



Tarea #4

“Tabla de Complejidad”

INTELIGENCIA ARTIFICIAL

18:00 – 19:00 pm

Profesor:

José Mario Ríos Félix

Alumno:

Estrada Valenzuela Jesús Ernesto

7to semestre

ING. En Sistemas Computacionales

23/09/2025

ANÁLISIS COMPARATIVO DE ALGORITMOS DE BÚSQUEDA INTELIGENTE

A continuación, se presenta una tabla detallada que analiza y compara las propiedades fundamentales de cuatro algoritmos de búsqueda implementados: Búsqueda en Anchura (BFS), Búsqueda en Profundidad (DFS), Búsqueda de Costo Uniforme (UCS)

Criterio de Evaluación	Búsqueda en Anchura (BFS)	Búsqueda en Profundidad (DFS)	Búsqueda de Costo Uniforme (UCS)
Tipo de Búsqueda	No informada (ciega)	No informada (ciega)	No informada (ciega)
Estrategia de Expansión	Expande el nodo menos profundo no expandido.	Expande el nodo más profundo no expandido.	Expande el nodo con el menor costo de camino (g(n)).
Estructura de Datos Clave	Cola (Queue - FIFO)	Pila (Stack - LIFO)	Cola de Prioridad (Priority Queue)
Complejidad Temporal	$O(b^d)$	$O(b^m)$	$O(b^{(1+\lceil C^*/\epsilon \rceil)})$
Complejidad Espacial	$O(b^d)$	$O(bm)$	$O(b^{(1+\lceil C^*/\epsilon \rceil)})$
¿Es Completo?	Sí. Siempre encontrará una solución si existe.	No. Puede quedar atrapado en bucles o ramas infinitas.	Sí. Siempre que los costos de paso sean positivos.
¿Es Óptimo?	Sí, pero solo si todos los costos de paso son iguales.	No. Encuentra la primera solución, no necesariamente la mejor.	Sí. Siempre encuentra la solución de menor costo.
Caso de Uso Ideal	Encontrar el camino más corto en grafos sin pesos o con pesos uniformes.	Cuando se sospecha que las soluciones están muy profundas y el espacio de estados es grande (ahorra memoria).	Encontrar el camino de menor costo acumulado cuando los costos varían y no se tiene heurística.
Principal Desventaja	Consume mucha memoria al almacenar todos los nodos en la cola.	Puede encontrar caminos largos y subóptimos. No es completo en grafos con ciclos.	Puede ser ineficiente, explorando en círculos si costos bajos lo alejan del objetivo.

GLOSARIO DE TÉRMINOS Y VARIABLES

b (Factor de Ramificación): Número máximo de sucesores de cualquier nodo.

d (Profundidad): Profundidad de la solución óptima (más superficial).

m (Profundidad Máxima): Profundidad máxima del árbol de búsqueda (puede ser infinita).

C^* (Costo Óptimo): Costo total de la solución óptima.

ϵ (Épsilon): El costo positivo más pequeño de una arista.

$g(n)$: Costo real del camino desde el nodo inicial hasta el nodo n .

$h(n)$: Valor de la función heurística; una estimación del costo desde n hasta el objetivo.

ANÁLISIS DE CRITERIOS CLAVE

Completitud: Un algoritmo es completo si, existiendo al menos una solución, el algoritmo tiene la garantía de encontrarla en un tiempo finito. DFS falla en este aspecto porque podría explorar una rama infinita del árbol de búsqueda sin retroceder nunca, ignorando una solución que se encontraba en una rama vecina.

Optimalidad: Un algoritmo es óptimo si garantiza encontrar la solución con el menor costo de camino entre todas las soluciones existentes. UCS es óptimo porque su naturaleza es expandir siempre el camino más barato hasta el momento, asegurando que cuando llega al objetivo, lo ha hecho por la vía de menor costo posible. A^* hereda esta optimalidad, pero la hace más eficiente. Al usar la heurística ($h(n)$), poda ramas del árbol que, aunque actualmente baratas ($g(n)$), se alejan del objetivo. Esto le permite llegar a la misma solución óptima que UCS, pero generalmente explorando muchos menos nodos.