerencias:

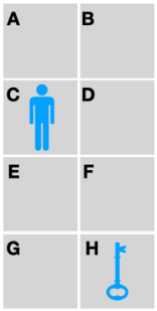
- Es posible representar cada estado del juego como una lista de 4 números.
- Antes de intentar programar la solución realice una especificación detallada del problema y encuentre la solución por simple inspección
- Debido a que el problema cuenta con un número pequeño de estados, puede encontrar la solución haciendo una búsqueda exhaustiva desde un estado inicial.

Recuerde que no es necesario generar la representación completa del árbol desde el inicio; podemos utilizar un estado inicial e ir generando los hijos para realizar la exploración en cada iteración.

2. Para el siguiente grafo de búsqueda encuentre la solución utilizando los algoritmos de UCS, voraz y A*. Construya un cuadro comparativo donde presente: la ruta entregada por cada algoritmo, la cantidad de nodos expandidos, la cantidad de nodos en la ruta solución y el costo de la ruta solución.



3. Considere la siguiente habitación en la cual un jugador se encuentra ubicado en la posición **C** y desea encontrar una llave para poder escapar. El jugador puede moverse a cualquiera de los cuadros adyacentes (siempre que permanezca dentro del laberinto). Asuma que los costos de moverse en diagonal son de 3 puntos mientras el costo de moverse arriba, abajo, izquierda o derecha es de 1 punto. Encuentre la solución al problema utilizando los algoritmos de UCS, Voraz y A*(con la distancia de Manhattan como heurística). Si tiene empates a la hora de seleccionar un nodo de la frontera, siga el orden alfabético. Genere un cuadro comparativo al igual que en el punto anterior



4. Considerando el mismo problema de la habitación propuesto en el punto tres, proponga heurísticas que cumplan las condiciones requeridas. Puede utilizar las heurísticas utilizadas usualmente en clase o proponer los valores directamente a los estados.

Para tener en cuenta

- La actividad puede realizarse en grupos.
- El objetivo del laboratorio es lograr un buen entendimiento de algoritmos de búsqueda informada
- El taller debe ser entregado a través de Google Colab

Recursos

- Estructuras de datos en Python (https://docs.python.org/3/tutorial/datastructures.html)
- Proyecto Pacman (https://inst.eecs.berkeley.edu/~cs188/sp21/project1/)