Escuela de
Ingeniería y Arquitectura
Universidad Zaragoza

Práctica 1

Resolución de problemas y búsqueda

Inteligencia Artificial

Grado en Ingeniería Informática

Autor

Javier Herrer Torres

Profesor

José Ángel Bañares

Escuela de Ingeniería y Arquitectura

Curso 2020/21

Escuela de Ingeniería y Arquitectura Universidad Zaragoza

Tabla de contenido

1 Introducción 2

2 8-Puzzle 3

2.1 Búsqueda primero en anchura 4

2.2 Búsqueda primero en profundidad 4

2.3 Búsqueda profundidad limitada 5

2.4 Búsqueda profundidad iterativa 5

2.5 Búsqueda coste uniforme 6

3 Misioneros y caníbales 6

4 Problema de las fichas 7

4.1 Búsqueda primero en anchura 8

4.2 Búsqueda profundidad limitada 9

4.3 Búsqueda profundidad iterativa 10

4.4 Búsqueda coste uniforme 11

1 INTRODUCCIÓN

El contenido de esta práctica versa sobre la aplicación de diferentes técnicas de búsqueda para los siguientes problemas: 8-puzzle, caníbales y misioneros, y el problema de las fichas. Para cada uno de estos problemas, haciendo uso de «aimacore», se tratará de: utilizar distintos algoritmos de búsqueda ciega, ser capaz de definir un problema y resolverlo con los algoritmos de búsqueda, y evaluar la aplicación de los algoritmos a los problemas y extraer algunas conclusiones.

Para el análisis y comparación de los diferentes algoritmos disponemos, a modo de resumen, de la siguiente tabla proporcionada en el material de la asignatura:

Criterio	Breadth-First	Uniform-cost	Depth-First	Depth-limited	Iterative deepening	Bidirectional search
Completitud	YES*	YES*	NO	YES, if $l \geq d$	YES	YES*
Temporal	b^{d+1}	$b^{C^*/e}$	b^m	b^l	b^d	$b^{d/2}$
Espacial	b^{d+1}	$b^{C^*/e}$	bm	bl	bd	$b^{d/2}$
Optimalidad	YES*	YES*	NO	NO	YES	YES

Tabla 1. Resumen de los diferentes algoritmos de búsqueda no informada

Además, antes de empezar con el análisis, es conveniente señalar que todos los problemas tienen un **espacio de estados finito**.

2 8-PUZZLE

Vamos a analizar los resultados obtenidos separados por algoritmo empleado. Para visualizar las métricas de la búsqueda se seguirá el código del enunciado de la práctica.

Problem	Depth	Expand	Q.Size	MaxQS	tiempo
BFS-G-3	3	5	4	5	5
BFS-T-3	3	6	9	10	0
DFS-G-3	59123	120491	39830	42913	500
DFS-T-3	---	---	---	---	(1)
DLS-9-3	9	10	0	0	0
DLS-3-3	3	4	0	0	0
IDS-3	3	9	0	0	0
UCS-F-3	3	16	9	10	1
UCS-T-3	3	32	57	58	0
BFS-G-9	9	288	198	199	1
BFS-T-9	9	5821	11055	11056	18
DFS-G-9	44665	141452	32012	42967	458
DFS-T-9	---	---	---	---	(1)
DLS-9-9	9	5474	0	0	4
DLS-3-9	0	12	0	0	0
IDS-9	9	9063	0	0	6
UCS-F-9	9	385	235	239	2
UCS-T-9	9	18070	31593	31594	22
BFS-G-30	30	181058	365	24048	408
BFS-T-30	---	---	---	---	(2)
DFS-G-30	62856	80569	41533	41534	331
DFS-T-30	---	---	---	---	(1)
DLS-9-30	0	4681	0	0	3
DLS-3-30	0	9	0	0	0
IDS-30	---	---	---	---	(2)
UCS-F-30	30	181390	49	24209	418
UCS-T-30	---	---	---	---	(2)

Tabla 2. Resultados obtenidos mediante distintos algoritmos en el problema del 8-puzzle

Procede aclarar el significado de los comentarios «(1)» y «(2)». El primero hace referencia a un problema para el que no se ha encontrado solución. El segundo indica que no se ha obtenido la solución al problema en un tiempo razonable.

2.1 BÚSQUEDA PRIMERO EN ANCHURA

Problema	Profundidad	Expand	Q.Size	MaxQS	tiempo
BFS-G-3	3	5	4	5	7
BFS-T-3	3	6	9	10	1
BFS-G-9	9	288	198	199	1
BFS-T-9	9	5821	11055	11056	12
BFS-G-30	30	181058	365	24048	449
BFS-T-30	---	---	---	---	(2)

Tabla 3. Resultados obtenidos en búsqueda primero en anchura

Podemos observar que se ha cumplido la completitud del algoritmo ya que encuentra «siempre» una solución al estar el nodo objetivo menos profundo en una profundidad finita *d*, y siendo *b* finito. Además, comparando los resultados obtenidos frente a otros algoritmos, podemos confirmar que siempre encuentra la solución óptima en número de pasos. Desgraciadamente, obtenemos unos malos resultados en complejidad temporal y espacial. De hecho, en el último caso (BFS-T-30) no obtenemos su resultado en un tiempo razonable.

2.2 BÚSQUEDA PRIMERO EN PROFUNDIDAD

Problema	Profundidad	Expand	Q.Size	MaxQS	tiempo
DFS-G-3	59123	120491	39830	42913	532
DFS-T-3	---	---	---	---	(1)
DFS-G-9	44665	141452	32012	42967	401
DFS-T-9	---	---	---	---	(1)
DFS-G-30	62856	80569	41533	41534	291
DFS-T-30	---	---	---	---	(1)

Tabla 4. Resultados obtenidos en búsqueda primero en profundidad

Esta búsqueda es claramente, comparada con el resto de los algoritmos, la que arroja resultados más altos numéricamente. Lo primero que refleja es que no hay completitud para las búsquedas en árbol ya que da lugar a bucles infinitos. Sin embargo, en las búsquedas en grafo sí que es completa ya que en el peor de los casos expandirá cada nodo. En ninguna de ellas será óptima.

Respecto a la complejidad, este algoritmo únicamente ofrece ventajas en la complejidad espacial para una búsqueda en árbol ya que únicamente necesita almacenar un único camino de la raíz a la hoja. Lamentablemente, no podemos obtener esos resultados al no encontrar una solución mediante ese método.

2.3 BÚSQUEDA PROFUNDIDAD LIMITADA

Problema	Profundidad	Expand	Q.Size	MaxQS	tiempo
DLS-9-3	9	10	0	0	2
DLS-3-3	3	4	0	0	0
DLS-9-9	9	5474	0	0	5
DLS-3-9	0	12	0	0	0
DLS-9-30	0	4681	0	0	2
DLS-3-30	0	9	0	0	0

Tabla 5. Resultados obtenidos en búsqueda profundidad limitada

En todos los casos, ha habido completitud y se ha encontrado solución al no ser $l < d$, siendo l la profundidad en la cual los nodos no tienen sucesores, y d la profundidad de la solución de menor coste.

Se obtienen unos buenos resultados tanto en la complejidad temporal como en la espacial. De hecho, esta última tiene un coste lineal.

2.4 BÚSQUEDA PROFUNDIDAD ITERATIVA

Problema	Profundidad	Expand	Q.Size	MaxQS	tiempo
IDS-3	3	9	0	0	0
IDS-9	9	9063	0	0	7
IDS-30	---	---	---	---	(2)

Tabla 6. Resultados obtenidos en búsqueda profundidad iterativa

Al no existir caminos infinitos, obtenemos solución para todos los problemas. Además, al igual que en la búsqueda en anchura, obtendremos el camino óptimo en número de pasos.

Este algoritmo tiene la ventaja de compartir la complejidad espacial con el de la búsqueda en profundidad, consiguiendo un coste lineal. Por otro lado, el coste temporal va a ser un problema, ya que en el último caso (IDS-30) no obtenemos la solución al problema en un tiempo razonable, igual que ocurría en BFS-T-30.

2.5 BÚSQUEDA COSTE UNIFORME

Problema	Profundidad	Expand	Q.Size	MaxQS	tiempo
UCS-F-3	3	16	9	10	1
UCS-T-3	3	32	57	58	0
UCS-F-9	9	385	235	239	2
UCS-T-9	9	18070	31593	31594	24
UCS-F-30	30	181390	49	24209	427
UCS-T-30	---	---	---	---	(2)

Tabla 7. Resultados obtenidos en búsqueda coste uniforme

Existe completitud ya que las acciones no tienen un coste nulo, de lo contrario podría dar lugar a bucles infinitos. Por otro lado, podemos confirmar que este algoritmo es óptimo al encontrar la solución de menor coste, esto es porque realiza el test objetivo al expandir y no al generar. Desgraciadamente, su complejidad espacial y temporal (son idénticas) no son ventajosas. De hecho, en el último caso (UCS-T-30) no obtenemos la solución al problema en un tiempo razonable.

3 MISIONEROS Y CANÍBALES

El método *agent.getActions()* devolverá la lista de acciones que llevaron al agente a encontrar la solución. También podrá devolver *NoOp* si ya estaba en el estado final al empezar, y una lista vacía en caso de no encontrar una solución. Posteriormente, llamando al método *executeActions* al final de *cannibalsSearch* mostraremos esas acciones una a una para los algoritmos: búsqueda primero en anchura, profundidad limitada, y profundidad iterativa.

Examinando la traza de ejecución del problema mediante los tres algoritmos podemos realizar las siguientes conclusiones. Existe completitud en todos ellos ya que encuentran la solución. El algoritmo de primero en anchura y de profundidad iterativa son óptimos, mientras que en el de profundidad limitada se le ha pasado el valor óptimo al agente de búsqueda.

La complejidad espacial será mayor en el algoritmo de búsqueda primero en anchura, ya que en los otros dos este coste es lineal. Y en la complejidad temporal se observa que el algoritmo primero en anchura encuentra la solución con un número de nodos expandidos muy inferior al resto.

Se ha decidido no representar en esta memoria la traza de la ejecución del programa debido a su extensión. En su lugar, esta se puede consultar tanto en el enunciado de la práctica como en la salida por pantalla resultante de la ejecución de *CanibalesPract1.java*.

4 PROBLEMA DE LAS FICHAS

Igual que ocurría en el anterior problema, el camino que siguen los cuatro algoritmos que se han empleado es el mismo. Se observan los mismos resultados en complejidad tanto espacial como temporal comentados en los dos puntos anteriores.

El problema se ha construido en el paquete *aima.core.enviroment.Fichas* en el que se han creado tres clases. La clase *FichasBoard.java* en la que se definen las acciones, la tupla de estado, y las operaciones que realizan los movimientos. La clase *FichasFunctionFactory.java* en la que encontramos las funciones *actionFunction*, *resultFunction*, y *stepCostFunction*. Y, por último, la clase *FichasGoalTest.java* que define el test objetivo.

Existen 6 acciones que consisten en mover el hueco del tablero a izquierda o derecha un número de posiciones que, como máximo, será 3 ya que el problema restringe explícitamente que el número de fichas que pueden ser saltadas es de un máximo de 2. Se deberá asegurar también que al realizar estas acciones no estamos moviendo la ficha fuera del tablero.

El coste de las acciones será igual al número de posiciones que se desplaza el hueco. Esto se ha especificado en la función *stepCostFunction*.

4.1 BÚSQUEDA PRIMERO EN ANCHURA

```
Problema de las fichas BFS -->
pathCost : 24.0
nodesExpanded : 134
queueSize : 3
maxQueueSize : 27
Tiempo:56

SOLUCIÓN:
GOAL STATE
       +---+---+---+---+---+---+---+
       | V | V | V |   | B | B | B |
       +---+---+---+---+---+---+---+
CAMINO ENCONTRADO
ESTADO INICIAL
       +---+---+---+---+---+---+---+
       | B | B | B |   | V | V | V |
       +---+---+---+---+---+---+---+
Action[name==Left2]
       +---+---+---+---+---+---+---+
       | B |   | B | B | V | V | V |
       +---+---+---+---+---+---+---+
Action[name==Right3]
       +---+---+---+---+---+---+---+
       | B | V | B | B |   | V | V |
       +---+---+---+---+---+---+---+
Action[name==Left2]
       +---+---+---+---+---+---+---+
       | B | V |   | B | B | V | V |
       +---+---+---+---+---+---+---+
Action[name==Right3]
       +---+---+---+---+---+---+---+
       | B | V | V | B | B |   | V |
       +---+---+---+---+---+---+---+
Action[name==Right1]
       +---+---+---+---+---+---+---+
       | B | V | V | B | B | V |   |
       +---+---+---+---+---+---+---+
Action[name==Left3]
       +---+---+---+---+---+---+---+
       | B | V | V |   | B | V | B |
       +---+---+---+---+---+---+---+
Action[name==Left3]
       +---+---+---+---+---+---+---+
       |   | V | V | B | B | V | B |
       +---+---+---+---+---+---+---+
Action[name==Right2]
       +---+---+---+---+---+---+---+
       | V | V |   | B | B | V | B |
       +---+---+---+---+---+---+---+
Action[name==Right3]
       +---+---+---+---+---+---+---+
       | V | V | V | B | B |   | B |
       +---+---+---+---+---+---+---+
Action[name==Left2]
       +---+---+---+---+---+---+---+
       | V | V | V |   | B | B | B |
       +---+---+---+---+---+---+---+
```

4.2 BÚSQUEDA PROFUNDIDAD LIMITADA

```
Problema de las fichas DLS(10) -->
pathCost : 24.0
nodesExpanded : 394681
queueSize : 0
maxQueueSize : 0
Tiempo:585

SOLUCIÓN:
GOAL STATE
       +---+---+---+---+---+---+---+
       | V | V | V |   | B | B | B |
       +---+---+---+---+---+---+---+
CAMINO ENCONTRADO
ESTADO INICIAL
       +---+---+---+---+---+---+---+
       | B | B | B |   | V | V | V |
       +---+---+---+---+---+---+---+
Action[name==Left2]
       +---+---+---+---+---+---+---+
       | B |   | B | B | V | V | V |
       +---+---+---+---+---+---+---+
Action[name==Right3]
       +---+---+---+---+---+---+---+
       | B | V | B | B |   | V | V |
       +---+---+---+---+---+---+---+
Action[name==Left2]
       +---+---+---+---+---+---+---+
       | B | V |   | B | B | V | V |
       +---+---+---+---+---+---+---+
Action[name==Right3]
       +---+---+---+---+---+---+---+
       | B | V | V | B | B |   | V |
       +---+---+---+---+---+---+---+
Action[name==Right1]
       +---+---+---+---+---+---+---+
       | B | V | V | B | B | V |   |
       +---+---+---+---+---+---+---+
Action[name==Left3]
       +---+---+---+---+---+---+---+
       | B | V | V |   | B | V | B |
       +---+---+---+---+---+---+---+
Action[name==Left3]
       +---+---+---+---+---+---+---+
       |   | V | V | B | B | V | B |
       +---+---+---+---+---+---+---+
Action[name==Right2]
       +---+---+---+---+---+---+---+
       | V | V |   | B | B | V | B |
       +---+---+---+---+---+---+---+
Action[name==Right3]
       +---+---+---+---+---+---+---+
       | V | V | V | B | B |   | B |
       +---+---+---+---+---+---+---+
Action[name==Left2]
       +---+---+---+---+---+---+---+
       | V | V | V |   | B | B | B |
       +---+---+---+---+---+---+---+
```

4.3 BÚSQUEDA PROFUNDIDAD ITERATIVA

```
Problema de las fichas IDLS -->
pathCost : 24.0
nodesExpanded : 724476
queueSize : 0
maxQueueSize : 0
Tiempo:782

SOLUCIÓN:
GOAL STATE
       +---+---+---+---+---+---+---+
       | V | V | V |   | B | B | B |
       +---+---+---+---+---+---+---+
CAMINO ENCONTRADO
ESTADO INICIAL
       +---+---+---+---+---+---+---+
       | B | B | B |   | V | V | V |
       +---+---+---+---+---+---+---+
Action[name==Left2]
       +---+---+---+---+---+---+---+
       | B |   | B | B | V | V | V |
       +---+---+---+---+---+---+---+
Action[name==Right3]
       +---+---+---+---+---+---+---+
       | B | V | B | B |   | V | V |
       +---+---+---+---+---+---+---+
Action[name==Left2]
       +---+---+---+---+---+---+---+
       | B | V |   | B | B | V | V |
       +---+---+---+---+---+---+---+
Action[name==Right3]
       +---+---+---+---+---+---+---+
       | B | V | V | B | B |   | V |
       +---+---+---+---+---+---+---+
Action[name==Right1]
       +---+---+---+---+---+---+---+
       | B | V | V | B | B | V |   |
       +---+---+---+---+---+---+---+
Action[name==Left3]
       +---+---+---+---+---+---+---+
       | B | V | V |   | B | V | B |
       +---+---+---+---+---+---+---+
Action[name==Left3]
       +---+---+---+---+---+---+---+
       |   | V | V | B | B | V | B |
       +---+---+---+---+---+---+---+
Action[name==Right2]
       +---+---+---+---+---+---+---+
       | V | V |   | B | B | V | B |
       +---+---+---+---+---+---+---+
Action[name==Right3]
       +---+---+---+---+---+---+---+
       | V | V | V | B | B |   | B |
       +---+---+---+---+---+---+---+
Action[name==Left2]
       +---+---+---+---+---+---+---+
       | V | V | V |   | B | B | B |
       +---+---+---+---+---+---+---+
```

4.4 BÚSQUEDA COSTE UNIFORME

```
Problema de las fichas UCS-f -->
pathCost : 24.0
nodesExpanded : 139
queueSize : 0
maxQueueSize : 25
Tiempo:6

SOLUCIÓN:
GOAL STATE
       +---+---+---+---+---+---+---+
       | V | V | V |   | B | B | B |
       +---+---+---+---+---+---+---+
CAMINO ENCONTRADO
ESTADO INICIAL
       +---+---+---+---+---+---+---+
       | B | B | B |   | V | V | V |
       +---+---+---+---+---+---+---+
Action[name==Left2]
       +---+---+---+---+---+---+---+
       | B |   | B | B | V | V | V |
       +---+---+---+---+---+---+---+
Action[name==Right3]
       +---+---+---+---+---+---+---+
       | B | V | B | B |   | V | V |
       +---+---+---+---+---+---+---+
Action[name==Left2]
       +---+---+---+---+---+---+---+
       | B | V |   | B | B | V | V |
       +---+---+---+---+---+---+---+
Action[name==Right3]
       +---+---+---+---+---+---+---+
       | B | V | V | B | B |   | V |
       +---+---+---+---+---+---+---+
Action[name==Right1]
       +---+---+---+---+---+---+---+
       | B | V | V | B | B | V |   |
       +---+---+---+---+---+---+---+
Action[name==Left3]
       +---+---+---+---+---+---+---+
       | B | V | V |   | B | V | B |
       +---+---+---+---+---+---+---+
Action[name==Left3]
       +---+---+---+---+---+---+---+
       |   | V | V | B | B | V | B |
       +---+---+---+---+---+---+---+
Action[name==Right2]
       +---+---+---+---+---+---+---+
       | V | V |   | B | B | V | B |
       +---+---+---+---+---+---+---+
Action[name==Right3]
       +---+---+---+---+---+---+---+
       | V | V | V | B | B |   | B |
       +---+---+---+---+---+---+---+
Action[name==Left2]
       +---+---+---+---+---+---+---+
       | V | V | V |   | B | B | B |
       +---+---+---+---+---+---+---+
```