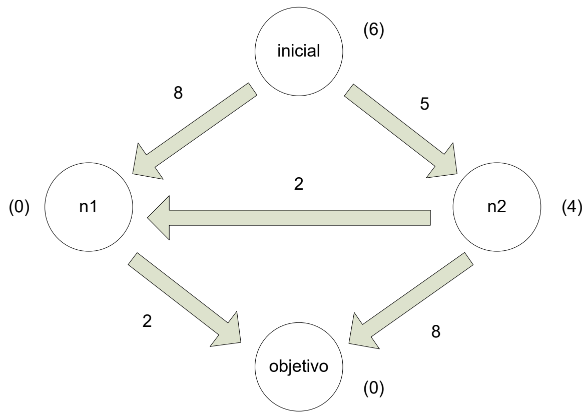
Ejercicio 1. Se trata de un ejercicio de propiedades formales de A* en un problema abstracto

Consideremos el espacio de búsqueda representado en el siguiente grafo. Los números entre paréntesis representan los valores de un heurístico h1.



Se pide responder de forma razonada, y breve, a las siguientes cuestiones. En las preguntas 1, 2, 3, 4, 6 y 7, hay que contestar utilizando las propiedades formales de A*, sin necesidad de hacer una traza de la ejecución del algoritmo. Sin embargo, en las preguntas 5, 8 y 9 hay que hacer una traza de la ejecución de A*. En estos casos hay que indicar claramente: i) la secuencia de nodos que se expanden, ii) los valores de f(n) para estos nodos, iii) el contenido de ABIERTA antes de cada iteración y iv) el camino registrado en la TABLA-A desde el inicial a cada nodo antes de cada iteración.

1.- ¿Está el heurístico h1 bien definido?

2.- ¿Cuál es el valor de C* y el de h*(n), para todos los nodos del espacio de búsqueda?

3.- Si se aplica A* con el heurístico h1, ¿Encontrará la mejor solución?

4.- ¿Es monótono el heurístico h1?

5.- Supongamos ahora que se aplica un algoritmo A* al espacio anterior utilizando el heurístico h1. Además de los puntos i,ii,iii y iv anteriores, hay que indicar claramente las rectificaciones que se producen (si las hay).

6.- En el caso de que el heurístico h1 anterior no fuese monótono, modificar el valor del heurístico para el nodo n1, de forma que el heurístico resultante, h2, sí sea monótono.

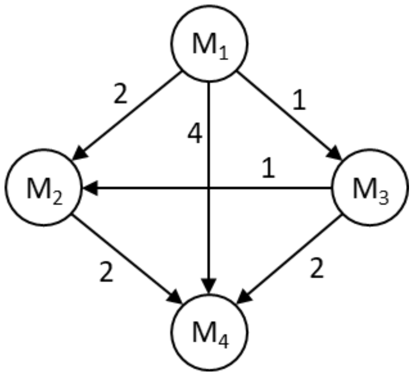
7.- ¿Qué solución (secuencia de nodos) encontrará A* si se aplica este nuevo heurístico h2 y cuáles serán los valores de g(n) de estos nodos en el momento de ser expandidos?

8.- Hacer la traza de A* con el heurístico h2.

9.- Definir ahora otro heurístico, h3, que haga que A* encuentre la solución (inicial→n1→objetivo), si se aplica a este espacio de búsqueda. Hacer la traza correspondiente.

Ejercicio 2. Otro de búsqueda en espacios abstractos

Consideremos el siguiente espacio de búsqueda abstracto, el algoritmo A* y los heurísticos cuyos valores se indican por extensión en la tabla



		h_1	h_2	h_3	h^*
(inic.)	M_1	2	2	2	
	M_2	0	1	2	
	M_3	2	2	2	
(obj.)	M_4	0	0	0	

Como siempre, se pide responder de forma razonada a las siguientes cuestiones

a) ¿Cuál es el valor de C*, y el de h* para cada uno de los nodos?

b) ¿Qué propiedades tienen los heurísticos h1, h2 y h3? ¿Se puede establecer algún tipo de comparación entre ellos?

c) De acuerdo con las propiedades anteriores, ¿Qué se puede decir con respecto a la expansión del nodo M_2 con cada uno de los tres heurísticos?

Ejercicio 3. Sobre el 8-puzzle, A* y heurísticos

Consideremos un heurístico h(n) tal que para cada estado suma 1 por cada ficha que no está en la columna que le corresponde en el objetivo y suma 1 también por cada ficha que no está en su fila en el objetivo.

a) Indicar de forma razonada si h está bien definido y cuáles son sus propiedades.

b) Comparar el heurístico h con los clásicos h1 y h2 en términos de calidad de la solución y nodos expandidos.

c) Indicar los valores de h, h1 y h2 para el siguiente estado y sus sucesores

$$\begin{bmatrix} 7 & 3 & 1 \\ 6 & & 4 \\ 2 & 8 & 5 \end{bmatrix}$$

Ejercicio 4. El problema de asignación simple con búsqueda heurística

Dada una matriz cuadrada de N×N números, se trata de calcular un subconjunto C de N elementos de la matriz cuya suma sea mínima, tales que no haya dos números de la misma fila ni de la misma columna en C

Ejemplo con N=4, la solución óptima es C={1,2,3,4}:

$$\begin{bmatrix} 7 & 5 & \mathbf{1} & 4 \\ 3 & \mathbf{2} & 5 & 7 \\ 8 & 4 & 6 & \mathbf{3} \\ \mathbf{4} & 9 & 8 & 3 \end{bmatrix}$$

Se pide lo de siempre:

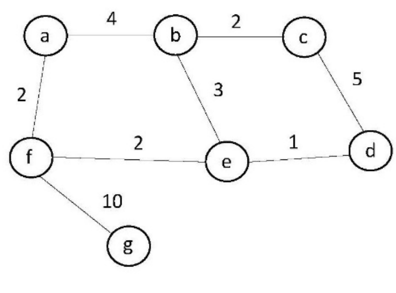
a) Describir el espacio de búsqueda, de forma genérica para cualquier tamaño N, y dibujar el espacio completo para la matriz anterior. Indicar claramente qué representa un estado. PISTA: un estado se puede representar de forma natural por un subconjunto X de {1..N}.

b) Definir un heurístico mediante el método de la relajación del problema e indicar los valores que toma el heurístico para cada uno de los nodos de un camino cualquiera del espacio de a) desde el inicial a un objetivo.

c) Para el nodo más intermedio del camino anterior, indicar lo que se puede decir con respecto a su expansión si se utilizase el algoritmo A* con el heurístico anterior.

Ejercicio 5. El problema del vigilante con búsqueda en espacios de estados

Tenemos un grafo que representa conexiones entre ciudades, como por ejemplo el siguiente:



Una conexión entre dos ciudades expresa que se puede ir de una a la otra con el coste que indica el arco, pero también que se puede ver una ciudad desde la otra (esto sin coste alguno). Si dos ciudades no están conectadas con un arco, no se puede ir de una a la otra directamente, ni se puede ver una desde la otra. Una ciudad se ve desde sí misma, por supuesto.

El objetivo es calcular una ruta que parta de una ciudad concreta, en el ejemplo anterior es la ciudad a, que pase por una serie de ciudades, pero solamente una vez por cada una de ellas (y no tiene que pasar por todas las ciudades del grafo) y que termine de nuevo en la ciudad de partida, de forma que todas las ciudades del grafo puedan ser vistas desde alguna de las ciudades de la ruta. Además, la ruta debe tener coste mínimo.

En el ejemplo anterior hay varias soluciones, la óptima es la ruta (a,b,e,f,a) con coste 11. En este problema es posible que no exista solución alguna, dependiendo de cómo sean las conexiones del grafo.

En este ejercicio se pide lo siguiente:

1.- Indicar otra solución al problema del grafo anterior que sea distinta a (a,b,e,f,a) y su coste. Indicar también un ciclo que empiece y termine en la ciudad a y que no sea solución del problema. Dar también un grafo de conexiones distinto al anterior para el que el problema no tenga solución.

2.- Modelar el problema para resolverlo con búsqueda en espacios de estados. Es decir, definir el espacio de búsqueda (estados, reglas, costes, estado inicial, estados objetivos). Dibujar una parte significativa del espacio de búsqueda para la instancia del problema definida por el grafo anterior.

3.- Proponer un heurístico para resolver el problema con el algoritmo A*, e indicar sus propiedades. Dar los valores de este heurístico para un conjunto significativo de los nodos del espacio anterior.

PISTAS: (a) No hay que confundir las ciudades visitadas con las ciudades vistas. Cuando llegamos a una ciudad desde otra la registramos como visitada. Cuando estamos en una ciudad vemos las adyacentes. (b) El algoritmo de Dijskstra puede ser de utilidad.

Ejercicio 6. Planificación de tareas (scheduling) con búsqueda en espacios de estados.

Esta vez consideramos el problema de planificación de un conjunto de *n* trabajos $\{J_1,\ldots,J_n\}$ en *m* máquinas $\{M_1,\ldots,M_m\}$. Cada trabajo J_i tiene una duración $d_i>0$, $i=1,\ldots,n$; y la ejecución del trabajo J_i en la máquina M_j tiene un coste $C_{ij}>0$, $i=1,\ldots,n$; $j=1,\ldots,m$. Cada máquina puede ejecutar cualquier trabajo, pero solo uno a la vez.

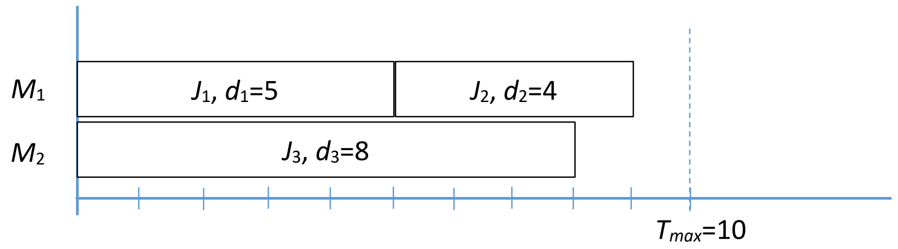
El objetivo es planificar los *n* trabajos en la *m* máquinas de forma que su ejecución no exceda un tiempo $T_{max}>0$ y que el coste total de ejecución sea mínimo.

En primer lugar, hay que resolver el problema con búsqueda en espacios de estados, para ello se pide:

1.- Dar una interpretación precisa del espacio de búsqueda (estados, estado inicial, estados objetivo, reglas y costes de las reglas)

2.- Dibujar el espacio de búsqueda completo para la instancia del problema definida por los siguientes datos: $n=3$, $m=2$, $d_1=5$, $d_2=4$, $d_3=8$, $C_{11}=5$, $C_{12}=15$, $C_{21}=4$, $C_{22}=16$, $C_{31}=7$, $C_{32}=6$ y $T_{max}=10$ (con estos datos si $T_{max}<9$ el problema no tendría solución).

El gráfico de Gantt siguiente muestra una solución para esta instancia. El coste de esta solución es $C_{11} + C_{21} + C_{32} = 5 + 4 + 6 = 15$



Además, consideramos que este espacio de búsqueda se recorre con el algoritmo A*, para ello se pide también:

3.- Definir un heurístico lo mejor informado posible, preferiblemente a través del método de relajación del problema. Indicar qué propiedades tiene el heurístico y dar sus valores para los nodos del espacio de búsqueda anterior.

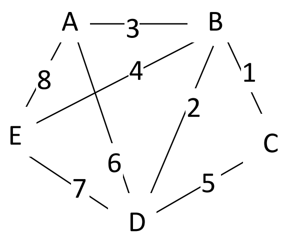
Ejercicio 7. El problema MAX_CUT con búsqueda heurística

El famoso problema MAX_CUT se puede enunciar del siguiente modo: dado un grafo no dirigido G =(V,E) con costes positivos en los arcos, se trata de calcular una partición del conjunto de nodos V en dos subconjuntos V_1 y V_2 (es decir $V = V_1 \cup V_2$ y $V_1 \cap V_2 = \emptyset$). Si denotamos por E_1 al subconjunto de arcos de E que conectan pares de nodos de V_1 (análogamente E_2), y por Coste(E_1) la suma de los costes de los arcos de E_1 (análogamente Coste(E_2)), el objetivo es encontrar la partición (V_1,V_2) que minimice el valor de Coste(E_1)+ Coste(E_2).

En este ejercicio se pide resolver el problema MAX_CUT utilizando búsqueda en espacios de estados, en particular el algoritmo A*. Mas concretamente:

1.- Describir de forma precisa los estados y las reglas, es decir el espacio de búsqueda, de forma genérica, indicando si es un árbol o un grafo (tened en cuenta que puede haber reglas de coste 0 siempre y cuando no se generen ciclos en el espacio de búsqueda de coste 0, o caminos de longitud infinita y coste 0).

2.- Dibujar una parte representativa del espacio de búsqueda para el problema dado por el siguiente grafo

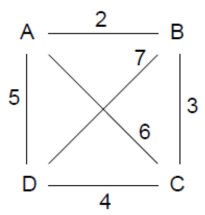


3.- Diseñar un heurístico razonable, a ser posible utilizando el método de la relajación del problema. Indicar las propiedades del heurístico, y dar sus valores para la fracción del espacio de búsqueda dibujado anteriormente. PISTA: una forma de relajar el problema es prescindir de algunos de los arcos.

Ejercicio 8. Dos versiones del TSP con el algoritmo A*

Se trata de utilizar el algoritmo A* para resolver dos variantes del problema del viajante de comercio (TSP). La primera es la variante clásica en la que un viajante debe partir de una ciudad A, pasar por el resto de las ciudades una y solo una vez y volver a la ciudad de partida con un coste mínimo. En este caso se pide:

a) Describir de forma genérica el espacio de búsqueda para un problema con N ciudades, conectadas directamente todas con todas, y dibujar el espacio completo para la siguiente instancia del problema:



b) Describir un heurístico, lo mejor posible, para el problema. Justificar sus propiedades y dar los valores que toma este heurístico para todos los nodos del espacio de búsqueda del ejemplo anterior.

c) Consideremos ahora la versión del TSP con "ventanas de tiempo". Es como el anterior, con una restricción añadida: el viajante tiene que pasar por cada ciudad X en un instante del intervalo $[t_{Xi}, t_{Xf}]$. En este caso se pide lo mismo que para la versión anterior. Como ejemplo concreto consideraremos el problema anterior en el que las ventanas de tiempo para las ciudades B, C, y D son respectivamente [8,19], [5,7] y [9,17].

Ejercicio 9. El problema RAP (Reviewer Assignment Problem) con búsqueda en espacios de estados

Se trata de resolver el problema de asignación de revisores a propuestas de proyectos de investigación (Reviewer Assignment Problem o RAP). En el RAP tenemos un conjunto de **R** revisores y otro conjunto de **P** propuestas de proyectos, cada proyecto debe ser revisado por un revisor, y un revisor no puede revisar más de **N** proyectos. Además, hay una matriz $\mathbf{C}_{R\times P}$ de conflictos; $\mathbf{C}_{ij}$ es el conflicto entre el revisor ***i*** y la propuesta ***j***. El objetivo es minimizar el conflicto total, calculado como la suma de los conflictos entre cada proyecto y el revisor asignado.

En el siguiente ejemplo, con **R**=3 revisores, **P**=4 proyectos, **N** = 2 y la matriz de conflictos:

$$\begin{vmatrix} 2 & 3 & \mathbf{1} & \mathbf{2} \\ 2 & 2 & 3 & 4 \\ \mathbf{1} & \mathbf{1} & 3 & 1 \end{vmatrix}$$

una solución óptima se obtiene asignando las dos primeras propuestas al revisor 3, y las dos siguientes al revisor 1 y ninguna al revisor 2. El coste de la solución es 5 (la suma de los conflictos en negrita). En este ejemplo, si el valor de **N** fuese 1 el problema no tendría solución.

En esta pregunta se pide lo siguiente:

a) Si la solución óptima anterior la representamos por (3, 3, 1, 1), dar la representación análoga para otras dos soluciones óptimas de esta instancia.

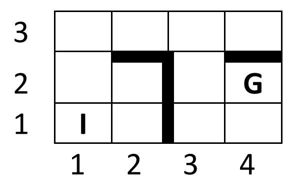
b) Resolver el problema con búsqueda en espacios de estados, concretamente:

b1) Describir un espacio de búsqueda adecuado. En primer lugar, dar una descripción general (estados, reglas y costes), y luego dibujar una parte significativa del espacio de búsqueda para la instancia anterior.

b2) Definir un heurístico para el algoritmo A* y el espacio anterior, mediante una relajación del problema. Dar los valores de este heurístico para el estado inicial y para alguno de los estados intermedios del espacio de búsqueda dibujado en la pregunta anterior.

Ejercicio 10. Cálculo de caminos en un grid bidimiensional con barreras

Se trata de resolver, con búsqueda en espacios de estados, el problema de cálculo del camino de menor coste entre dos posiciones de un grid bidimensional de tamaño N×M con barreras, como el que se muestra en el siguiente ejemplo para N=3 y M=4:



I representa la posición inicial y **G** la posición final del trayecto. Los movimientos permitidos son: up, down, left y rigth, siempre y cuando no nos salgamos del grid. Las fichas se pueden mover a través de las barreras, pero con un coste que es 4 veces el de los movimientos sin barrera.

En esta pregunta se pide responder de forma razonada a las siguientes cuestiones:

a) Describir un espacio de búsqueda adecuado. En primer lugar, dar una descripción general (estados, reglas y costes), y luego dibujar el espacio completo para la instancia anterior.

b) En el ejemplo anterior, indicar una solución óptima, y la peor solución que podría encontrar el algoritmo A* con un heurístico no admisible.

c) Definir un heurístico para el algoritmo A* y el espacio de búsqueda definido en a) mediante una relajación del problema. Dar los valores de este heurístico para todos los estados del espacio de búsqueda correspondiente al grid anterior.

d) Si consideramos el algoritmo PEA* con un heurístico admisible y un valor del parámetro ε=0,5 ¿cuál es la peor solución que se podría encontrar para la instancia anterior?