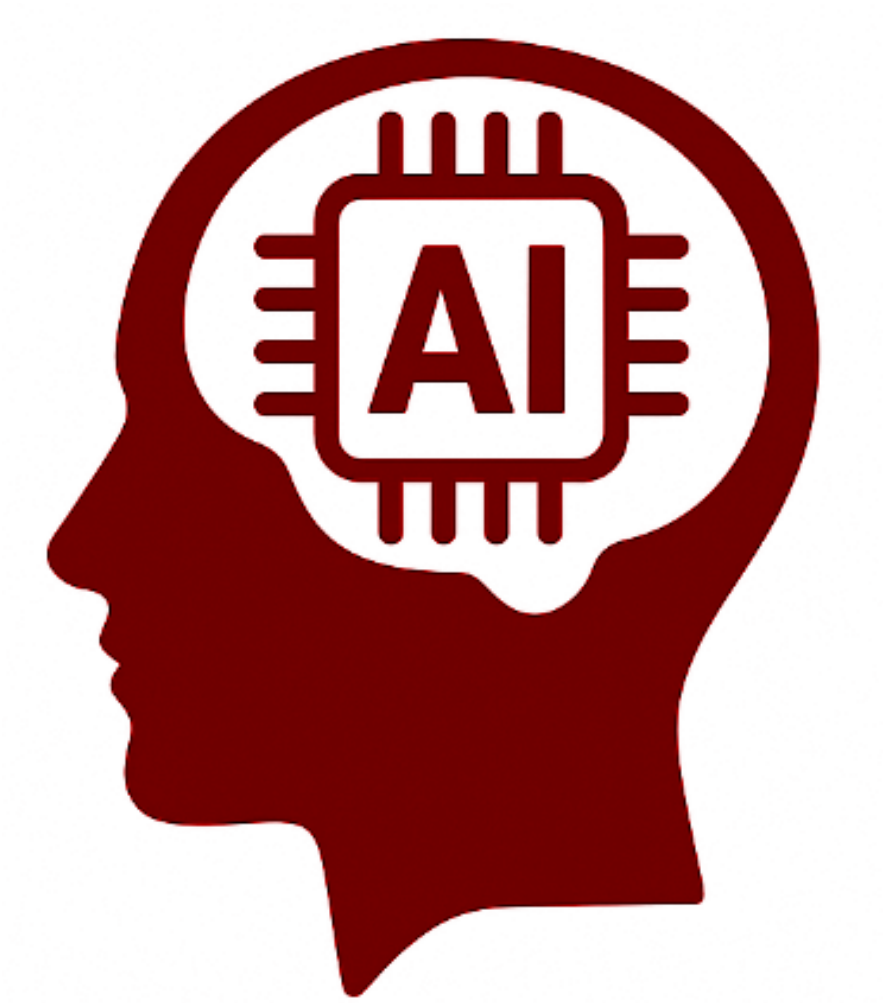
Entrega Práctica 1 - Inteligencia Artificial

Resolución de problemas y búsqueda: Búsqueda no informada.



Autor: Fernando Pastor Peralta (897113)

Fecha de entrega: 06/10/2025

Grupo: Martes B, 16:00-19:00

Ingeniería Informática - Universidad de Zaragoza, Curso 2025-2026

Índice

Introducción: **3**
Metodología: **3**
Problema del 8-puzzle: 3
Problema de los misioneros y los caníbales: 5
Conclusión: **5**
Anexo 1: **6**
Tabla con métricas de distintos algoritmos de búsqueda sobre 8-puzzle: 6
Anexo 2: **7**
Acciones de algoritmos de búsqueda no informada sobre misioneros y caníbales: 7

Introducción:

El objetivo de esta práctica es familiarizarse con el código en java para la resolución de problemas de búsqueda. Entenderemos en profundidad cómo funcionan los algoritmos utilizados y cómo desarrollar el entorno para la resolución de estos problemas.

Utilizaremos distintas técnicas de búsqueda ciega para los problemas del 8-puzzle y el de los Misioneros y Caníbales (tendremos que definir este problema) y posteriormente tomaremos métricas de ambos problemas, con los resultados de cada algoritmo, para sacar conclusiones de cuales son los más adecuados en cada uno.

Las métricas obtenidas se podrán consultar en los anexos.

Metodología:

Problema del 8-puzzle:

Para empezar, se ha realizado el análisis y la comprensión de los códigos que definen el problema: EightPuzzleBoard (estado), EightPuzzleFunctionFactory (acciones) y EightPuzzleGoalTest (estado objetivo). Después se ha ejecutado la demo (EightPuzzleDemo), que muestra la ejecución del 8-puzzle con distintos algoritmos de búsqueda, tanto no informada (DLS y IDDFS), como informada (Greedy Best-First con MisplacedTilleHeuristic y ManhattanHeuristic, A* con mismas heurísticas y Temple simulado).

De esta demo, podemos sacar algunas conclusiones:

1. El DLS al tener un límite justo, encuentra la mejor solución con muy pocos nodos generados/expandidos (si el límite se quedase corto, sería un CutOff, si no, si nos pasamos, podríamos encontrar una solución que no es la mejor y con mayor coste).

2. El Iterative DLS tiene que generar y expandir más nodos, ya que cada vez aumenta el límite del DLS (límite = 0,1,..), hasta encontrar la primera solución (que será la más óptima).

3. Greedy Best-First con ambas heurísticas da una solución en tres pasos, lo cuál es correcto, es muy eficiente, pero no se garantiza siempre la optimalidad de la solución.

4. En A*, llega al camino óptimo con ambas heurísticas, pero con gran diferencia entre ellas: con MispacedTitles, 25 expandidos y con Manhattan, 11 expandidos. Esto implica una superioridad de Manhattan sobre MisplacedTitles en el 8-puzzle (menos expansiones y menos memoria).

5. Simulated Annealing, puede fallar, en este caso falla tras 100 expandidos. Es útil para espacios enormes, cuando no exigimos garantías, pero en este caso, su uso no es deseado.

Luego, se ha diseñado nuestro propio demo en el que agrupamos todos los algoritmos disponibles de búsqueda no informada (BFS en grafo y árbol, DFS en grafo y árbol, DLS con profundidad 9 y 3, IDDFS y UCS en grafo y árbol.), probando con distintos estados iniciales con solución efectiva en 3, 9 y 30 pasos respectivamente. Se agrupan datos sobre: profundidad de la solución, nodos expandidos, cantidad de elementos en la cola frontera al momento de encontrar la solución, máximos elementos que ha habido en la cola frontera, tiempo, nodos generados y ritmo (nodos generados por segundo). Cabe destacar que aquellos algoritmos que no terminan en un tiempo razonable, o que ocupan demasiada memoria, serán marcados con (1) o (2), y al ser abortados, no se mostrarán datos sobre su ejecución.

La tabla con las métricas está disponible en el anexo 1.

De esta tabla podemos sacar ciertas conclusiones:

1. A medida que crece la profundidad de la solución (3, 9, 30), el trabajo crece de forma explosiva en todos los algoritmos, se ve perfectamente en BFS-GRAPH: 5, 288 y 181058 nodos expandidos respectivamente.
2. Los algoritmos con grafos contienen más crecimiento en memoria, los árboles tienen muchísimo más crecimiento en comparación (incluso muchos no llegan a terminar).
3. En coste de memoria los más eficientes son BFS-GRAPH o UCS-GRAPH.

Para el desarrollo de la demo, se ha creado una función eightPuzzleSearch(), en la que se recibe un tipo de algoritmo (cualquiera registrado en tipoSearch()) y un estado inicial del tablero, con ello se gestiona la ejecución del mismo, además de la impresión en pantalla de los resultados. Para la gestión de errores, se usa exceptions (tanto espaciales, como temporales), en cuanto a la gestión de tiempo, se lanza un hilo que ejecuta la búsqueda, si este hilo no termina antes de un tiempo determinado, se lanza un TimeoutException y se para el hilo. Desde el método main, se lanzan todos los algoritmos, con los tres distintos estados iniciales.

En cuanto costes temporales, teóricamente sabemos que:

- BFS: coste temporal = O(b^d) y coste espacial = O(b^d),
  con b = factor de ramificación y d = profundidad de la solución.
- DFS: coste temporal = O(b^m) y coste espacial = O(b*m),
  con m = máxima profundidad del árbol
- DLS: coste temporal = O(b^l) y coste espacial = O(b*l),
  con l = límite de profundidad
- IDDFS: coste temporal = O(b^d) y coste espacial = O(b*d).
- UCS: coste temporal = O(b^(1 + ⌊C*/e⌋)) y coste espacial = O(b^(1 + ⌊C*/e⌋)).

Como se ha visto, los resultados concuerdan con la teoría.

Problema de los misioneros y los caníbales:

Con lo ya aprendido hasta el momento, desarrollamos el código de definición del problema de los misioneros y caníbales, además de la demo para ver las métricas y la evolución de las acciones en cada caso:

- CanibalesBoard: Se define el estado actual del juego, se guardan los caníbales de un lado y con esa información se conocen los del otro (y se podrán consultar).

- CanibalesFunctionFactory: Se definen las distintas acciones posibles, y métodos para poder obtener la lista de estados siguientes posibles, para cambiar el estado de un problema tras hacer una acción y para conocer el coste de cada acción (aunque es 1 para todos).

- CanibalesGoalTest: Sirve para conocer si el estado actual es el estado objetivo o no.

- CanibalesPract1: Es nuestra demo en la que ejecutamos los algoritmos de búsqueda no informada sobre un estado inicial con 3 caníbales y 3 misioneros en la orilla izquierda. En diseño, es muy similar a la demo del 8-puzzle, sin embargo, se pone más detalle en la presentación por pantalla usando varias funciones.

La tabla con las métricas y acciones hasta llegar al estado objetivo, está disponible en el anexo 2.

DFS-TREE no llega a terminar antes del tiempo establecido, los DLS no encuentran solución y lanzan un CutOff. Todos los demás algoritmos, encuentran la solución óptima con 11 acciones. UCS-GRAPH, BFS-GRAPH y DFS-GRAPH son las más óptimas tanto temporal como espacialmente.

Conclusión:

He aprendido a definir problemas de exploración, y sobre las especificaciones de cada algoritmo de búsqueda no informada (y algunos de informada) en detalle.

Dependiendo del problema, algunos algoritmos son más óptimos que otros, y merece la pena conocer en detalle cada algoritmo, ya que se pueden usar específicamente para los entornos de búsqueda en nuestro problema, haciendo que las búsquedas sean más rápidas y/o con mayor eficiencia en el uso de espacio de memoria.

También hemos visto que en ocasiones merece la pena hacer programas para comparar qué método es el más efectivo para distintos casos iniciales de nuestro problema.

Anexo 1:

Tabla con métricas de distintos algoritmos de búsqueda sobre 8-puzzle:

Problema	Profundidad	Expand	Q.Size	MaxQS	Tiempo(ms)	Generados	Ritmo
BFS-GRAPH-3	3	5	4	5	6,533	14	2142,9
BFS-TREE-3	3	6	9	10	0,658	17	25847,7
DFS-GRAPH-3	59123	120491	39830	42913	405,988	321072	790841,1
DFS-TREE-3	---	---	---	---	(1)	---	---
DLS-9-3	9	10	0	0	1,386	26	18763,1
DLS-3-3	3	4	0	0	0,289	11	38036,0
IDDFS-3	3	9	0	0	0,435	25	57484,5
UCS-GRAPH-3	3	16	9	10	209,681	40	190,8
UCS-TREE-3	3	32	57	58	0,535	89	166355,1
BFS-GRAPH-9	9	288	198	199	1,541	801	519657,5
BFS-TREE-9	9	5821	11055	11056	9,742	16878	1732551,8
DFS-GRAPH-9	44665	141452	32012	42967	340,317	374614	1100781,2
DFS-TREE-9	---	---	---	---	(1)	---	---
DLS-9-9	9	5474	0	0	4,249	15924	3747881,8
DLS-3-9	0	12	0	0	0,218	35	160256,4
IDDFS-9	9	9063	0	0	6,207	25940	4179489,2
UCS-GRAPH-9	9	385	235	239	1,842	1040	564757,0
UCS-TREE-9	9	18070	31593	31594	20,283	49663	2448515,7
BFS-GRAPH-30	30	181058	365	24048	340,332	482871	1418824,6
BFS-TREE-30	---	---	---	---	(1)	---	---
DFS-GRAPH-30	62856	80569	41533	41534	258,271	219540	850036,0
DFS-TREE-30	---	---	---	---	(1)	---	---
DLS-9-30	0	4681	0	0	2,537	12872	5073509,1
DLS-3-30	0	9	0	0	0,251	24	95617,5
IDDFS-30	---	---	---	---	(1)	---	---
UCS-GRAPH-30	30	181390	49	24209	348,067	483730	1389760,8
UCS-TREE-30	---	---	---	---	(1)	---	---

Anexo 2:

Acciones de algoritmos de búsqueda no informada sobre misioneros y caníbales:

```
Misioneros y caníbales BFS-GRAPH -->
pathCost : 11.0
nodesExpanded : 13
queueSize : 1
maxQueueSize : 3
Tiempo:7ms

SOLUCIÓN:
GOAL STATE
RIBERA-IZDA --RIO-- BOTE M M M C C C RIBERA-DCHA

CAMINO ENCONTRADO
ESTADO INICIAL  RIBERA-IZDA M M M C C C BOTE --RIO-- RIBERA-DCHA
        Action[name==M2C]       RIBERA-IZDA M M M C --RIO-- BOTE C C RIBERA-DCHA
        Action[name==M1C]       RIBERA-IZDA M M M C C BOTE --RIO-- C RIBERA-DCHA
        Action[name==M2C]       RIBERA-IZDA M M M --RIO-- BOTE C C C RIBERA-DCHA
        Action[name==M1C]       RIBERA-IZDA M M M C BOTE --RIO-- C C RIBERA-DCHA
        Action[name==M2M]       RIBERA-IZDA M C --RIO-- BOTE M M C C RIBERA-DCHA
        Action[name==M1M1C]     RIBERA-IZDA M M C C BOTE --RIO-- M C RIBERA-DCHA
        Action[name==M2M]       RIBERA-IZDA C C --RIO-- BOTE M M M C RIBERA-DCHA
        Action[name==M1C]       RIBERA-IZDA C C C BOTE --RIO-- M M M RIBERA-DCHA
        Action[name==M2C]       RIBERA-IZDA C --RIO-- BOTE M M M C C RIBERA-DCHA
        Action[name==M1M]       RIBERA-IZDA M C BOTE --RIO-- M M C C RIBERA-DCHA
        Action[name==M1M1C]     RIBERA-IZDA --RIO-- BOTE M M M C C C RIBERA-DCHA

Misioneros y caníbales BFS-TREE -->
pathCost : 11.0
nodesExpanded : 5078
queueSize : 6673
maxQueueSize : 6674
Tiempo:18ms

SOLUCIÓN:
GOAL STATE
RIBERA-IZDA --RIO-- BOTE M M M C C C RIBERA-DCHA

CAMINO ENCONTRADO
ESTADO INICIAL  RIBERA-IZDA M M M C C C BOTE --RIO-- RIBERA-DCHA
        Action[name==M2C]       RIBERA-IZDA M M M C --RIO-- BOTE C C RIBERA-DCHA
        Action[name==M1C]       RIBERA-IZDA M M M C C BOTE --RIO-- C RIBERA-DCHA
        Action[name==M2C]       RIBERA-IZDA M M M --RIO-- BOTE C C C RIBERA-DCHA
        Action[name==M1C]       RIBERA-IZDA M M M C BOTE --RIO-- C C RIBERA-DCHA
        Action[name==M2M]       RIBERA-IZDA M C --RIO-- BOTE M M C C RIBERA-DCHA
        Action[name==M1M1C]     RIBERA-IZDA M M C C BOTE --RIO-- M C RIBERA-DCHA
        Action[name==M2M]       RIBERA-IZDA C C --RIO-- BOTE M M M C RIBERA-DCHA
        Action[name==M1C]       RIBERA-IZDA C C C BOTE --RIO-- M M M RIBERA-DCHA
        Action[name==M2C]       RIBERA-IZDA C --RIO-- BOTE M M M C C RIBERA-DCHA
        Action[name==M1M]       RIBERA-IZDA M C BOTE --RIO-- M M C C RIBERA-DCHA
        Action[name==M1M1C]     RIBERA-IZDA --RIO-- BOTE M M M C C C RIBERA-DCHA
```

```
Misioneros y caníbales DFS-GRAPH -->
pathCost : 11.0
nodesExpanded : 11
queueSize : 3
maxQueueSize : 4
Tiempo:2ms

SOLUCIÓN:
GOAL STATE
RIBERA-IZDA --RIO-- BOTE M M M C C C RIBERA-DCHA

CAMINO ENCONTRADO
ESTADO INICIAL  RIBERA-IZDA M M M C C C BOTE --RIO-- RIBERA-DCHA
        Action[name==M1M1C]     RIBERA-IZDA M M C C --RIO-- BOTE M C RIBERA-DCHA
        Action[name==M1M]       RIBERA-IZDA M M M C C BOTE --RIO-- C RIBERA-DCHA
        Action[name==M2C]       RIBERA-IZDA M M M --RIO-- BOTE C C C RIBERA-DCHA
        Action[name==M1C]       RIBERA-IZDA M M M C BOTE --RIO-- C C RIBERA-DCHA
        Action[name==M2M]       RIBERA-IZDA M C --RIO-- BOTE M M C C RIBERA-DCHA
        Action[name==M1M1C]     RIBERA-IZDA M M C C BOTE --RIO-- M C RIBERA-DCHA
        Action[name==M2M]       RIBERA-IZDA C C --RIO-- BOTE M M M C RIBERA-DCHA
        Action[name==M1C]       RIBERA-IZDA C C C BOTE --RIO-- M M M RIBERA-DCHA
        Action[name==M2C]       RIBERA-IZDA C --RIO-- BOTE M M M C C RIBERA-DCHA
        Action[name==M1C]       RIBERA-IZDA C C BOTE --RIO-- M M M C RIBERA-DCHA
        Action[name==M2C]       RIBERA-IZDA --RIO-- BOTE M M M C C C RIBERA-DCHA

Misioneros y caníbales DFS-TREE -->
pathCost : (1)
nodesExpanded : -
queueSize : -
maxQueueSize : -
Tiempo:8136ms

Misioneros y caníbales DLS-9 -->
pathCost : 0
nodesExpanded : 1463
queueSize : 0
maxQueueSize : 0
Tiempo:4ms

Search Outcome=CUTOFF

Misioneros y caníbales DLS-3 -->
pathCost : 0
nodesExpanded : 9
queueSize : 0
maxQueueSize : 0
Tiempo:0ms

Search Outcome=CUTOFF
```

```
Misioneros y caníbales IDDFS -->
pathCost : 11.0
nodesExpanded : 9010
queueSize : 0
maxQueueSize : 0
Tiempo:7ms

SOLUCIÓN:
GOAL STATE
RIBERA-IZDA --RIO-- BOTE M M M C C C RIBERA-DCHA

CAMINO ENCONTRADO
ESTADO INICIAL  RIBERA-IZDA M M M C C C BOTE --RIO-- RIBERA-DCHA
        Action[name==M2C]       RIBERA-IZDA M M M C --RIO-- BOTE C C RIBERA-DCHA
        Action[name==M1C]       RIBERA-IZDA M M M C C BOTE --RIO-- C RIBERA-DCHA
        Action[name==M2C]       RIBERA-IZDA M M M --RIO-- BOTE C C C RIBERA-DCHA
        Action[name==M1C]       RIBERA-IZDA M M M C BOTE --RIO-- C C RIBERA-DCHA
        Action[name==M2M]       RIBERA-IZDA M C --RIO-- BOTE M M C C RIBERA-DCHA
        Action[name==M1M1C]     RIBERA-IZDA M M C C BOTE --RIO-- M C RIBERA-DCHA
        Action[name==M2M]       RIBERA-IZDA C C --RIO-- BOTE M M M C RIBERA-DCHA
        Action[name==M1C]       RIBERA-IZDA C C C BOTE --RIO-- M M M RIBERA-DCHA
        Action[name==M2C]       RIBERA-IZDA C --RIO-- BOTE M M M C C RIBERA-DCHA
        Action[name==M1M]       RIBERA-IZDA M C BOTE --RIO-- M M C C RIBERA-DCHA
        Action[name==M1M1C]     RIBERA-IZDA --RIO-- BOTE M M M C C C RIBERA-DCHA

Misioneros y caníbales UCS-GRAPH -->
pathCost : 11.0
nodesExpanded : 14
queueSize : 0
maxQueueSize : 3
Tiempo:1ms

SOLUCIÓN:
GOAL STATE
RIBERA-IZDA --RIO-- BOTE M M M C C C RIBERA-DCHA

CAMINO ENCONTRADO
ESTADO INICIAL  RIBERA-IZDA M M M C C C BOTE --RIO-- RIBERA-DCHA
        Action[name==M1M1C]     RIBERA-IZDA M M C C --RIO-- BOTE M C RIBERA-DCHA
        Action[name==M1M]       RIBERA-IZDA M M M C C BOTE --RIO-- C RIBERA-DCHA
        Action[name==M2C]       RIBERA-IZDA M M M --RIO-- BOTE C C C RIBERA-DCHA
        Action[name==M1C]       RIBERA-IZDA M M M C BOTE --RIO-- C C RIBERA-DCHA
        Action[name==M2M]       RIBERA-IZDA M C --RIO-- BOTE M M C C RIBERA-DCHA
        Action[name==M1M1C]     RIBERA-IZDA M M C C BOTE --RIO-- M C RIBERA-DCHA
        Action[name==M2M]       RIBERA-IZDA C C --RIO-- BOTE M M M C RIBERA-DCHA
        Action[name==M1C]       RIBERA-IZDA C C C BOTE --RIO-- M M M RIBERA-DCHA
        Action[name==M2C]       RIBERA-IZDA C --RIO-- BOTE M M M C C RIBERA-DCHA
        Action[name==M1M]       RIBERA-IZDA M C BOTE --RIO-- M M C C RIBERA-DCHA
        Action[name==M1M1C]     RIBERA-IZDA --RIO-- BOTE M M M C C C RIBERA-DCHA
```

```
Misioneros y caníbales UCS-TREE -->
pathCost : 11.0
nodesExpanded : 18153
queueSize : 20458
maxQueueSize : 20459
Tiempo:14ms

SOLUCIÓN:
GOAL STATE
RIBERA-IZDA --RIO-- BOTE M M M C C C RIBERA-DCHA

CAMINO ENCONTRADO
ESTADO INICIAL  RIBERA-IZDA M M M C C C BOTE --RIO-- RIBERA-DCHA
        Action[name==M1M1C]     RIBERA-IZDA M M C C --RIO-- BOTE M C RIBERA-DCHA
        Action[name==M1M]       RIBERA-IZDA M M M C C BOTE --RIO-- C RIBERA-DCHA
        Action[name==M2C]       RIBERA-IZDA M M M --RIO-- BOTE C C C RIBERA-DCHA
        Action[name==M1C]       RIBERA-IZDA M M M C BOTE --RIO-- C C RIBERA-DCHA
        Action[name==M2M]       RIBERA-IZDA M C --RIO-- BOTE M M C C RIBERA-DCHA
        Action[name==M1M1C]     RIBERA-IZDA M M C C BOTE --RIO-- M C RIBERA-DCHA
        Action[name==M2M]       RIBERA-IZDA C C --RIO-- BOTE M M M C RIBERA-DCHA
        Action[name==M1C]       RIBERA-IZDA C C C BOTE --RIO-- M M M RIBERA-DCHA
        Action[name==M2C]       RIBERA-IZDA C --RIO-- BOTE M M M C C RIBERA-DCHA
        Action[name==M1M]       RIBERA-IZDA M C BOTE --RIO-- M M C C RIBERA-DCHA
        Action[name==M1M1C]     RIBERA-IZDA --RIO-- BOTE M M M C C C RIBERA-DCHA
```