Sistemas de Inteligencia Artificial

Métodos de Búsqueda
No Informados e Informados

1er Cuatrimestre 2019

Grupo 08

- Diego Bruno
- Tomás Ferrer
- Matías Heimann
- Marcos Lund

Índice General

Objetivo

Se propuso desarrollar un modelo basado en el juego ‘Fill Zone’ con el fin de implementar métodos de búsqueda tanto informados (Greedy, A*) como no informados (DFS, BFS, IDDFS), de tal manera que se pudiera establecer una comparación en cuanto a rendimiento y efectividad. Asimismo, se buscó poner en práctica diversas heurísticas para profundizar el contraste en lo que respecta a los métodos informados, adentrándose en el concepto de admisibilidad de las mismas y los costos que involucran la expansión de nodos en la búsqueda de una solución óptima.

Descripción del Juego: ‘Fill Zone’

Para comprender el desarrollo del proyecto es necesaria una breve explicación acerca del funcionamiento del juego.

Se comienza con un tablero donde cada celda está ‘pintada’ con un color, y donde el jugador controla la celda superior izquierda del mismo. En cada movimiento, se es capaz de capturar áreas de celdas adyacentes a la zona controlada si coinciden con un color elegido, con el cual se ‘pinta’ dicha zona por completo. El objetivo consiste en controlar todas las celdas del tablero en el menor número de movimientos posibles.

A lo largo del presente informe se utilizará el concepto de ‘isla’ como área del tablero cuyas celdas están conectadas entre sí y que están ‘pintadas’ del mismo color.

El número de colores que se utilizará a lo largo del trabajo es de 6 (seis), aunque podría a futuro utilizarse una cantidad deseada por el mismo usuario sin inconvenientes.

Desarrollo del Proyecto

- *Modelado del Problema*

Como punto de partida se utilizaron el motor de búsqueda y las interfaces provistas por la cátedra, sin realizar ninguna modificación a las mismas.

En la búsqueda de una solución se explora un árbol de estados. El costo de expandir nodos es constante y equivale a 1 (uno), ya que lo que se busca es minimizar la cantidad de turnos necesarios para terminar el juego y no hay diferencia entre aplicar una regla u otra, a excepción de que puede variar la cantidad de turnos necesarios para alcanzar una solución. Cada una de dichas reglas consiste en ‘pintar’ el área de celdas controladas por el jugador de un determinado

color. La única diferencia entre las mismas es que cada una 'pinta' la zona de un color distinto.

Los estados del árbol de búsqueda están compuestos por dos componentes principales: una representación matricial del tablero de juego, y un grafo que representa adyacencias entre islas. Al comenzar el proyecto se implementó sólamente la matriz, ya que se consideró la representación más obvia e intuitiva. Sin embargo, con el fin de acelerar tiempos de búsqueda en la aplicación de heurísticas se decidió incluir un grafo, aunque en realidad sea posible usar sólamente una de estas dos representaciones. Nótese que cada nodo del grafo contiene un Set de celdas que componen a esa isla (con sus respectivas coordenadas).

Finalmente, un estado solución es aquél cuya matriz tiene todas las celdas de un mismo color, o cuyo grafo contiene un solo nodo.

- *Implementación de Métodos de Búsqueda No Informados*

Se implementaron los métodos de búsqueda BFS, DFS y IDDFS. Dado que estos algoritmos fueron vistos en clase no se profundizará el funcionamiento de los mismos. Para su implementación se partió de una cola de prioridad tanto para BFS como DFS, de tal manera que a medida que se vayan explotando los distintos nodos del árbol de búsqueda, se ordenen según lo requieren estos métodos. En el caso de BFS, si un nodo es más profundo que otro estará posicionado en un índice mayor en la cola, mientras que en DFS sucede lo contrario. Entonces, al desencolar nodos de la cola, se asegura que siempre se respeten los algoritmos de búsqueda.

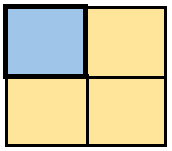
En cuanto a IDDFS, utiliza una máxima profundidad inicial de 15 (quince) para evitar que rápidamente se llegue a un comportamiento similar a BFS. Utiliza una lista y una vez alcanzado el límite de profundidad, se incrementa la profundidad máxima y se parte de los nodos a los que se llegó en la profundidad anterior, por lo que es eficiente.

- *Implementación de Métodos de Búsqueda Informados*

Para implementar métodos de búsqueda informados fue necesario el desarrollo de las estrategias de búsqueda Greedy y A*, así como diversas heurísticas, tanto admisibles como no admisibles. Cabe destacar que el grafo incorporado como estructura subyacente simplificó notablemente las implementaciones de las heurísticas, que inicialmente fueron desarrolladas partiendo de la representación matricial.

La primera heurística fue un resultado directo de entender que una heurística analiza un estado independientemente de los estados que fueron recorridos para llegar al mismo, así como de los estados que puedan surgir en base a él. Entonces, resultó intuitivo implementar una heurística que contara cuántas celdas sin capturar quedan en el tablero (llamada **Unvisited Cells Heuristic**). Si bien esta heurística no

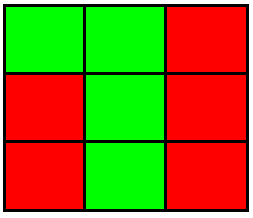
necesariamente lleva a la menor cantidad de pasos para llegar a una solución, distingue bien los diversos estados. No obstante, esta heurística no es admisible. Por ejemplo, dado el siguiente tablero:



la misma retornará 3 (tres), porque faltan tres cuadros por pintar, pero el costo real es 1 (uno) porque basta un solo turno para alcanzar la solución.

Otras ideas, como contar los colores adyacentes a la zona capturada, o contar los colores que aún quedan por capturar, debido a que son triviales son mucho menos específicas y acaban describiendo de la misma forma a tableros que pueden ser casi completamente diferentes.

También se desarrolló otra heurística no admisible, **Isle Count Heuristic**, que cuenta todas las 'islas' que aún no han sido pintadas. No es admisible porque si quedaran solamente dos islas, y éstas fueran del mismo color, el costo sería 1 (uno) y la heurística retornaría 2 (dos), como se observa en el siguiente tablero.



Por último, se implementó una heurística (**Max Distance Heuristic**) que, en base al grafo que representa las islas en el tablero, devuelve la mínima distancia a la que se encuentra la isla más lejana, siendo esta distancia la cantidad de islas que es necesario capturar para alcanzarla. Se garantiza la admisibilidad debido a que no es posible capturar dicha isla en un número menor de turnos que la distancia retornada.

Resultados

Para la obtención de resultados se utilizaron distintos tableros con un número prefijado de islas iniciales, en particular tableros con 5 (cinco), 10 (diez), 15 (quince), 20 (veinte), 25 (veinticinco) y 30 (treinta) islas. Para contar con una muestra más representativa de la que se hubiera obtenido al utilizar simplemente 6 (tableros) de dichas características, se optó por generar 4 (tableros) de cada tipo para luego calcular un promedio de los resultados obtenidos.

En cuanto a los tiempos de ejecución, como se puede observar en la *Figura 1* y la *Figura 6* del Anexo, por lo general se alcanzaron tiempos cercanos a unas

pocas centésimas de segundo y que suelen incrementar monótonamente al aumentar la complejidad del tablero. La heurística Max Distance alcanzó tiempos considerables, muy posiblemente debido al manejo adicional del grafo de islas que debe realizar. La estrategia IDDFS también resultó afectada, sin embargo en menor medida que BFS, la cual llegó a alcanzar tiempos de más de 8 (ocho) segundos. Esto se supone debido a que navega por todas las ramas del árbol de búsqueda hasta encontrar una solución.

En la cantidad de movimientos para alcanzar una solución (profundidad), se observó que la gran mayoría de los casos tuvo un rendimiento similar, así como puede apreciarse en la *Figura 2* y la *Figura 7* del Anexo. Sin embargo, se destaca BFS como la estrategia de búsqueda que menos turnos consumió debido al hecho que recorre los estados horizontalmente por nivel de profundidad, lo cual asegura una solución de profundidad mínima. Por el otro lado, DFS (y en menor medida, IDDFS) encontró soluciones aplicando reglas un mayor número de veces que los demás casos lo cual se debió a que se recorre una sola rama del árbol de búsqueda hasta encontrar una solución sin discriminar entre otros mejores caminos. Se asegura encontrar una solución al ser todas las hojas de dicho árbol estados deseados.

Finalmente, en cuanto a la manipulación de nodos en el árbol de búsqueda se observó una cantidad considerablemente mayor de nodos expandidos, analizados y frontera al utilizar la estrategia BFS. Por el contrario, la heurística Unvisited Cells aplicada tanto en el algoritmo Greedy como en A* resultó ser la que menos nodos manipula.

Conclusiones

Tomando en cuenta los resultados obtenidos en la ejecución del proyecto, se pueden obtener múltiples conclusiones. Por un lado, resultó enriquecedor apreciar la dicotomía “Tiempo de ejecución - Profundidad de exploración” en los métodos de búsqueda no informada, y también resultó notable que no necesariamente sucede lo mismo al utilizar heurísticas. La estrategia de búsqueda BFS llegó a los mejores resultados en cuanto al número de jugadas tomadas para arribar a una solución. Sin embargo, no se debe dejar de lado el hecho de que resultó ser el peor método en cuanto al tiempo requerido para hacerlo, lo cual la hace prácticamente inviable para tableros de muchas celdas e islas. Con DFS, se llegó a un caso opuesto. Resulta deseable un balance entre ambos parámetros de eficiencia en la búsqueda de una solución óptima, lo cual fue en parte conseguido aplicando las heurísticas Unvisited Cells y Isle Count. Para esto, la utilización de búsquedas informadas, como Greedy y A*, es esencial.

Anexo

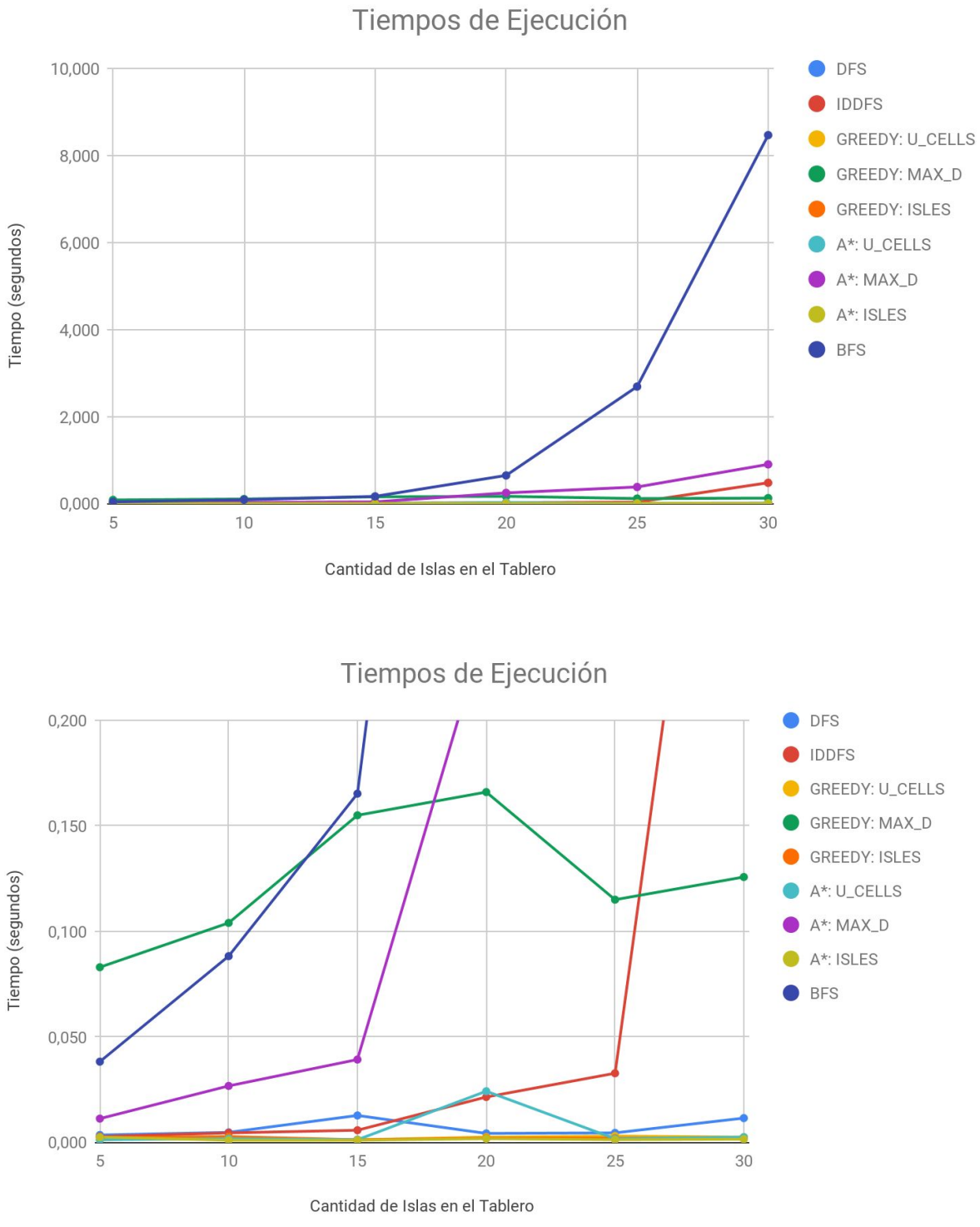


Figura 1: Gráficos de tiempo de ejecución en función de la cantidad de islas inicial.

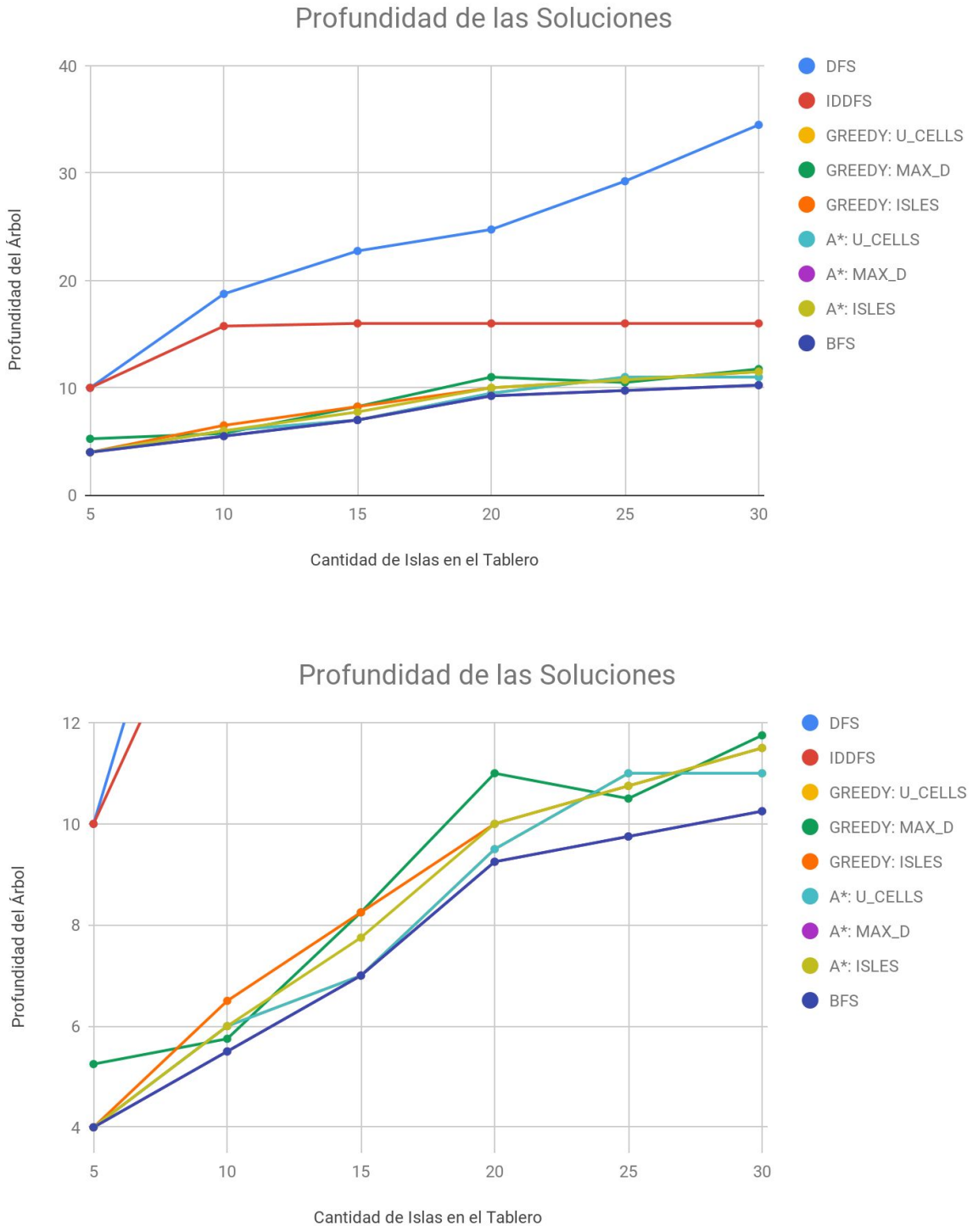


Figura 2: Gráficos de profundidad de la solución en el árbol de búsqueda en función de la cantidad de islas inicial.

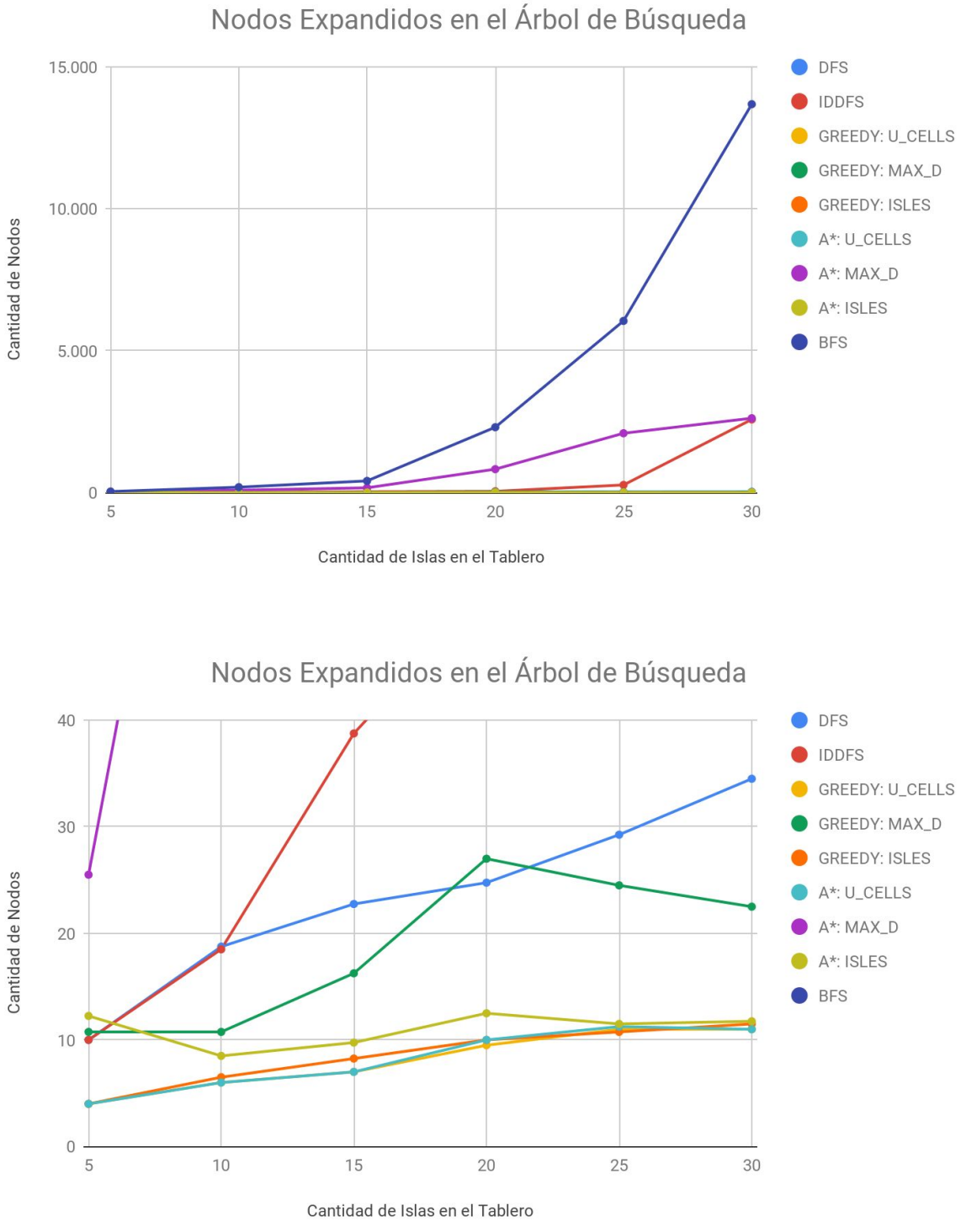


Figura 3: Gráficos de cantidad de nodos expandidos en el árbol de búsqueda en función de la cantidad de islas inicial.

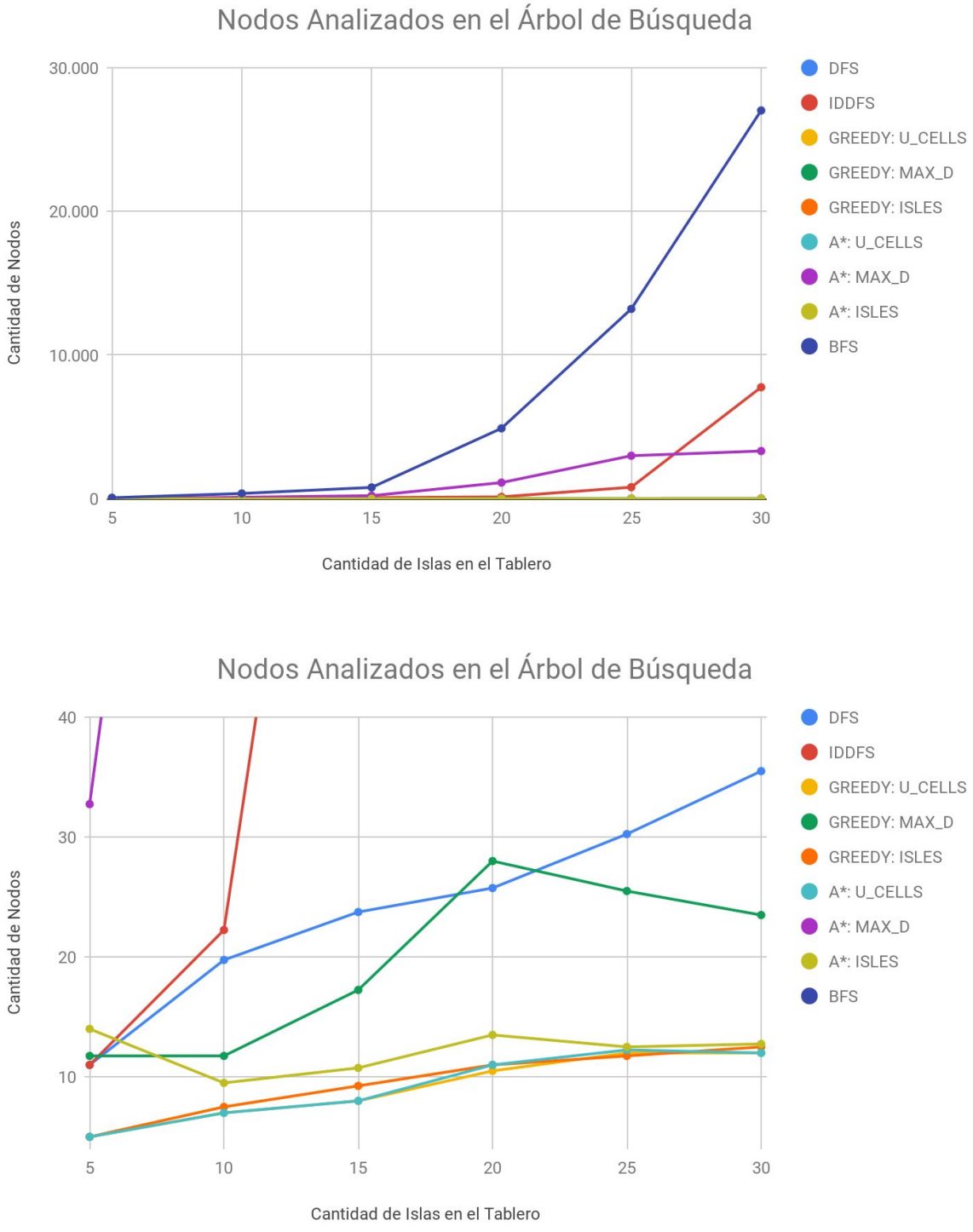


Figura 4: Gráficos de cantidad de nodos analizados en el árbol de búsqueda en función de la cantidad de islas inicial.

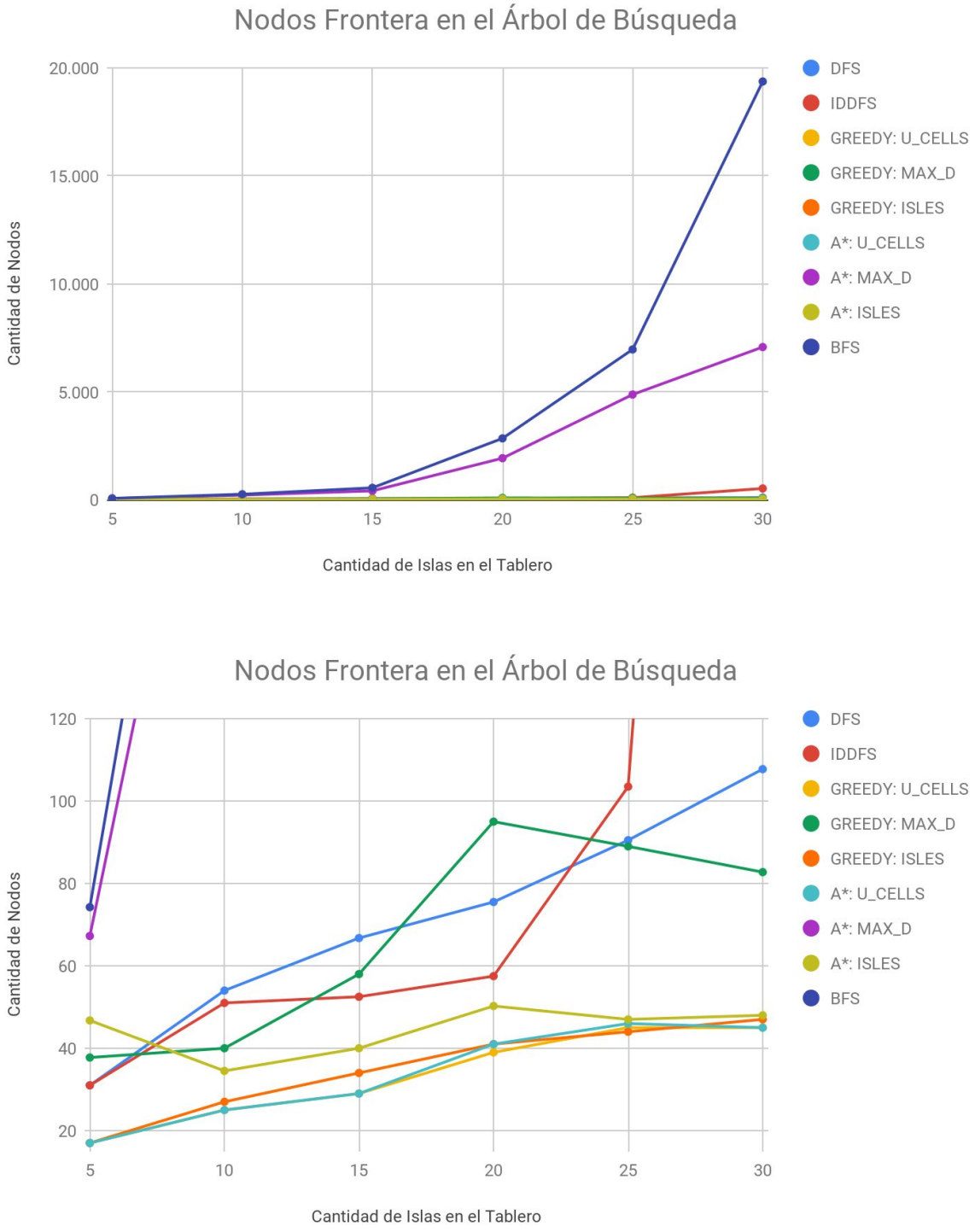


Figura 5: Gráficos de cantidad de nodos frontera en el árbol de búsqueda en función de la cantidad de islas inicial.

	BFS	DFS	IDDFS	GREEDY: U_CELLS	GREEDY: MAX_D	GREEDY: ISLES	A*: U_CELLS	A*: MAX_D	A*: ISLES
5	0,038	0,004	0,003	0,001	0,083	0,002	0,001	0,011	0,003
10	0,088	0,005	0,005	0,002	0,104	0,003	0,002	0,027	0,001
15	0,165	0,013	0,006	0,001	0,155	0,001	0,001	0,039	0,001
20	0,647	0,004	0,022	0,003	0,166	0,002	0,024	0,245	0,002
25	2,689	0,005	0,033	0,003	0,115	0,002	0,002	0,381	0,001
30	8,469	0,012	0,478	0,003	0,126	0,002	0,003	0,901	0,002

Figura 6: Tabla de tiempo de ejecución en función de la cantidad de islas inicial (expresado en promedios).

	BFS	DFS	IDDFS	GREEDY: U_CELLS	GREEDY: MAX_D	GREEDY: ISLES	A*: U_CELLS	A*: MAX_D	A*: ISLES
5	4	10	10	4	5	4	4	4	4
10	6	19	16	6	6	7	6	6	6
15	7	23	16	7	8	8	7	7	8
20	9	25	16	10	11	10	10	9	10
25	10	29	16	11	11	11	11	10	11
30	10	35	16	11	12	12	11	10	12

Figura 7: Tabla de profundidad de la solución en el árbol de búsqueda en función de la cantidad de islas inicial (expresado en promedios).

	BFS	DFS	IDDFS	GREEDY: U_CELLS	GREEDY: MAX_D	GREEDY: ISLES	A*: U_CELLS	A*: MAX_D	A*: ISLES
5	45	10	10	4	11	4	4	26	12
10	198	19	19	6	11	7	6	92	9
15	418	23	39	7	16	8	7	175	10
20	2.310	25	52	10	27	10	10	832	13
25	6.055	29	276	11	25	11	11	2.097	12
30	13.689	35	2.583	11	23	12	11	2.627	12

Figura 8: Tabla de cantidad de nodos expandidos en el árbol de búsqueda en función de la cantidad de islas inicial (expresado en promedios).

	BFS	DFS	IDDFS	GREEDY: U_CELLS	GREEDY: MAX_D	GREEDY: ISLES	A*: U_CELLS	A*: MAX_D	A*: ISLES
5	83	11	11	5	12	5	5	33	14
10	379	20	22	7	12	8	7	117	10
15	799	24	97	8	17	9	8	224	11
20	4.911	26	142	11	28	11	11	1.135	14
25	13.223	30	815	12	26	12	12	3.005	13
30	27.033	36	7.772	12	24	13	12	3.333	13

Figura 9: Tabla de cantidad de nodos analizados en el árbol de búsqueda en función de la cantidad de islas inicial (expresado en promedios).

	BFS	DFS	IDDFS	GREEDY: U_CELLS	GREEDY: MAX_D	GREEDY: ISLES	A*: U_CELLS	A*: MAX_D	A*: ISLES
5	74	31	31	17	38	17	17	67	47
10	265	54	51	25	40	27	25	225	35
15	561	67	53	29	58	34	29	420	40
20	2.851	76	58	39	95	41	41	1.933	50
25	6.969	91	104	45	89	44	46	4.880	47
30	19.373	108	531	45	83	47	45	7.079	48

Figura 10: Tabla de cantidad de nodos frontera en el árbol de búsqueda en función de la cantidad de islas inicial (expresado en promedios).