# 1 Tabla con los Resultados Obtenidos

| Algoritmo | Mapa | Runtime (acumulado) | Tamaño de la ruta | Número de nodos expandidos | Máximo número de nodos en memoria |
|---|---|---|---|---|---|
| BFS | Muy pequeño | 2 | 15 | 96 | 97 |
| | Pequeño | 3 | 34 | 130 | 131 |
| | Mediano | 18 | 110 | 846 | 846 |
| | Grande | 29 | 209 | 4630 | 4633 |
| DFS | Muy pequeño | 1 | 27 | 71 | 71 |
| | Pequeño | 5 | 64 | 81 | 81 |
| | Mediano | 2 | 238 | 402 | 402 |
| | Grande | 20 | 935 | 1213 | 1213 |
| A* | Muy pequeño | 3 | 15 | 40 | 57 |
| | Pequeño | 8 | 34 | 86 | 104 |
| | Mediano | 25 | 110 | 717 | 735 |
| | Grande | 326 (TO) | 209 | 3607 | 3702 |
| IDA* | Muy pequeño | 9 | 15 | 122 | 16 |
| | Pequeño | 19 | 34 | 423 | 35 |
| | Mediano | 445 (TO) | 110 | 484625 | 111 |
| | Grande | | | | |
| RTA* | Muy pequeño | 12 | 53 | 53 | 37 |
| | Pequeño | 9 | 38 | 38 | 36 |
| | Mediano | 712 | 1594 | 1594 | 544 |
| | Grande | 701 | 1567 | 1567 | 513 |

Figura 1: Tabla con los Resultados Obtenidos

# 2 Respuestas a las Preguntas Propuestas

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

**Respuesta:** Analizaremos la parte teórica y práctica de cada uno de ellos a continuación.

*Runtime:* DFS es más eficiente en el caso medio que BFS. Para los valores obtenidos, podemos observar como los tiempos obtenidos suelen ser menores para DFS. Sin embargo, la diferencia no es tampoco significativamente alta.

*Tamaño de la ruta calculada:* sabemos que, teóricamente, DFS no garantiza encontrar el camino óptimo. Por su parte BFS si, puesto que el coste de desplazamiento entre nodos es constante (en nuestro caso 1). Vemos como esto se cumple en los resultados que hemos obtenido, siendo el tamaño de la ruta bastante mayor para DFS. La diferencia se hace sustancial en el último mapa principalmente.

*Número de Nodos expandidos / Consumo Memoria:* en este caso, vemos como DFS es más eficiente pues obtiene en todos los mapas un menor número de nodos expandidos y por consecuente, un menor consumo de memoria.

Teniendo todo esto en cuenta, puesto que hablamos de encontrar el camino óptimo, me decanto por la búsqueda en anchura. Aunque DFS obtenga mejores resultados en cuanto a eficiencia, vemos como la ruta generalmente es cuanto menos óptima. Por ejemplo para el mapa grande la diferencia entre el tamaño de ruta entre DFS y BFS es de ¡¡ 726 !! Además ya hemos visto que las diferencias de tiempo no son excesivas, algo que también me hace decantarme por BFS.

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

**Respuesta:** Al igual que ocurría antes nos encontramos ante un algoritmo que encuentra el camino óptimo como BFS, caso de A*, y el DFS que no garantiza encontrarlo. En cuanto a lo teórico, tenemos que:

- *Complejidad Computacional*: Para A*, esta está íntimamente relacionada con la calidad de la heurística que se utilice en el problema. En el caso peor, con una heurística de pésima calidad, la complejidad será exponencial, mientras que en el caso mejor, con una buena $h'(n)$ el algoritmo se ejecutará en tiempo lineal. Para DFS, en el peor caso, es $O(b^m)$, siendo b el factor de ramificación (número promedio de ramificaciones por nodo) y m la máxima profundidad del espacio de estados.
- *Complejidad en Memoria:* Para DFS es $O(b^d)$, siendo b el factor de ramificación y d la profundidad de la solución menos costosa, pues cada nodo generado permanece en memoria, almacenándose la mayor cantidad de nodos en el nivel meta. En cambio, para A* dado que tiene que almacenar todos los posibles siguientes nodos de cada estado, la cantidad de memoria que requerirá será exponencial con respecto al tamaño del problema.

Ahora pasamos a hablar de los resultados obtenidos. En cuanto a eficiencia son mejores para DFS (exceptuando el tamaño de la ruta claro). Sin embargo, esto no lo podemos tratar como el caso anterior. Esto es porque, la diferencia de tiempo para mapas sumamente grandes es significativa. Podemos ver como de hecho, A* no llega a realizar el último mapa porque excede el tiempo límite.

Es por esto que me decanto en este caso por la búsqueda en profundidad. No garantizará encontrar el camino óptimo como A*, pero nos permite encontrar un camino hasta el objetivo sin llegar a exceder el tiempo límmite. Además en cuanto al resto de características es mucho más eficiente que A*.

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

**Respuesta:** Sabemos que ambos algoritmos encuentran, en nuestro caso, siempre los caminos óptimos. En cuanto al algoritmo A*, podemos observar como la principal ventaja que ofrece es un menor runtime que IDA*. Esto se debe en gran parte, a que el número de nodos que se expanden es significativamente menor. Para el mapa mediano, mientras que IDA* excede el timeout con un tiempo de 445, A* logra encontrar la solución con un runtime acumulado de 25ms. Diferencia notable a tener en cuenta.

Por el contrario, uno de los puntos fuertes que presenta IDA* es que presenta un menor consumo de memoria, llegando a evidenciarse bastante a partir del mapa pequeño / mediano.

Es por esto que el uso de bien un algoritmo u otro depende de las circunstancias que se nos presenten. Es decir:

- Cuando se nos exiga disponer de un algoritmo eficiente en cuanto a velocidad de cálculo, es decir, con un tiempo bajo de runtime a ser posible, nuestra elección debería ser el algoritmo de A*.
- Por otra lado, para problemas que tengan ciertas limitaciones en cuanto al uso de memoria, pese a que tarde más, IDA* debería ser nuestra elección sin lugar a duda.

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

**Respuesta:** En esta pregunta nos encontramos ante una difícil comparación. Por un lado, A* obtiene un camino óptimo siempre pero no llega a ejecutar todos los mapas. Por el otro, nos encontramos que RTA* si consigue llegar al objetivo en todos los mapas, teniendo un menor consumo de memoria y número de nodos expandidos. La desventaja que presenta este último es la gran cantidad de tiempo que necesita.

Entonces, teniendo en cuanto nuestros mapas de evaluación, podemos distinguir dos casos:

- Para mapas similares a los 3 primeros, me decantaría por el algoritmo de A* como más eficiente. Tal y como se puede apreciar en la tabla de resultados, encuentra un camino hacia el objetivo en muchísimo menos tiempo que con RTA*. Además las diferencias en cuanto a consumo de memoria y nodos expandidos, no son muy altas.
- Para mapas grandes, como es el caso del último, mi elección sería RTA*. A diferencia de A* consigue ejecutar el juego sin excederse del tiempo límite. Así mismo, el consumo de memoria y los nodos expandidos son mucho menores que para A*. Luego todo son puntos a favor del algoritmo en tiempo real.