

Universidad Simón Bolívar

Departamento de Computación y Tecnología de la Información

CI5437 - Inteligencia Artificial I – Ene-Mar 2018

Prof. Blai Bonet

Proyecto 1

Yarima Luciani – 13-10770

Lautaro Villalón – 12-10427

Top Spin

Heurística: Se generó una max PDB utilizando tres abstracciones distintas por cada modalidad del rompecabezas. Se *mappearon* diferentes variables de estados para simplificar el problema, dejando solamente estados impares/pares e igualando diferentes grupos de variables a una sola variable de estado.

Algoritmos de búsqueda: En general, los algoritmos A* e IDA* (ambos con eliminación parcial de duplicados) se comportaron de manera esperada, siendo el último más eficiente en torno a velocidad y memoria. Sin embargo, se notó una gran diferencia en los tiempos de corrida y consumo de memoria entre la modalidad 17-4 y las otras, siendo estos más lentos y pesados.

Árbol de búsqueda:

| Modalidad | Pruning | Profundidad | Tiempo | ¿Sin memoria? | ¿Usó Swap? |
|-----------|---------|-------------|--------|---------------|------------|
| 12-4 | No | 7 | 1 min | Sí | Sí |
| 12-4 | Sí | 9 | 9 min | Sí | Sí |
| 14-4 | No | 6 | 40 seg | Sí | Sí |
| 14-4 | Sí | 7 | 15 min | No | Sí |
| 17-4 | No | 6 | 30 seg | Sí | Sí |
| 17-4 | Sí | 6 | 15 min | No | Sí |

Cubo de Rubik

Heurística: Se generó una max PDB donde se realizó una abstracción para mantener las piezas de las esquinas y dos para mantener las de los lados. En todas, se dividió el problema en tres colores para evitar que fuese excesivamente pesado.

Algoritmos de búsqueda: Al igual que el caso anterior, hubo un comportamiento esperado. Con algunos problemas para correr los algoritmos por la complejidad de los casos de prueba.

Árbol de búsqueda:

| Modalidad | Pruning | Profundidad | Tiempo | ¿Sin memoria? | ¿Usó Swap? |
|-----------|---------|-------------|--------|---------------|------------|
| 3x3x3 | No | 5 | 20 seg | Sí | Sí |
| 3x3x3 | Sí | 4 | 15 min | No | Sí |

15-Puzzle

Heurística: Se generó una PDB aditiva, utilizando tres abstracciones. En cada una, se *mappeó* el problema para que quedaran 5 *tiles*, más el blanco, de forma disjunta entre las abstracciones. Aparte, se implementó la heurística Manhattan para comparaciones.

Algoritmos de búsqueda: A* e IDA* obtuvieron un comportamiento comparativo parecido al de Top Spin. Sin embargo, al comparar PDB con Manhattan, obtuvimos resultados variados, hubo casos en que con Manhattan lograba terminar antes y otros en que con PDB terminaba antes, sin embargo, en general, con la PDB era más rápido y eficiente en cantidad de nodos abiertos.

Árbol de búsqueda:

| Modalidad | Pruning | Profundidad | Tiempo | ¿Sin memoria? | ¿Usó Swap? |
|-----------|---------|-------------|--------|---------------|------------|
| 15-Puzzle | No | 16 | 30 seg | Sí | No |
| 15-Puzzle | Sí | 24 | 2 min | Sí | No |

24-Puzzle

Heurística: Se generó una PDB aditiva, utilizando cuatro abstracciones. En cada una, se *mappeó* el problema para que quedaran 6 *tiles*, más el blanco, de forma disjunta entre las abstracciones.

Algoritmos de búsqueda: No se contaron con los recursos necesarios para probar comparativamente los casos de prueba.

Árbol de búsqueda:

| Modalidad | Pruning | Profundidad | Tiempo | ¿Sin memoria? | ¿Usó Swap? |
|-----------|---------|-------------|--------|---------------|------------|
| 24-Puzzle | No | 15 | 20 seg | Sí | No |
| 24-Puzzle | Sí | 20 | 1 min | Sí | No |

Torres de Hanoi

Heurística: Se generó una max PDB, utilizando tres abstracciones. En la primera, se proyectaron todas las variables de estado, exceptuando las últimas ocho. En la segunda, se proyectaron las

últimas ocho variables que se conservaron en la primera abstracción. Y en la tercera, se proyectaron la mitad de los estados del juego.

Algoritmos de búsqueda: No se contaron con los recursos necesarios para probar comparativamente los casos de prueba.

Árbol de búsqueda:

| Modalidad | Pruning | Profundidad | Tiempo | ¿Sin memoria? | ¿Usó Swap? |
|-----------|---------|-------------|--------|---------------|------------|
| 4-12 | No | 10 | 20 seg | Sí | Sí |
| 4-12 | Sí | 14 | 2 min | Sí | Sí |
| 4-14 | No | 10 | 3 min | Sí | Sí |
| 4-14 | Sí | 14 | 5 min | Sí | Sí |
| 4-18 | No | 10 | 20 seg | Sí | Sí |
| 4-18 | Sí | 14 | 50 seg | Sí | Sí |

Nota: Se utilizó el algoritmo BFS para realizar los árboles de búsqueda, razón por la cual en la mayoría de los casos se consumía la memoria del computador antes de poder cumplir el tiempo estipulado. El algoritmo más eficiente para poder realizar esto es el IDDFS, el cual no consume memoria. De igual manera, en el caso del 15 puzzle obtuvimos los mismos resultados que se encuentran en las láminas de clase.

Generadores de Casos de Pruebas:

Todos los generadores se lograron compilar y funcionaron de forma esperada. Para Top Spin se hicieron casos de prueba fáciles, medios y difíciles, mientras que para las Torres de Hanoi se realizaron casos de prueba fáciles y difíciles.

Nota final: En la carpeta *Results* de cada problema, se encuentran las tablas donde se reportan el número de estados a cada profundidad en el árbol de búsqueda a partir del estado *goal*.