Artificial Intelligence
HW 1: Search and games

Published: 2021/02/09
Due date: 2021/03/08 23:55
In class solutions: 2021/03/09

---

1. Un supermercado quiere instalar un sistema para reconocer las matrículas de las furgonetas de los proveedores que acceden a su parking. El sistema tiene fallos por el efecto de sombras en la imagen, matrículas que están sucias, etc. Hay caracteres que tiende a confundir más, como el 1 con el 7. Por ejemplo, la matrícula **1**234ABC puede confundirla con **7**234ABC. A veces también un carácter puede no ser leído. Por ejemplo la matrícula **1**234ABC puede confundirla con 234ABC. El supermercado tiene la lista de matrículas de los proveedores, por lo que en el futuro quiere implementar un sistema que tome la matrícula leída y devuelva la matrícula del listado más cercana.

Para ello, vamos a enfocarnos en el primer problema, que es definir la distancia entre dos matrículas:

- La distancia entre una matrícula M y otra Mg se define como el **coste mínimo** de convertir M en Mg
- Para convertir M en Mg se realiza una secuencia de acciones individuales. Cada una de ellas puede ser:

Convertir un carácter en otro. El coste de la acción depende del carácter convertido:

- Convertir un "1" en un "7": **coste 1**
- Convertir un "7" en un "1": **coste 1.5**

- **Añadir "1" a la izquierda de M: coste 1**. Se puede realizar sólo si la longitud de M es un carácter
- **Añadir "7" a la izquierda de M: coste 3**. Se puede realizar sólo si la longitud de M es un carácter
- **Eliminar último carácter de M** (sólo en caso de que la longitud de M es dos caracteres): **coste 1**

Con estas definiciones, la distancia entre "17" y "77" es 1, ya que la forma menos costosa de convertir "17" en "77" es simplemente reemplazar el 1 por 7.

En nuestro primer prototipo consideramos sólo consideraremos los caracteres 1 y 7. Si M es el estado actual de la matrícula, y Mg una matrícula objetivo:

1. Propón una heurística monótona para este problema (convertir M en Mg con el menor coste total posible).

2. Considerando que la matrícula leída es "7", y la matrícula objetivo es "71", desarrolla el árbol de búsqueda empleando el algoritmo A* **con** eliminación de estados repetidos. Para cada nodo, indicar el orden de expansión y los valores de coste g+h = f. El orden de las acciones aplicables a cada estado es el mismo que el orden en el que están definidas más arriba

3. ¿Cuál es la distancia entre "7" y "71"?

4. ¿Es A* **con** eliminación de estados repetidos con la heurística propuesta óptima?

5. ¿Sería A* **sin** eliminación de estados repetidos con la heurística propuesta óptima?

2. Se tiene un puzle formado por 2 fichas negras, 2 fichas blancas y 1 hueco, colocados inicialmente de la siguiente forma:

N	N		B	B

En este juego, una ficha puede moverse a una posición adyacente vacía **con coste 1**. Ejemplo:

N	N	B		B

Blanca mueve a adyacente

Además, una ficha puede saltar sobre otra (ojo, no más de 1) hasta alcanzar un hueco. Ejemplo:

	N	N	B		B

Negra salta 1 ficha

En este caso el coste es **2**. El objetivo consiste en conseguir que todas las fichas blancas estén a la izquierda de todas las negras. La posición del hueco no tiene importancia.

a) Formaliza los estados, acciones/operadores y costes asociados en este problema. ¿Cuál es el estado S0? ¿y Sf?

b) Analiza el espacio de estados correspondiente a este problema: i) di cuál es el factor de ramificación mínimo y máximo de los nodos; ii) comenta sobre la existencia de ciclos y las distintas formas de evitarlos.

c) Propón una heurística admisible para este problema, y justifica por qué es admisible

d) ¿Es A* con eliminación de estados repetidos con la heurística propuesta óptima?

e) ¿Sería A* sin eliminación de estados repetidos con la heurística propuesta óptima?

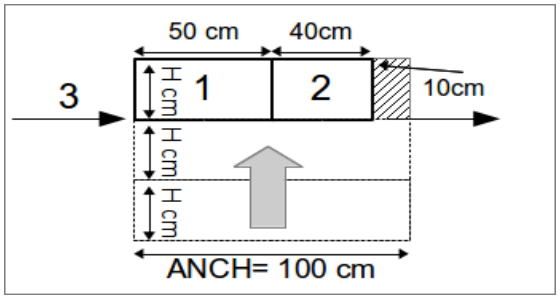
f) Desarrolla el árbol de búsqueda empleando el algoritmo A* **CON** eliminación de estados repetidos. Para cada nodo, indicar el orden de generación y el de expansión y los valores de coste $g+h=f$. En caso de empates se expanden los nodos que se hayan generado primero. Elije tú el orden de expansión e indícalo y sé consistente con ese orden en toda la generación del árbol. **Se pide expandir el árbol hasta encontrar una solución o cuando se hayan expandido (que no generado) 7 nodos.**

3. Se quiere resolver mediante búsqueda un problema de corte unidimensional (*one-dimensional cutting problem*): Imaginemos que tenemos un rollo o lámina de un material plano y de anchura ANCH. El rollo se va desenrollando a medida que hace falta. Queremos obtener un conjunto de "tiras" de material de altura fija H y longitud Li ≤ ANCH. En nuestro caso particular tendremos ANCH=100 cm. Por otra parte Li y el número de tiras (Ni) que queremos de cada longitud vienen especificadas en una tabla como la de la figura 1.

Se pueden hacer cortes verticales y cortes horizontales. La forma de proceder es la siguiente: se hacen cortes verticales de altura H a diferentes distancias, y cuando no "cabe" una nueva tira en el sobrante se corta en horizontal el rollo, desperdiciando por tanto una cantidad de material.

Por ejemplo (ver figura 1 izquierda): si el ancho es 100 cm, y deseamos (figura 1 derecha) obtener 3 tiras de 25 cm, 2 tiras de 40 cm y 3 tiras de 50 cm, podríamos empezar haciendo un corte a 50 cm del borde izquierdo (paso 1), luego otro a 40 cm del anterior (paso 2), y luego cortar en horizontal (paso 3), produciendo dos de las tiras pedidas y desperdiciando 10 cm de material tras el corte horizontal. Tras estos tres pasos iniciales el sistema debería seguir cortando hasta obtener el total de tiras solicitadas.

El objetivo del problema es obtener las tiras especificadas en la tabla (3 de 25 cm, 2 de 40 cm y 3 de 50 cm) desperdiciando la menor cantidad total posible de material. Cuando el sistema ha cortado todas las tiras especificadas en la tabla, debe terminar cortando a lo ancho para dejar el rollo como al principio.



Id	Li	Ni
1	25cm	3
2	40cm	2
3	50cm	3

Figura 1. Izquierda: Instancia del problema y ejemplo de corte de 50 cm y 40 cm. Derecha: ejemplo de tabla en el que se especifican las longitudes y número de cortes deseados.

a) **Define** cómo representarías los estados de un problema genérico de este tipo. ¿Cuál sería el estado inicial para el problema especificado en la tabla? ¿y el estado final?

b) **Especifica** con claridad los operadores (acciones) del problema, incluyendo sus posibles restricciones. ¿Cuál es el factor de ramificación? ¿Se puede determinar la profundidad máxima de la solución?

c) Especifica cuál es el **coste** de cada acción posible.

d) **Define** una heurística admisible para este problema que ayude a encontrar la solución de mínimo coste, justificando por qué lo sería.

e) **Genera** el árbol de búsqueda que resulta tras las dos primeras expansiones cuando la tabla de tiras deseadas es la especificada en la figura 1 derecha.

f) Con la heurística propuesta, ¿es óptima A* sin eliminación de estados repetidos? ¿y con eliminación de estados repetidos?

g) Desarrolla el árbol de búsqueda empleando el algoritmo A* **CON** eliminación de estados repetidos. Para cada nodo, indicar el orden de generación y el de expansión y los valores de coste $g+h = f$. En caso de empates se expanden los nodos que se hayan generado primero. Elije el orden de expansión e indícalo y sé consistente con ese orden en toda la generación del árbol. **Se pide expandir el árbol hasta encontrar una solución, o cuando se hayan expandido (que no generado) 7 nodos.**

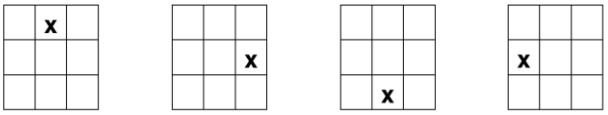
4. Se quiere desarrollar un programa que juegue a las 3 en raya. Las fichas de un jugador se representan como **X** y las del otro con **O**. Definimos X_n como el número de filas, columnas o diagonales con n fichas de **X** y ninguna de **O**. Similarmente O_n, representa el número de filas, columnas o diagonales con sólo n fichas de **O**. Por otra parte, a los estados terminales con X_3= 1 se les asigna el valor +1 y a los que tienen valor O_3=1 se les asigna -1. A todos los demás nodos terminales se les asigna el valor 0. Se va a utilizar el método minimax con una **función de evaluación** $E = 3 \cdot X_2 + X_1 - (3 \cdot O_2 + O_1)$

Empieza la partida **X**.

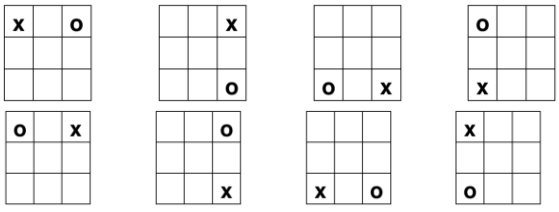
Se pide:

a) ¿Qué tipo de juego es?
   i. Determinista / aleatorio
   ii. Información completa / información no completa
   iii. Suma cero / suma constante / suma variable

b) Representar el árbol de juego hasta el nivel 2 (siendo el nivel de la raíz 0), empezando con el tablero vacío como nodo raíz y eliminando estados simétricos. Por ejemplo los siguientes estados los consideraremos simétricos (equivalentes), por lo que uno de ellos es equivalente al resto:

i. 

ii. Otro ejemplo de estados equivalentes son los ocho estados siguientes:

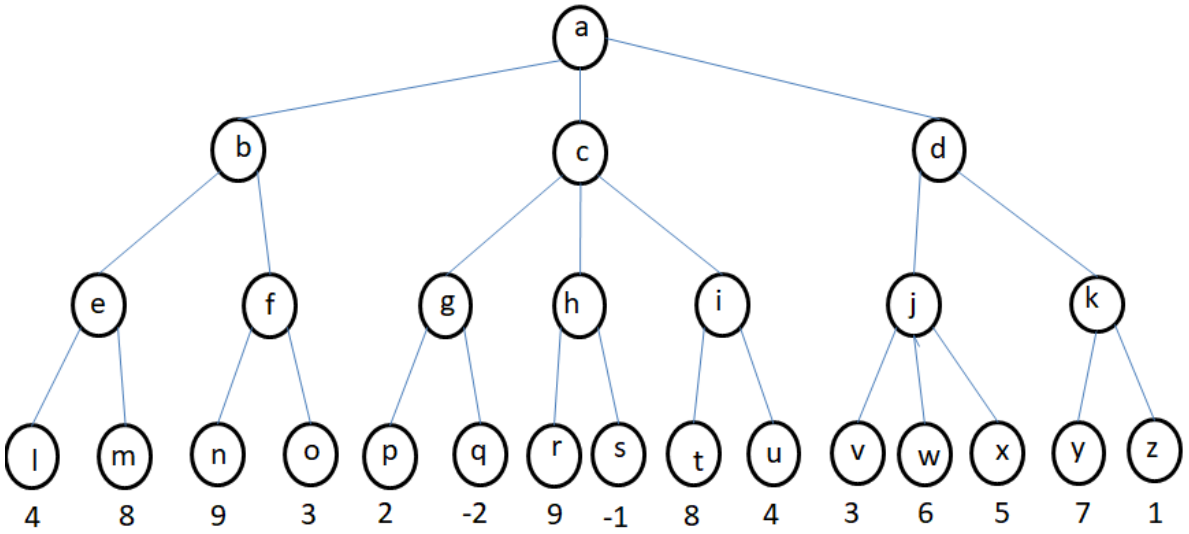
iii. 

c) Señalar en cada nivel si corresponde a MIN o a MAX.

   a. Sobre el árbol anterior, indicar los valores obtenidos para los nodos de los niveles 0 y 1 aplicando el algoritmo mini-max sin poda alfa beta, propagando hacia arriba los valores estimados mediante la función de evaluación en el nivel 2. Utilizar dichos valores para seleccionar el mejor movimiento inicial que debería realizar **X**.

   b. Marcar los nodos del nivel 2 que no serían evaluados si se aplicara poda alfa-beta.

5. Considérese el árbol de juego de la figura, correspondiente a un juego de suma 0. Los valores con los que las hojas están etiquetadas corresponden a la función de utilidad para el jugador que tiene actualmente el turno de juego.



a) Establecer los valores que serán propagados por el algoritmo minimax sin poda al resto de nodos del árbol.

b) Señalar en cada nivel si corresponde a MIN o a MAX

c) ¿Qué movimiento será el seleccionado por el jugador que tiene el turno de juego, si es un jugador inteligente que aplica teoría de juegos a sus decisiones, en caso de que utilice minimax sin poda?

d) Consideremos ahora que el jugador realizara el análisis de su jugada óptima pero usando minimax con poda alfa-beta. Dibuja el árbol correspondiente indicando claramente los pasos del algoritmo, enumerando y etiquetando cada uno de ellos con la información relevante, e indicando claramente los momentos de poda. Se asume que en un mismo nivel los nodos se van generando de izquierda a derecha según aparecen en la figura. ¿Cúal sería el valor minimax asignado a la raíz? ¿qué movimiento debería realizar el jugador con el turno de juego según este análisis?

6. Considérese el siguiente juego de dos jugadores: "*Se dispone de un montón de palillos situado en el centro de una mesa. Inicialmente tiene 6 palillos. Hay dos jugadores: A y B, que irán retirando X palillos de ese montón común alternándose hasta que no queden palillos. Los jugadores A y B alternan sus jugadas, comenzando por el jugador A. En cada jugada un jugador puede retirar del montón 1, 2 o 3 palillos siempre que haya un número suficiente de palillos en éste. Siempre que pueda retirará los palillos en ese orden. El jugador que retira el último palillo del montón es el que pierde*"

IMPORTANTE: Proporciona respuesta para cada uno de estos apartados

a) ¿Qué tipo de juego es?
   a. Determinista / aleatorio
   b. Información completa / información no completa
   c. Suma cero / suma constante / suma variable

b) Para este tipo de juego, suponiendo que tanto A como B realizan elecciones óptimas ¿cuál es el algoritmo que determina cuál es la jugada óptima?

c) Sobre el árbol de juego desplegado aplica el algoritmo minimax. De acuerdo con este algoritmo, ¿cuál sería el valor minimax en la raíz del árbol? ¿cuál sería la jugada óptima?

d) Aplica el algoritmo minimax con poda alfa-beta. En este apartado se deben indicar claramente los pasos del algoritmo, enumerando y etiquetando cada uno de ellos con la información relevante e indicando claramente los momentos de poda. ¿Cuál sería el valor minimax en la raíz del árbol? ¿cuál sería la jugada óptima? ¿Qué jugador gana la partida?