Tecnológico Nacional de México / Instituto Tecnológico de Culiacán.

Ingeniería en Sistemas Computacionales.

Inteligencia Artificial.

Tabla Comparativa de Rendimiento para Puzzle-8

Velázquez Sánchez Jesús Diego.

22170849.

José Mario Rios Félix.

Horario de 18:00 a 19:00 PM.

Culiacán Sinaloa a 02 de marzo de 2026.

TABLA COMPARATIVA DE LOS ALGORITMOS DE BÚSQUEDA PARA EL PUZZLE-8

Antes de evaluar el rendimiento de cada algoritmo, primero hay que definir de qué trata cada uno:

Criterio	Búsqueda en Anchura (BFS)	Búsqueda en Profundidad (DFS)	Búsqueda por Costo Uniforme (UCS)	Búsqueda con Heurística (A∗)
Estrategia de búsqueda	Explora nivel por nivel (nodos hermanos primero).	Explora la rama más profunda antes de retroceder.	Expande el nodo con el menor costo acumulado *g(n).*	Usa costo acumulado + estimación al objetivo *f(n) = g(n) + h(n).*
Estructura de Datos	Cola (**FIFO**).	Pila (**LIFO**).	Cola de prioridad (por *g(n)*).	Cola de prioridad (por *f(n)*).
¿Puede terminar?	**Sí** (si el factor de ramificación es finito).	**A veces** (puede perderse en ramas infinitas).	**Sí** (siempre y cuando se manejen costos mayores a 0)	**Sí** (similar a la Búsqueda por Costo Uniforme).
¿Es Óptimo?	**Sí** (si todos los pasos valen lo mismo).	**No** (encuentra la primera solución que toca, no la mejor).	**Sí** (siempre encuentra el camino de menor costo).	**Sí** (si la heurística es admisible y consistente).
Complejidad de Tiempo	**Exponencial:** Procesa todos los nodos de cada nivel hasta encontrar la solución. $O(b^d)$ *d=depth.*	**Exponencial**: Puede ser muy rápido si la solución está al inicio de una rama, o muy lento si está al final. $O(b^m)$ *m = maximun depth.*	**En función del costo**: Depende del costo de la ruta óptima. $O(b^{1+[C^*/\varepsilon]})$.	**Eficiente**: Depende de la calidad de la heurística. Si *h(n)* es buena, reduce drásticamente los nodos expandidos.
Uso de Memoria	Muy alto (guarda todos los nodos en memoria).	Bajo (solo guarda la ruta actual y hermanos).	Alto (similar a BFS, pero basado en costo).	Moderado/Alto (guarda nodos en la "frontera").

TABLA COMPARATIVA DE RENDIMIENTO DE BÚSQUEDA PARA EL PUZZLE-8

Para probar el rendimiento de cada uno, se decidió probar en tres niveles de dificultad diferentes: Fácil, Intermedio y Difícil. **Estas pruebas se realizaron en el proyecto "8puzzle", disponible en:** https://github.com/Jesus-Velazquez-mx/Repositorio-IA.git

Fácil

Estado Inicial: 12384 765

Estafo Meta: 1238 4765

	Búsqueda en Anchura	Búsqueda en Profundidad	Búsqueda por Costo Uniforme	Búsqueda con Heurística
Nodos expandidos	3	181439	3	2
Tiempo (ms)	0.44 ms	116.39 ms	0.44 ms	0.02 ms
Longitud/Número de movimientos	1	1	1	1

Intermedio

Estado Inicial: 84712635

Estado Meta: 1238 4765

	Búsqueda en Anchura	Búsqueda en Profundidad	Búsqueda por Costo Uniforme	Búsqueda con Heurística
Nodos expandidos	170240	58043	227840	121496
Tiempo (ms)	116.73 ms	42.48 ms	193.40 ms	147.42 ms
Longitud/Número de movimientos	26	50396	26	26

Difícil

Estado Inicial: 5674 8321

Estafo Meta: 1238 4765

	Búsqueda en Anchura	Búsqueda en Profundidad	Búsqueda por Costo Uniforme	Búsqueda con Heurística
Nodos expandidos	181404	23791	240561	236520
Tiempo (ms)	128.69 ms	16.66 ms	191.39 ms	202.34 ms
Longitud/Número de movimientos	30	21188	30	30

CONCLUSIÓN

El análisis de rendimiento sobre los distintos escenarios del Puzzle-8 permite concluir que la selección del algoritmo de búsqueda es necesaria para equilibrar la calidad de la solución con el consumo de recursos. Los algoritmos de **Búsqueda en Anchura (BFS)** y **Búsqueda por Costo Uniforme (UCS)** demostraron ser consistentes en la obtención de rutas óptimas, logrando soluciones de 26 y 30 movimientos en los niveles intermedio y difícil, cada uno. Sin embargo, su naturaleza de búsqueda ciega implica una carga masiva en memoria y tiempo, llegando a expandir más de 240,000 nodos en escenarios complejos debido a que procesan todos los estados posibles.

Por otro lado, la **Búsqueda en Profundidad (DFS)** resultó ser el método menos efectivo para este problema; aunque puede registrar tiempos de ejecución menores en algunos casos al no explorar niveles completos, la longitud de solución es inaceptable, teniendo rutas de hasta 50,396 movimientos en lugar de los 26 requeridos por un camino óptimo. Finalmente, el algoritmo **A*** es la opción más equilibrada gracias al uso de una función heurística que guía el proceso hacia el objetivo. En el nivel intermedio, logró encontrar la solución óptima expandiendo significativamente menos nodos que UCS (121,496 frente a 227,840), lo que confirma que la incorporación de conocimiento del problema con estimaciones de costos al objetivo es fundamental para optimizar la búsqueda.