TSI PRACTICA 1

Pablo Huertas Arroyo

16 de abril de 2022

UNIVERSIDAD DE GRANADA

Correo: phuertas@correo.ugr.es
DNI:77033078Y
Grupo 3A, subgrupo 2
Horario: Luneas de 17:30 a 19:30

1. Cuestiones a resolver

Entre BFS y DFS, ¿qué algoritmo puede ser considerado más eficiente de cara a encontrar el camino óptimo?

En este caso, el algoritmo más eficiente es el BFS, ya que es el que se encarga de encontrar el camino más corto al ir recorriendo todos los nodos nivel por nivel, terminando el algoritmo que encuentra el camino más corto. En este problema nos encontramos con que todos los movimientos que se realizar tienen el mismo coste (1), por lo que en este caso el algoritmo BFS nos da el camino más corto y con menor coste. En cambio, DFS nos da un camino de manera más rápida, pero no tiene por qué ser el camino más corto.

¿Se podría decir que A* es más eficiente que DFS?

Si nos referimos a tiempo, el algoritmo DFS tarda menos en encontrar un camino desde el nodo inicial hasta el nodo objetivo. Lo podemos observar en la tabla de resultados que en todos los mapas el tiempo empleado por el algoritmo DFS es menor que el tiempo empleado por el algoritmo A*. El algoritmo A* calcula el camino óptimo, pero de una manera mucho más costosa que requiere de más tiempo.

¿Cuáles son las principales diferencias entre A* e IDA*? ¿En qué contextos es más conveniente usar uno u otro?

A* es un algoritmo de búsqueda que, a través de una heurística, busca el camino más corto entre el nodo inicial y el nodo. Almacena muchos más nodos en memoria que IDA*, ya que el algoritmo de búsqueda A* guarda en memoria dos listas, una de Abiertos y otra de Cerrados. *Abiertos en mi caso una cola con prioridad, y Cerrados una tabla hash.* IDA* usa el planteamiento heurístico de A*, pero este en vez de realizar un recorrido en anchura de los nodos generados, realiza un recorrido en profundidad. Con cada iteración no se incrementa la profundidad de los niveles, sino el costo total del camino.ç IDA* no puede detectar estados repetidos ya que no cuenta con almacenamiento de estados como A*. En contextos donde se necesite mayor rapidez, se recomienda usar A*, siempre y cuando la limitación de memoria no sea un problema. En estos casos, donde la memoria es escasa, se recomienda usar IDA*, ya que no requiere prácticamente de almacenamiento de estados.

¿Se podría decir que RTA* es más eficiente que A*?

No, RTA* es un algoritmo que en memoria solo almacena los estados que han sido visitados, y tiene un espacio local de búsqueda muy limitado. Es decir, si nos imaginamos un pequeño robot que se mueve en una grilla de 10x10, y el sensor del robot tiene un alcance de 1 casilla, por cada paso que da el robot, solo podría ampliar su conocimiento de la grilla hasta un máximo de 4 casillas (suponiendo que puede ir hacia arriba, abajo, izquierda y derecha). El espacio local de aprendizaje sería la casilla del grid en la que se encuentra el robot, y que a medida que avanza puede ir aprendiendo más del entorno. Requiere de muy poca memoria, y en ocasiones puede ser bastante rápido, siempre y cuando no caiga en bucles intentando escapar de máximos locales. A* sabemos que necesita mucha memoria, pero tiene la capacidad de calcular un camino completo, sin necesidad de replanificar, y además encontrará el camino óptimo, siempre y cuando la heurística sea adecuada al problema.

2. Resultados

Algoritmo	Mapa	Runtime(acumulado)	Tamaño de la ruta calculada	Nº de nodos expandidos	Máximo nº de nodos en memoria
BFS	Muy pequeño	2	15	96	97
	Pequeño	4	34	130	131
	Mediano	10	110	846	846
	Grande	22	209	4632	4635
DFS	Muy pequeño	1	27	71	71
	Pequeño	2	64	81	81
	Mediano	6	238	402	402
	Grande	13	935	1213	1213
A*	Muy pequeño	2	15	37	55
	Pequeño	4	34	79	100
	Mediano	11	110	562	571
	Grande	27	209	3587	3674
IDA*	Muy pequeño	1	15	89	16
	Pequeño	4	34	513	35
	Mediano	328	110	498177	111
	Grande	-	-	-	-
RTA*	Muy pequeño				
	Pequeño				
	Mediano				
	Grande				

Figura 1: **Aclaración**: En el algoritmo IDA* en el mapa mediano se produce el Controller Disqualified, pero no lo he puesto como Timeout(TO) porque creo que es diferente
El algoritmo RTA* no he podido implementarlo por tiempo, por lo que se encuentra vacio