



Universidad  
Andrés Bello®  
Conectar • Innovar • Liderar

# Introducción a la inteligencia artificial

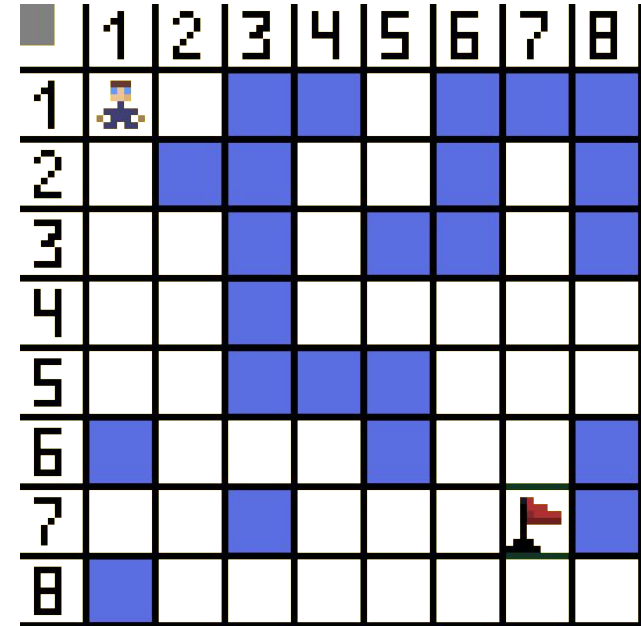
## Fundamentos e Implementación de A\*

### Objetivos

- Entender la función de evaluación  $f(n) = g(n) + h(n)$ .
- Diferenciar  $g(n)$  (costo real acumulado) y  $h(n)$  (heurística).
- Implementar A\* en una grilla 2D con movimientos 4-dir (N,S,E,O).
- Comparar A\* vs BFS/DFS en costo y nodos expandidos.





Un **algoritmo de búsqueda** en Inteligencia Artificial es un **procedimiento sistemático** que, partiendo de un **estado inicial** y conociendo las **acciones posibles** y un **objetivo**, explora el **espacio de estados** generado por el modelo de transición, con el fin de **encontrar una secuencia de acciones (plan)** que lleve desde el estado inicial hasta un estado objetivo, optimizando según una medida de rendimiento (ej: costo, tiempo, pasos).





## Componentes clave

1. **Espacio de estados** → conjunto de todas las configuraciones alcanzables del problema.
2. **Nodo** → representación de un estado dentro del árbol de búsqueda, con información adicional (padre, acción, costo g).
3. **Frontera** → estructura de datos que guarda los nodos por expandir (cola, pila, prioridad).
4. **Expansión** → proceso de aplicar las acciones al estado de un nodo y generar sus sucesores.
5. **Estrategia de selección** → criterio para decidir qué nodo expandir (ej: BFS usa FIFO, DFS usa LIFO, A\* usa  $f=g+h$ ).

	1	2	3	4	5	6	7	8
1								
2								
3								
4								
5								
6								
7								
8								

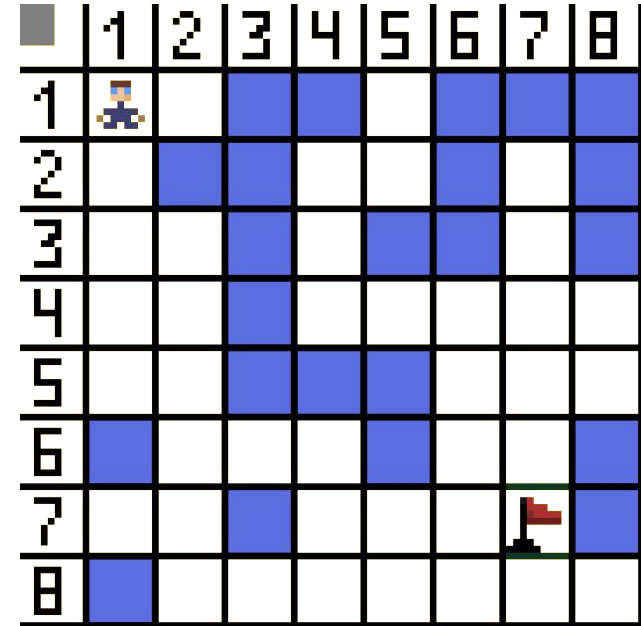
## Clasificación general

- **No informados (ciega)** → no usan conocimiento adicional del objetivo (BFS, DFS, UCS).
- **Informados (con heurística)** → aprovechan estimaciones del objetivo (Greedy, A\*, IDA\*).

	1	2	3	4	5	6	7	8
1								
2								
3								
4								
5								
6								
7								
8								

## Ejemplo sencillo

- **Problema:** encontrar salida en un laberinto.
- **Estados:** posiciones posibles del agente.
- **Acciones:** mover norte/sur/este/oeste.
- **Algoritmo de búsqueda:**
  - BFS expande nivel por nivel hasta hallar la meta.
  - A\* usa una heurística (distancia Manhattan a la salida) para priorizar caminos más prometedores.





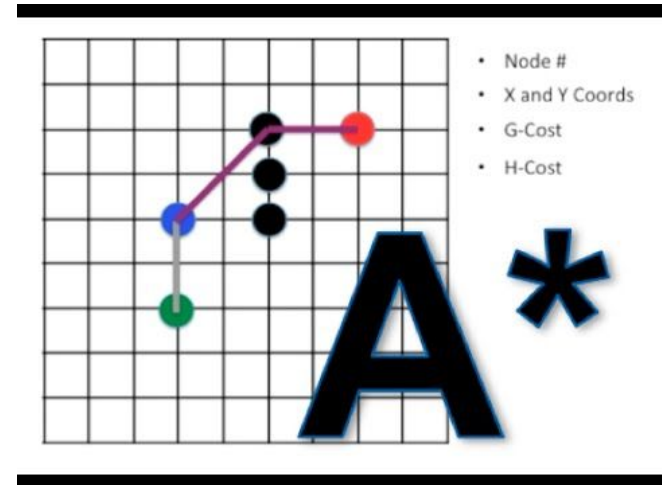
# Búsqueda A\*: minimizar el costo estimado total de la solución

A\* (A estrella) es un algoritmo de búsqueda utilizado para encontrar el camino más corto en un grafo o mapa, desde un punto de partida hasta un destino.

Evalúa los nodos combinando  $g(n)$ , el coste para alcanzar el nodo, y  $h(n)$ , el coste de ir al nodo objetivo:

$$f(n)=g(n)+h(n)$$

- $g(n)$ : el costo real acumulado desde el inicio hasta el nodo actual.
- $h(n)$ : una **estimación** del costo desde ese nodo hasta el objetivo.
- $f(n)$ : costo total estimado si paso por ese nodo.



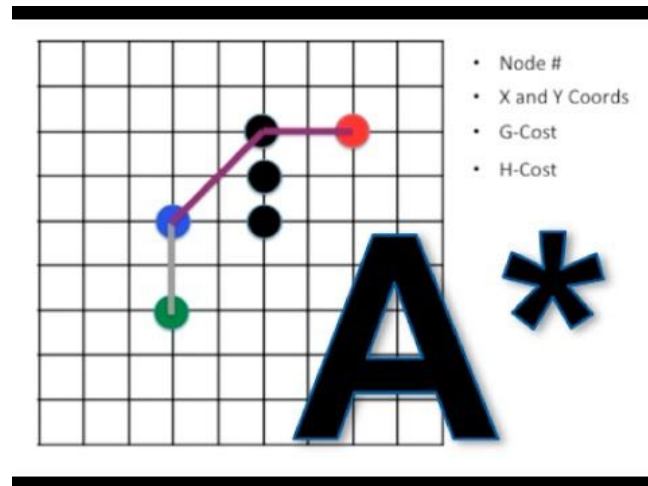
## 📌 ¿Por qué es heurístico?

👉 Porque **no se limita a explorar ciegamente** (como BFS o DFS), sino que **usa conocimiento adicional del problema** para guiar la búsqueda hacia la meta más rápido.

- La función  **$h(n)$**  es la **heurística**, un valor calculado a partir de propiedades del dominio.
- Ejemplo en un mapa: usar la **distancia Manhattan** o **Euclidiana** como estimación de la distancia que falta.
- Ese conocimiento permite a A\* “adivinar” por dónde conviene seguir antes de llegar realmente a la meta.

En otras palabras:




- Si  $h(n)=0 \rightarrow$  A\* se convierte en **Uniform Cost Search**, que es **no informada**.
- Con  $h(n) \neq 0 \rightarrow$  la búsqueda se vuelve **informada** o **heurística**, porque está **guiada por una estimación**.





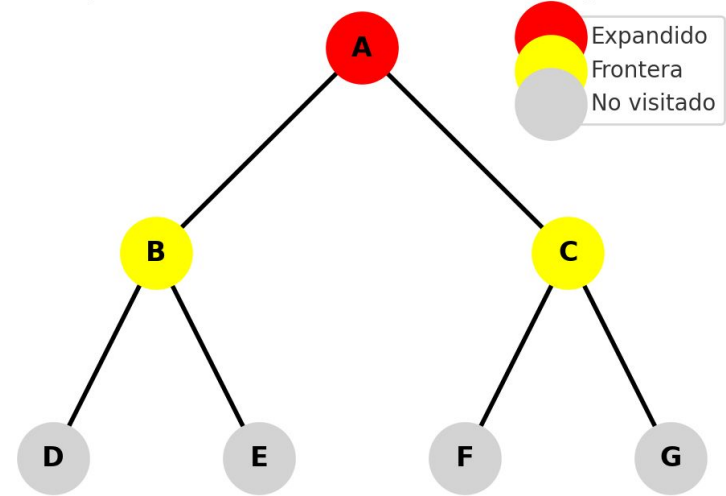
# Búsqueda A\*: minimizar el costo estimado total de la solución

Aquí tienes la **gráfica de la expansión de nodos**:

-  **Nodo expandido (A)** → se sacó de la frontera y se generaron sus hijos.
-  **Nodos frontera (B, C)** → fueron añadidos a la frontera tras la expansión.
-  **Nodos no visitados (D, E, F, G)** → aún no han sido alcanzados.

Esto muestra cómo en la búsqueda se avanza expandiendo un nodo, generando sus sucesores y actualizando la frontera.

Expansión de un nodo en búsqueda



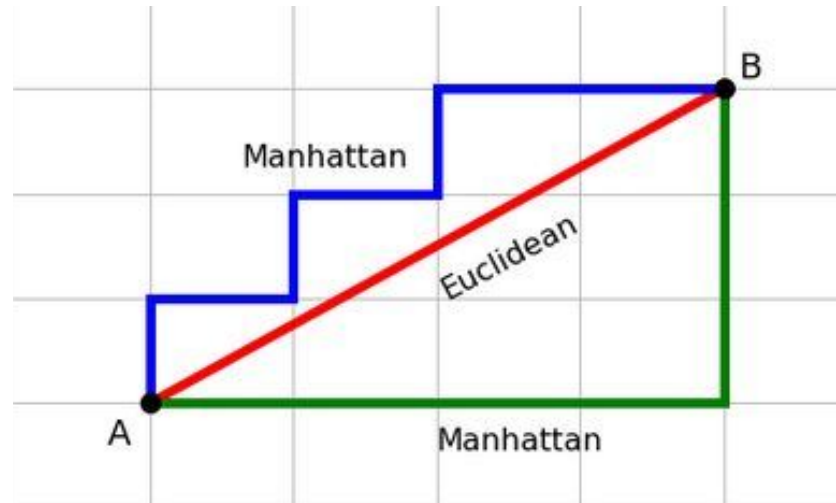
## Ejemplo intuitivo

Cuando A\* decide qué nodo expandir, no lo hace “a ciegas” (como BFS o DFS), sino que prioriza los nodos con menor **f(n)**.

- **g(n)**: lo que ya costó llegar hasta aquí.
  - Es el **costo real acumulado** desde el estado inicial hasta el nodo actual.
  - Ejemplo: si cada movimiento cuesta 1 y hemos hecho 7 movimientos, entonces  $g(n)=7$ .
  - Si el terreno tiene pesos (pasto=1, arena=3),  $g(n)$  refleja esa suma real.
- **h(n)**: lo que se estima que falta.
  - Es la **heurística**: una estimación del costo desde el nodo actual hasta el objetivo.
  - Nunca se ha recorrido todavía: es una “apuesta informada”.
  - Ejemplo: en una grilla 2D, la distancia Manhattan o Euclidiana al objetivo.
- **f(n)**: mezcla los dos.
  - Representa el **costo total estimado** si paso por este nodo: lo ya pagado + lo que falta según mi estimación.
  - A\* expande siempre el nodo con **menor f(n)** en la frontera.
- 

Eso hace que A\* sea **heurístico**: depende de un cálculo basado en conocimiento del entorno, no solo en reglas abstractas de exploración.

Investigue de qué se trata y cómo se calcula la distancia, Manhattan y euclidiana



## Understanding A\* Search Algorithm

### Dijkstra's Algorithm

Provides the actual cost to reach nodes



### Greedy Best-First Search

Uses heuristic for estimated cost

### Heuristic Function

Estimates cost to goal



