Introducción a la Inteligencia Artificial
Entregable Nº 1: Problema de la Torre de Hanoi

21 de septiembre de 2024

Autores	E-mail	Nº SIU
Leonardo Centurión	centurionm.leo@gmail.com	a1803
Braian Desía	b.desia@hotmail.com	a1804
Juan José Cardinali	juanchijc@gmail.com	a1089
Rodrigo Meza	rodrigolx31@gmail.com	a1815
Juan Ramos Nervi	jern10@gmail.com	a1821

Enunciado 1

¿Cuáles son los PEAS de este problema? (Performance, Environment, Actuators, Sensors)

Resolución

- **Performance**:
  - Si alcanza o no una solución. En este caso, si logra mover todos los discos de forma tal que se alcance una configuración en la que esten ordenados todos en forma descendente en una misma varilla distinta a la inicial.
  - Cantidad de movimientos hasta alcanzar una solución.
- **Environment**: Las m varillas y n discos. En este caso, 3 varillas y 5 discos.
- **Actuators**: Lo que sea que mueva un disco de una posición a otra.
- **Sensors**: Lo que sea que detecte la cantidad de discos y varillas (parámetros de entrada) y la configuración inicial de los discos (estado inicial).

Enunciado 2

¿Cuáles son las propiedades del entorno de trabajo?

Resolución

- **Totalmente observable**: El agente tiene noción plena del entorno de trabajo, sabe perfectamente la cantidad de varillas y discos, y la distribución de estos últimos.
- **Determinista**: El siguiente estado del entorno de trabajo esta plenamente determinado con la configuración actual y la acción que lleva a cabo el agente.
- **Secuencial**: Cada movimiento de un disco, condiciona los siguientes movimientos, es decir, las posibles configuraciones admisibles.
- **Estático**: El entorno de trabajo no cambia por si mismo o por otros agentes, mientras el agente razona. El entorno cambia pura y exclusicamente por acción del agente.
- **Discreto**: Las variables que definen el entorno del trabajo (cantidad de varillas y discos, cantidad de discos en cada varilla, orden de los discos) están representadas por valores discretos. En este caso, estas variables pertenecen al conjunto de los enteros $\mathbb{N}$.
- **Agente individual**.

Enunciado 3

En el contexto de este problema, establezca cuáles son los: estado, espacio de estados, árbol de búsqueda, nodo de búsqueda, objetivo, acción y frontera.

Resolución

- **Estado**: Para el caso de 3 varillas y 5 discos, se representa mediante una 3-upla de listas de enteros del 1 a 5 $[l_1, l_2, l_3]$ donde cada l_1 es el estado de la varilla i, siendo el primer elemento de $l_i[1]$ el disco del fondo de la varilla y el siguiente el disco que se encuentra por encima, y así sucesivamente.
- **Espacio de estados**: Dada la definición de estado de más arriba, el espacio de estados se conforma por todas las combinaciones posibles de los elementos del conjunto [1,5] distribuidos en la 3-upla $[l_1, l_2, l_3]$, con cada l_1 ordenado de forma descendiente.
- **Árbol de búsqueda**: Todas las posibles secuencias de movimientos que el agente puede realizar para pasar de un estado inicial al estado objetivo final (todos los discos en otra varilla distinta a la inicial, en orden descendente de tamaño).
- **Nodo de búsqueda**: Comprende:
  - State: El estado, del espacio de estados, que corresponde el nodo representado en este problema como la 3-upla.
  - Node Parent: El nodo en el árbol de búsqueda que ha generado al nodo, es decirm el estado del espacio de estados del cual deriva la configuración actual.
  - Action: La acción que se aplicará al padre (estado anterior) para generar el nodo (estado actual).
  - Path-Cost: El costo desde el estado inicial al estado actual.
- **Objetivo**: Alcanzar una configuración donde todos los discos se encuentren en una única varilla, distinta a la inicial, ordenados en forma descendente.
- **Acción**: Tomar el último disco de una de las varillas, partiendo de arriba hacia abajo, y colocarlo en otra siempre y cuando se cumpla que el disco a colocar sea de menor tamaño que el último disco en la varilla nueva. En el código, tomar el último elemento de una de las listas de la 3-upla y colocarlo al final de una de las listas restantes verificando que la lista recibidora del disco permanece ordenada en forma descendente.
- **Frontera**: Los posibles estados alcanzables a partir del estado actual en el árbol de búsqueda que aún no se han explorado.

Enunciado 4

Implemente algún método de búsqueda. Puedes elegir cualquiera menos búsqueda en anchura primero (el desarrollado en clase). Sos libre de elegir cualquiera de los vistos en clases, o inclusive buscar nuevos.

Resolución

Se adjunta código. También disponible en el repositorio de github. Se implemento la funcion dfs_tree_search en el archivo search.py

Enunciado 5

A nivel implementación, ¿qué tiempo y memoria ocupa el algoritmo? (Se recomienda correr 10 veces y calcular promedio y desvío estándar de las métricas).

Enunciado 6

Si la solución óptima es $2^k - 1$ movimientos con k igual al número de discos. Qué tan lejos está la solución del algoritmo implementado de esta solución óptima (se recomienda correr al menos 10 veces y usar el promedio de trayecto usado).

Resolución

Se obtienen las siguientes métricas al realizar 10 mediciones de la implementación del algoritmo de DFS:

Cuadro 1: Tabla analizando el algoritmo de busqueda en profundidad (DFS).

Metric	Value
Avg cpu time	0.12 s
Avg mem used	0.149 Mb
Avg path length	81.0 nodes
Optimal path length	31 nodes

Comparando con las metricas del algoritmo provisto de BFS se puede apreciar que, en efecto, consume mas tiempo y memoria pero obtiene la solución óptima.

Cuadro 2: Tabla analizando el algoritmo de busqueda en anchura (BFS).

Metric	Value
Avg cpu time	0.360 s
Avg mem used	0.24 Mb
Avg path length	31.0 nodes
Optimal path length	31 nodes