Soluciones: Búsqueda en espacios de estados

Ejercicio 1: Búsqueda no informada (0.5 puntos)

Dar una formulación completa de cada uno de los siguientes problemas. La formulación tiene que ser lo suficientemente precisa como para que permita una implementación directa.

a) Tienes que colorear un mapa plano usando solo 4 colores de tal forma que dos regiones adyacentes no tengan el mismo color.

b) Un chimpancé de 1 metro de altura está en una habitación donde unos plátanos están colgados del techo que está a una altura de 2 metros y medio del suelo. El chimpancé quiere coger los plátanos. La habitación dispone de dos cajones en forma de cubo de 1 metro de lado. Estos cajones son apilables y el chimpancé puede moverlos y escalarlos con facilidad.

c) Tienes 3 jarras de capacidades 12, 8 y 3 decilitros. Además tienes una fuente de agua y un sumidero. Puedes llenar las jarras o vaciarlas echando su contenido en otra jarra o bien en el sumidero o con la fuente. Necesitas medir exactamente 1 decilitro.

a) **Estado inicial**: Todas las regiones sin color.
Objetivo: todas las regiones coloreadas de tal forma que dos regiones adyacentes no tengan el mismo color.
Coste: Número de asignaciones.
Función Sucesor: Asignar un color a una región.

b) **Estado inicial**: La situación descrita en el enunciado.
Objetivo: Que el chimpancé coja los plátanos.
Coste: Número de acciones.
Función Sucesor: Subir al cajón; poner un cajón encima de otro; empujar el cajón desde un sitio hasta otro; caminar desde un sitio a otro; coger plátanos (si está sobre el cajón).

c) **Estado inicial**: Las jarras con 0 decilitros (0, 0, 0).
Objetivo: Tener una jarra con un decilitro.
Coste: Número de acciones.
Función Sucesor: rellenar la jarra de 12 decilitros, rellenar la jarra de 8 decilitros, rellenar la jarra de 3 decilitros, vaciar la jarra de 12 decilitros, vaciar la jarra de 8 decilitros, vaciar la jarra de 3 decilitros, pasar una cantidad ***y*** de decilitros de una jarra con ***z*** decilitros a otra jarra con ***x*** decilitros (esto aumentaría la cantidad de la segunda jarra que sería de ***x+y*** decilitros y disminuiría la cantidad de la primera jarra que sería de ***z-y*** decilitros).

Ejercicio 2: Búsqueda informada (a) 0.5 puntos + b) 0.5 puntos)

Usar el prototipo *http://aispace.org/search/*. Carga el grafo de los barrios de Vancouver *File* >>*Load Sample Problem* >> *Vancouver Neighbourhood Graph.* Este ejercicio consiste en hacer búsquedas de caminos desde el barrio UBC a los barrios: SP, SRY, RM, BBY, AP. Para definir qué nodos son finales o nodos regulares, en la pestaña "*create*", sobre cada nodo, con el botón derecho se podrá definir el carácter del nodo y el valor de la función heurística. Es interesante usar el menú "*view*" para poder inspeccionar los costes de los arcos y los valores de la función heurística utilizados en cada momento. Para seleccionar los diferentes algoritmos use las opciones del menú *Search Options* >> *Search Algorithm*. Se pide:

a) Construir una tabla con el número de nodos expandido y el coste del mejor camino hallado para cada destino con los siguientes algoritmos:
   1. *Breadth First*.
   2. *A** con los valores de la función heurística que sale por defecto en el prototipo (función heurística por defecto).
   3. A* con la función heurística h(n) = coste del camino más barato desde el nodo n a un nodo objetivo (función heurística óptima). Tenga en cuenta que si no existe ningún camino para llegar desde un nodo n al nodo objetivo el valor de la función heurística es infinito (un valor grande en la práctica).

b) Discute los resultados de la tabla generada en el apartado a).

a)

	A* con función heurística por defecto		A* con función heurística óptima		Breadth First	
Destino	#nodos expandidos	path cost	#nodos expandidos	path cost	#nodos expandidos	path cost
SP	7	11	5	11	13	11
SRY	21	30	5	30	19	30
RM	15	9	4	9	11	9
BBY	16	11	4	11	8	**13**
AP	17	12	5	12	18	12

b) El algoritmo A* encuentra la solución de menor coste (es decir la solución ÓPTIMA) con las dos funciones heurísticas. El algoritmo *Breadth First* no usa función heurística y no es óptimo puesto que no encuentra siempre la solución de menor coste. Por ejemplo, cuando el destino es BBY, encuentra un camino de longitud 13, aunque hay otro de longitud 11.

Las funciones heurísticas influyen de manera muy significativa en el número de nodos expandidos para alcanzar la solución. El número de nodos expandidos con la heurística óptima es menor que el número de nodos con la heurística por defecto porque la heurística óptima domina a la heurística por defecto. Además, el número de nodos expandido con el algoritmo *Breadth First* siempre es superior al del A* con la función heurística óptima, aunque es casi siempre mejor que el A* con la función heurística por defecto debido a que esta heurística está diseñada sólo para el destino SP.