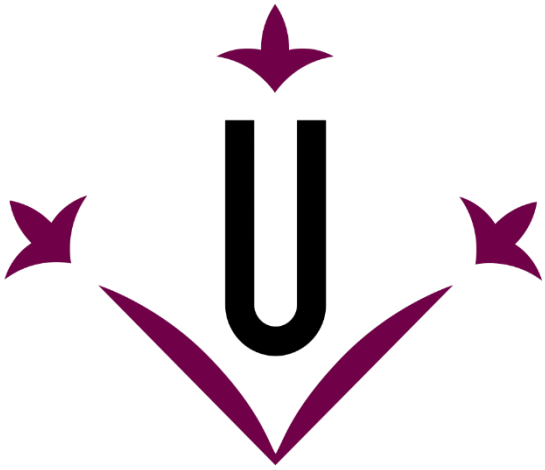


Universitat de Lleida

Primera practica

Algoritmos de búsqueda informada

Problema bien definido

Alberto Susin Nasarre
Marcos Susin Nasarre

Algoritmos de búsqueda informada

Se pide implementar los algoritmos de búsqueda informada estudiados en clase. Se ha conseguido implementar de forma correcta lo siguiente algoritmos:

- UCS
- A*
- Best-H

Se implementa como opcional la búsqueda bidireccional, funcionando de forma correcta para la gran mayoría de los mapas propuestos salvo en la búsqueda a mapa abierto.

A la hora de implementar los 3 algoritmos, se decide juntarlos en una misma función de búsqueda en la que variaran los parámetros a usar en el momento de asignar costes o heurísticas.

Para el UCS se desactivará el uso de heurística manteniendo solo el coste del problema, se realizará esta acción usando una heurística nula ya proporcionada la cual devolverá siempre 0.

En el caso del A* se hace uso de la función con todos sus parámetros activados, tanto la heurística que decidamos usar como el coste por defecto.

Finalmente, para el Best-H determinaremos los costes a 0 dejando como guía los valores de la heurística proporcionada.

El mecanismo de almacenamiento en la frontera ha sido una cola de prioridad ya proporcionada por el ejercicio.

En los algoritmos con uso de heurísticas se ha implementado el mecanismo de Path-max para trabajar fortalecer la heurística y que fuese monótona.

Para la búsqueda bidireccional se desarrollan dos algoritmos BFS, uno comenzando desde el inicio y otro desde el objetivo final.

La estrategia de búsqueda es ir expandiendo nodos hasta que ambos caminos se junten, para ello se comprueban si los nodos que actualmente están evaluando se encuentran ya en visitados por el otro algoritmo que está ejecutando a su vez.

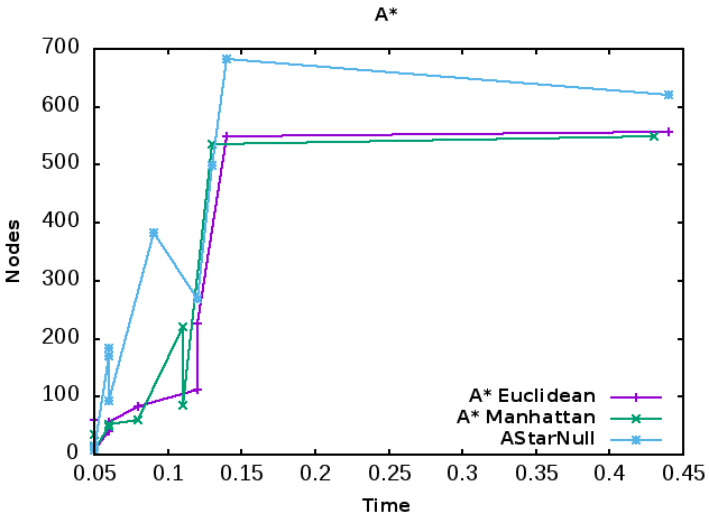
Para que el resultado sea satisfactorio hemos de tener en cuenta que el resultado del algoritmo que comienza desde el final está en estado “espejo”, por lo que deberemos cambiar el sentido de todas las acciones tomadas.

El algoritmo ha demostrado un funcionamiento correcto en los mapas proporcionados y en los creados por nosotros, el único fallo encontrado ha sido en el openMaze en el cual se queda expandiendo nodos sin llegar a juntar caminos. La razón por la que esto ocurre no ha sido clara y no se ha podido encontrar la causa del error.

Nota: Este algoritmo se encuentra desactivado por defecto para que la herramienta de corrección no fallase a la hora de analizar el openMaze. Se encuentra desactivada en la zona de abreviaciones.

A*	Euclidea		Manhattan	
Mapa	Time	Nodos	Time	Nodos
Amaze	0.06	41	0.05	34
bigmaze	0.44	557	0.43	549
bmaze	0.08	83	0.08	60
Cmaze	0.12	113	0.11	85
contourMaze	0.05	60	0.06	49
mediumMaze	0.12	226	0.11	221
openMaze	0.14	550	0.13	535
smallMaze	0.06	56	0.06	53
testMaze	0.05	7	0.05	7
tinyMaze	0.05	13	0.05	14

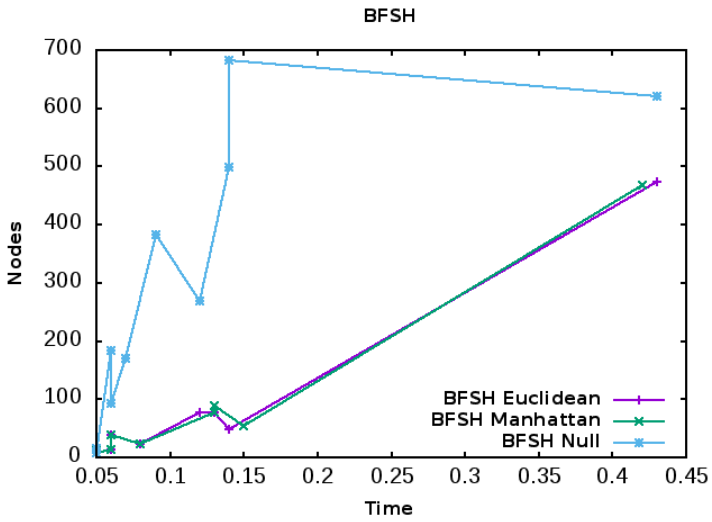
Tabla 1:Estadisticas



Se observa que el tiempo de ejecución del algoritmo con distintas heurísticas varia de forma mínima, lo mismo ocurre con su cantidad de nodos expandidos, donde solo en algún mapa puntual se nota la ventaja de Manhattan sobre la Euclidea.

BFSH	Manhattan		Euclidea	
Mapa	Time	Nodos	Time	Nodos
Amaze	0.06	14	0.06	14
bigmaze	0.42	468	0.43	473
bmaze	0.08	24	0.08	24
Cmaze	0.15	55	0.14	49
contourMaze	0.05	13	0.05	13
mediumMaze	0.13	78	0.12	78
openMaze	0.13	89	0.13	77
smallMaze	0.06	39	0.06	39
testMaze	0.05	7	0.05	7
tinyMaze	0.05	8	0.05	8

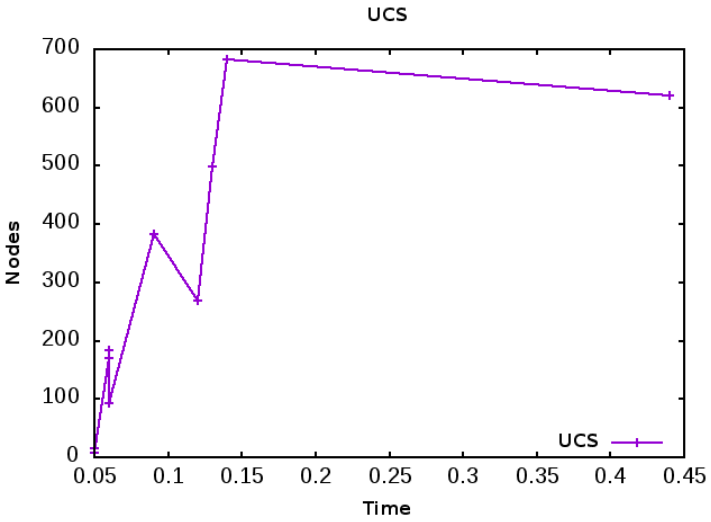
Tabla 2:Estadisticas BFSH



En este caso la variación entre el uso la heurística es mínima, tal y como se observa en la gráfica adjunta. La notable mejora con el uso de heurística es claro, en el otro caso se comportaría como un UCS tal y como se verá en la gráfica de este algoritmo.

UCS		
Mapa	Time	Nodos
Amaze	0.06	183
bigmaze	0.44	620
bmaze	0.09	383
Cmaze	0.13	498
contourMaze	0.06	170
mediumMaze	0.12	269
openMaze	0.14	682
smallMaze	0.06	92
testMaze	0.05	7
tinyMaze	0.05	15

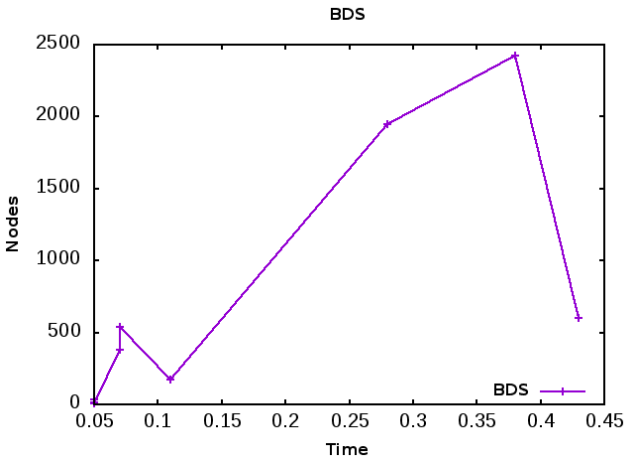
Tabla 4: Estadísticas UCS



En mapas de tamaño reducido la variación respecto a los demás algoritmos es minima, pero a medida que existen más estados a analizar la cantidad de nodos expandidos se dispara con el UCS. Notable diferencia en mapas como el Cmaze (Mapa propio) y openMaze respecto al uso del BFSH.

Bidireccional		
Mapa	Time	Nodos
Amaze	0.07	380
bigmaze	0.43	600
bmaze	0.38	2424
Cmaze	0.28	1946
contourMaze	0.07	540
mediumMaze	0.11	176
openMaze	-	-
smallMaze	0.05	34
testMaze	0.05	8
tinyMaze	0.05	14

Tabla 3: Estadísticas búsqueda bidireccional (Sin OpenMaze)



Datos relativos al funcionamiento correcto del algoritmo: La cantidad de nodos expandidos aumenta de manera gigantesca para algunos tipos de mapa como es el Bmaze y el Cmaze, ambos mapas generados aleatoriamente por el generador de mapas. Para la resta de mapas el comportamiento es parecido a los demás algoritmos analizados anteriormente.

Problema bien definido

Se toma como parámetros principales: Estado inicial y final ya definidos en el formato que se requiere (Lista de listas con los respectivos elementos en la posición correcta). Se ha definido esta entrada ya que no se especifica ningún estado fijo inicial ni final. De forma adicional se pasan los límites de cada elemento (Longitud, altura y bloques).

Ejemplo de un estado:

K → bloques = 4 (A,B,C,D) // M → posiciones = 5 // C → maxima altura = 2

[[A] , [B, C] , [ ] , [ ] , [D]]

El objetivo será obtener un estado idéntico al introducido como objetivo final por parámetros al inicio del problema.

Los costes de paso serán 1 por cada posición que se desee mover. Si tomamos el ejemplo de arriba, mover el bloque “A” a la última posición donde se encuentra el bloque “D”, tendría un coste de 4.

La acción definida es “mover” en la cual solamente se indica desde que posición se desea mover hasta que posición de destino. No se especifica el bloque por cuestiones de sencillez, este control se habrá de tener a la hora de aplicar sobre este problema un algoritmo.