

TP 1: Métodos de Búsqueda

Grupo 6

Alberto Abancens (62581)

Kevin Catino (61643)

Agustín Galarza (61481)

Abril Occhipinti (61159)

Maiwenn Boizumault (65988)

Agustin Benvenuto (61448)

Contenidos

01

8 - Puzzle

02

Grid-World

03

Conclusiones



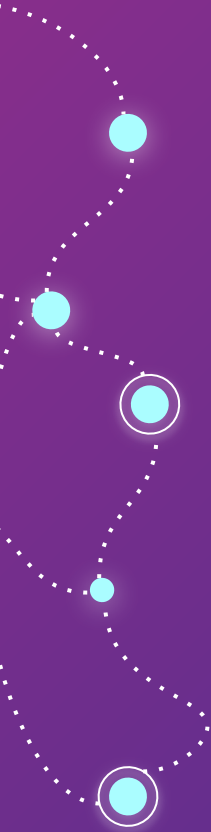


| 01

8 - Puzzle

Estructura de Estado

- Se puede utilizar una matriz de 3x3 para representar el tablero
- Cada posición de la matriz contiene un valor del 1 al 8
- La casilla vacía se representa con un espacio en blanco



5	7	3
8	2	
1	6	4

Estado Inicial



1	2	3
8		4
7	6	5

Estado Final

Acciones

- Mover las fichas adyacentes al espacio en blanco
- Dependiendo de la ubicación del espacio en blanco, la ficha se puede mover hacia arriba, abajo, izquierda o derecha

Objetivo

- Ordenar las fichas de la siguiente manera:

1	2	3
8		4
7	6	5

Heurística 1:

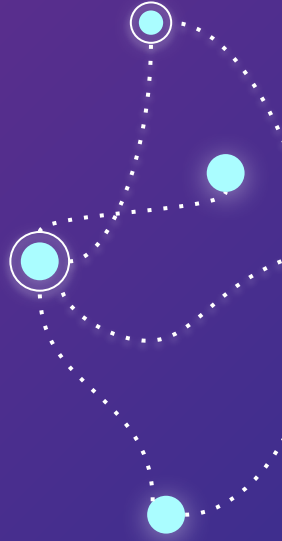
Cantidad de fichas mal ubicadas

$h1(n)$ = cantidad de fichas que no se encuentran en la posición correcta

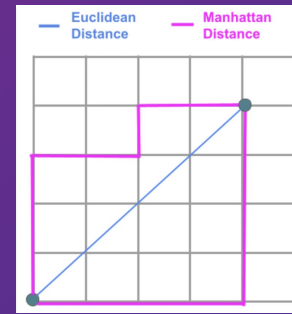
Es **admisable** ya que nunca sobrestima el costo real, siempre se deberán mover al menos $h1(n)$ fichas

5	7	3
8	2	
1	6	4

Fichas mal ubicadas: 5, 7, 4, 2, 1 -> $h1(n) = 5$



Heurística 2: Distancia Manhattan



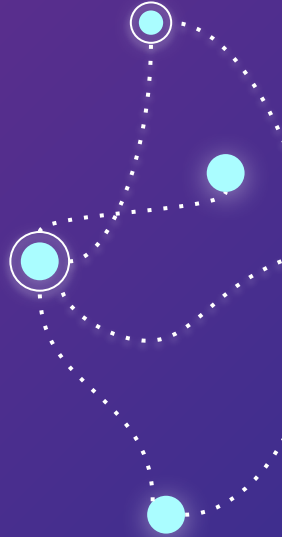
$h_2(n)$ = sumatoria de las distancias verticales y horizontales de cada ficha (1 a 8) desde su posición actual hasta su posición objetivo.

Es **admisible** ya que nunca sobrestima el costo real, en 8-puzzle los movimientos válidos son solo horizontal y vertical.

5	7	3
8	2	
1	6	4

- 5 -> 2 vertical + 2 horizontal = 4
- 7 -> 2 vertical + 1 horizontal = 3
- 4 -> 1 vertical + 0 horizontal = 1
- 1 -> 2 vertical + 0 horizontal = 2
- 2 -> 1 vertical + 0 horizontal = 1

$$h_2(n) = 4 + 3 + 1 + 2 + 1 = 11$$



Métodos de Búsqueda

DESINFORMADOS

No utilizan ninguna información adicional sobre el problema, son poco eficientes para el 8-puzzle donde sí se pueden realizar estimaciones.

Por eso, quedan descartados para este problema

INFORMADOS

Utilizan las heurísticas para evaluar los nodos en el árbol de búsqueda y seleccionar el camino más prometedor hacia la solución.

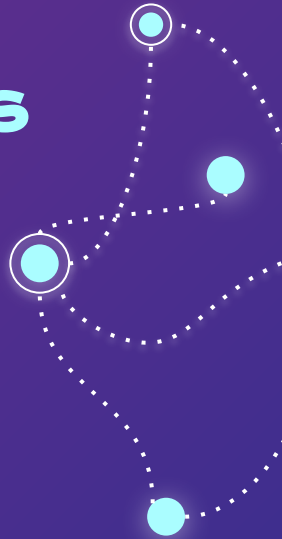
Métodos de búsqueda informados

Greedy Search

- Expande nodo con menor $h(n)$ pero no tiene en cuenta el costo acumulado hasta el momento
- No es óptimo
- No es completo
- Es muy veloz

A* Search

- Expande nodo con menor $f(n) = h(n) + g(n)$
- Si la heurística es admisible y consistente, la búsqueda es óptima

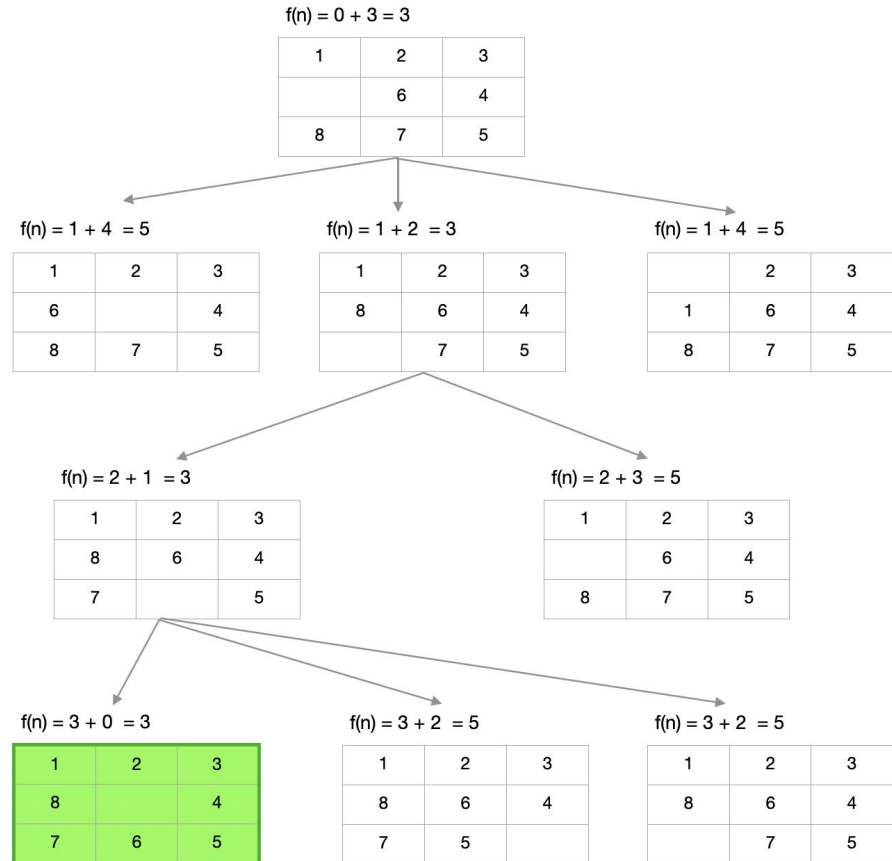


Método de búsqueda elegido: A* con heurística de Distancia Manhattan

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

$g(n)$ = movimientos realizados desde el nodo inicial hasta el nodo n

$h(n)$ = heurística (Distancia Manhattan)



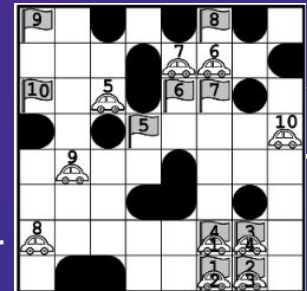


| 02

GridWorld

Reglas del juego

- El objetivo del juego es que un conjunto de agentes llegue a su respectivo “target” moviéndose ortogonalmente en una grilla de $N \times N$.
- La grilla contiene obstaculos “walls”.
- Un agente no puede compartir un espacio en la grilla con otro agente o con un obstaculo.
- Los agentes se mueven por turnos. Pueden elegir moverse arriba, abajo, derecha o izquierda.
- Si en el turno de un agente este no puede moverse a ningún lado o ya se encuentra en su “target” debe pasar el turno.



Heurísticas

Distancia de Manhattan

Se calcula la distancia de Manhattan de todos los agentes hasta su respectivo objetivo y se suma.

Distancia al cuadrado

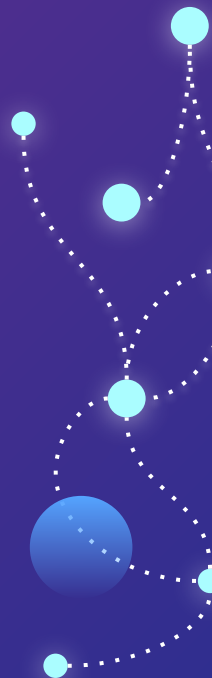
Se calcula la distancia al cuadrado de todos los agentes hasta su respectivo objetivo y se suma.

Distancia Euclídea

Se calcula la distancia Euclídea de todos los agentes hasta su respectivo objetivo y se suma.

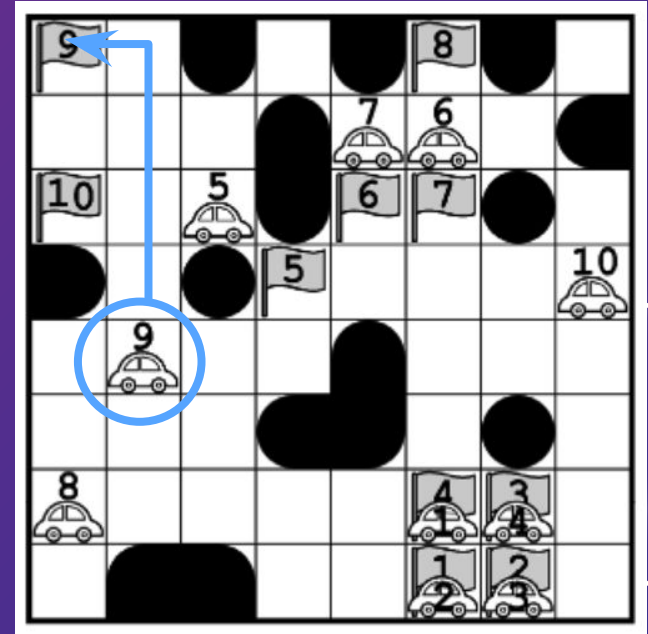
Distancia en el eje x/y

Se calcula la distancia de todos los agentes hasta su respectivo objetivo en el eje x o y únicamente y se suma.



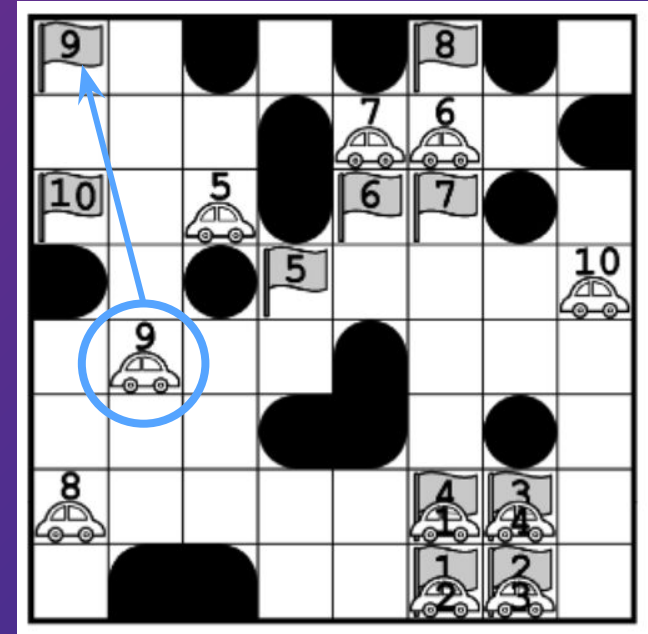
Distancia de Manhattan

- **Es admisible** ya que el costo mínimo para que un agente pueda llegar a su objetivo (cuando no hay obstáculos) siempre equivale a la distancia de Manhattan.
- La presencia de obstáculos en la grilla siempre hacen que el costo final sea igual o mayor.
- Si para cada agente se cumple que la distancia de Manhattan es admisible, para n agentes podemos afirmar que la suma de distancias Manhattan también es admisible.



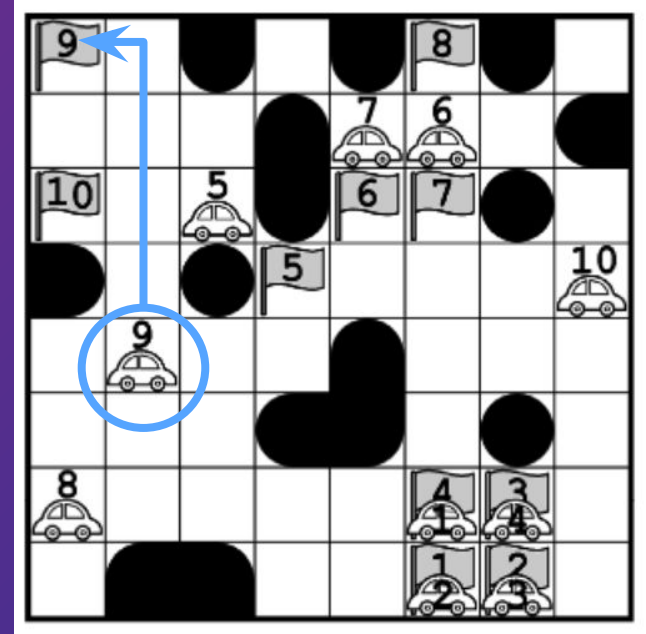
Distancia Euclídea

- **Es admisible** ya que el costo mínimo para que un agente pueda llegar a su objetivo (cuando no hay obstáculos) siempre equivale a la distancia de Manhattan, la cual es **mayor** a la distancia Euclídea.
- La presencia de obstáculos en la grilla siempre hacen que el costo final sea igual o mayor.
- Si para cada agente se cumple que la distancia Euclídea es admisible, para n agentes podemos afirmar que la suma de distancias Euclídeas también es admisible.



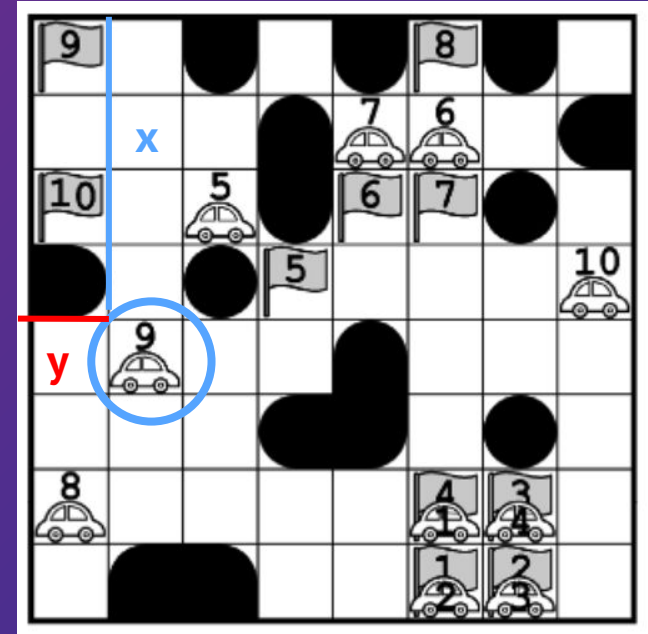
Distancia al cuadrado

- **No es admisible** ya que puede llegar a sobreestimar el costo final:
 - La heurística nos da un valor de $4^2 + 1^2 = 17$
 - Sin embargo, el costo es de 5
- Sabiendo que la heurística puede sobreestimar el costo para un agente, podemos decir que en general no es admisible.



Distancia en el eje x/y

- **Es admisible** ya que el costo mínimo para que un agente pueda llegar a su objetivo (cuando no hay obstáculos) siempre equivale a la distancia de Manhattan, la cual es **mayor** a la proyección de la misma en uno de los ejes
- La presencia de obstáculos en la grilla siempre hacen que el costo final sea igual o mayor.
- Si para cada agente se cumple que la distancia en el eje x/y es admisible, para n agentes podemos afirmar que la suma también es admisible.

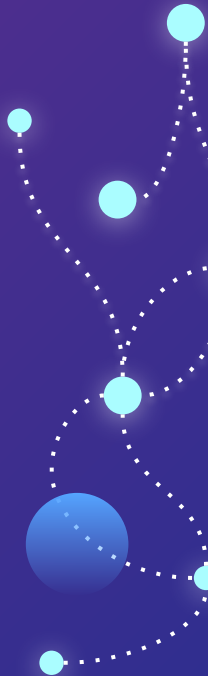


Estructura de Estado

- El estado del juego se representa por un objeto de la clase GridWorld que contiene:
 - Una colección de objetos de la clase Agent donde cada uno contiene un vector posición y un objeto de la clase Target.
 - A su vez, cada objeto de la clase Target contiene un vector que indica su posición dentro de la grilla.
 - La grilla del juego (que es una matriz de $N \times N$ con obstáculos en ciertas posiciones) se genera una única vez y se comparte entre todos los objetos GridWorld que representan un estado.

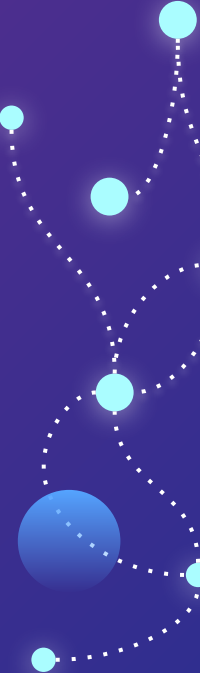
Transición de estados

- A partir de un estado existente contenido en GridWorld, cuando se decide por el próximo movimiento válido de un agente:
 - Se “clona” la estructura, recibiendo un nuevo objeto GridWorld con la misma referencia a la grilla del juego, pero con copias de los agentes y sus vectores posición.
 - La referencia a los objetivos continúa siendo la misma ya que no varía de estado a estado.
 - De esta manera, armar el árbol de búsqueda consiste en clonar una instancia del juego, realizar un movimiento sobre la estructura, y luego almacenar dicho objeto en un nodo.



Definiciones

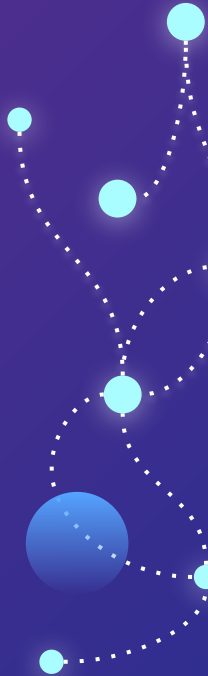
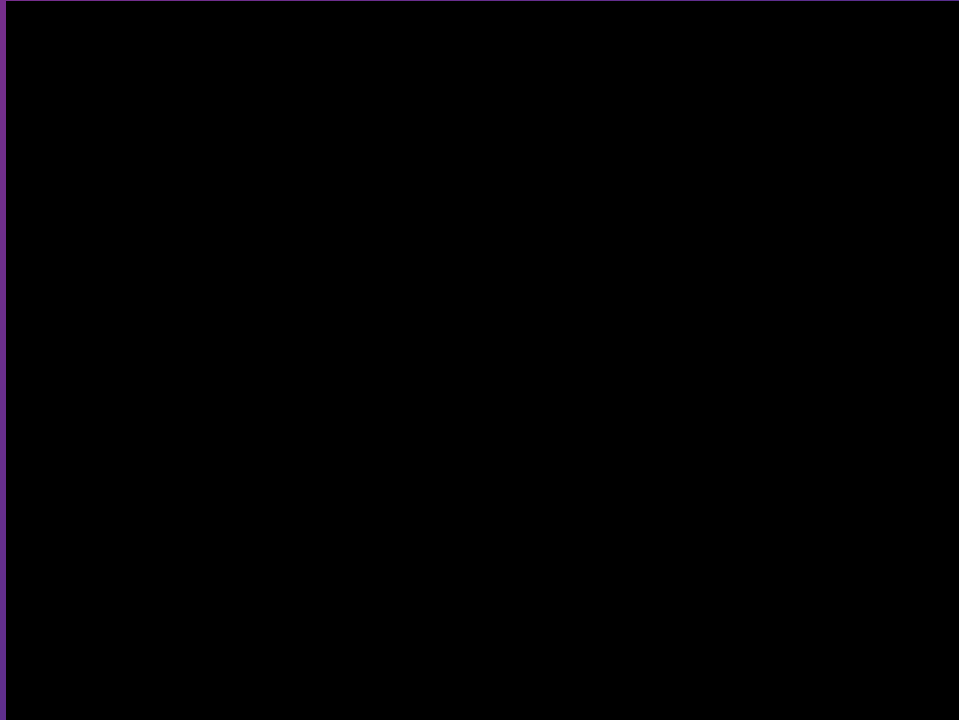
- L: Tamaño de la grilla
- N: Cantidad de agentes
- O: Porcentaje de obstáculos en el mapa



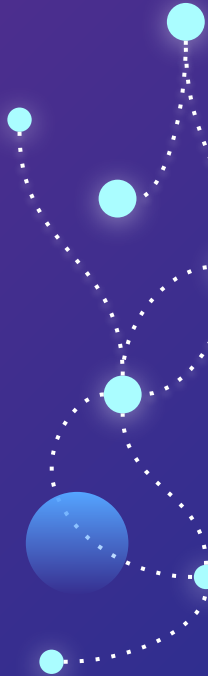
Visualización

Global Greedy Manhattan

- $L = 6$
- $N = 3$
- $O = 0.15$

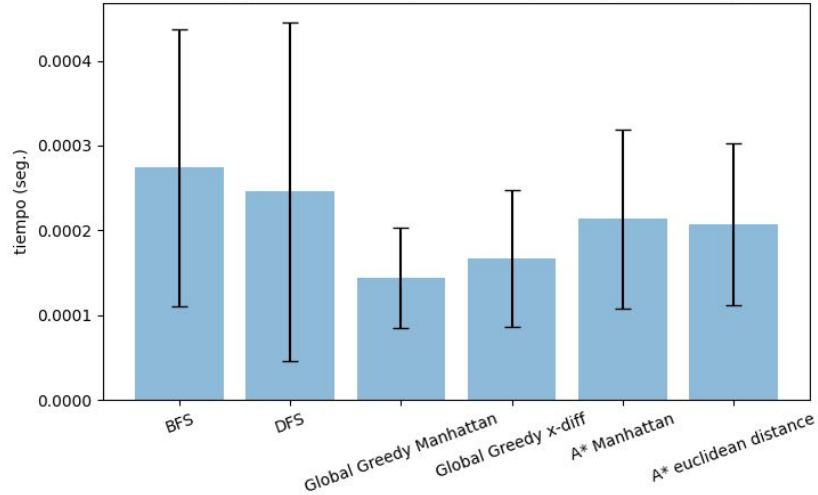


**Resultados (grilla de tamaño 5,
proporción de obstáculos = 0,15)**

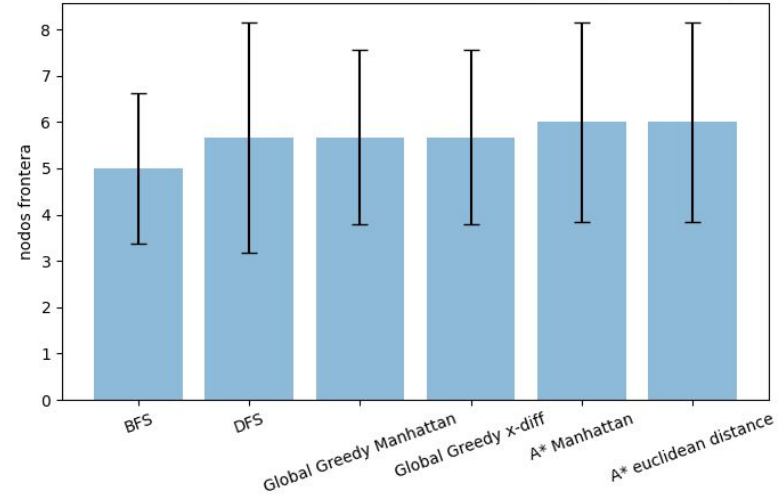


Resultados (L=5, N=1, O=0.15)

Tiempo / método de búsqueda - L = 5, N = 1 (3 iteraciones)

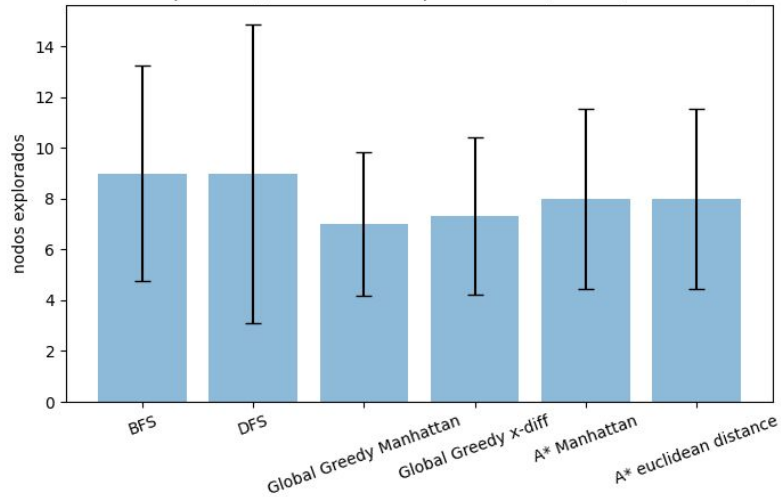


Nodos frontera / método de búsqueda - L = 5, N = 1 (3 iteraciones)

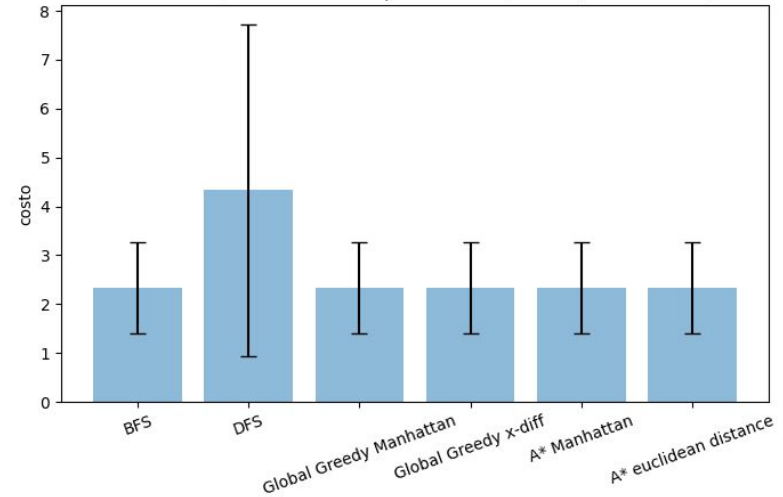


Resultados (L=5, N=1, O=0.15)

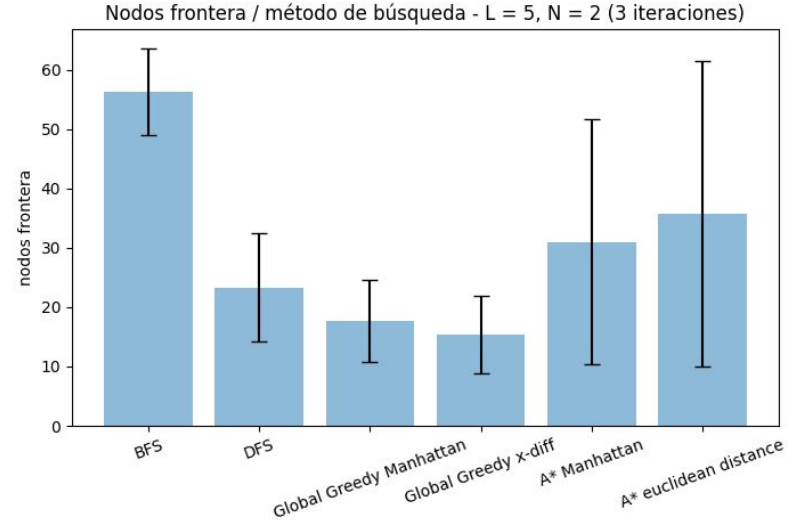
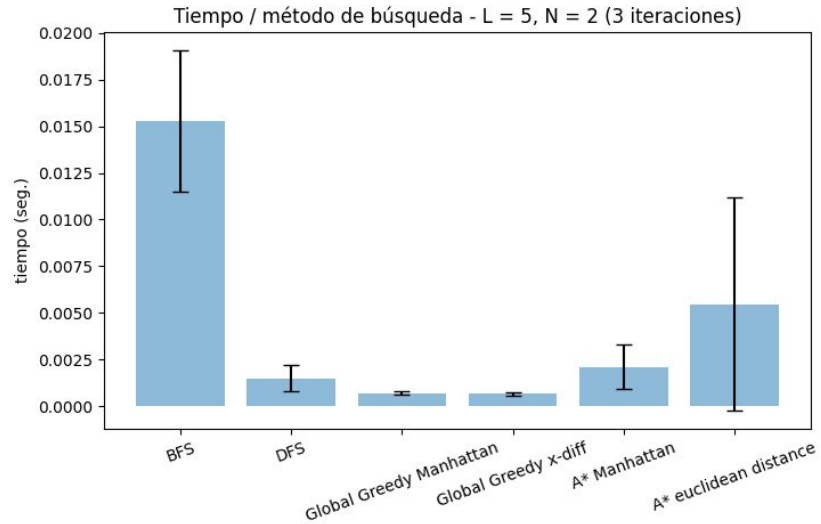
Nodos explorados / método de búsqueda - L = 5, N = 1 (3 iteraciones)



Costo total / método de búsqueda - L = 5, N = 1 (3 iteraciones)

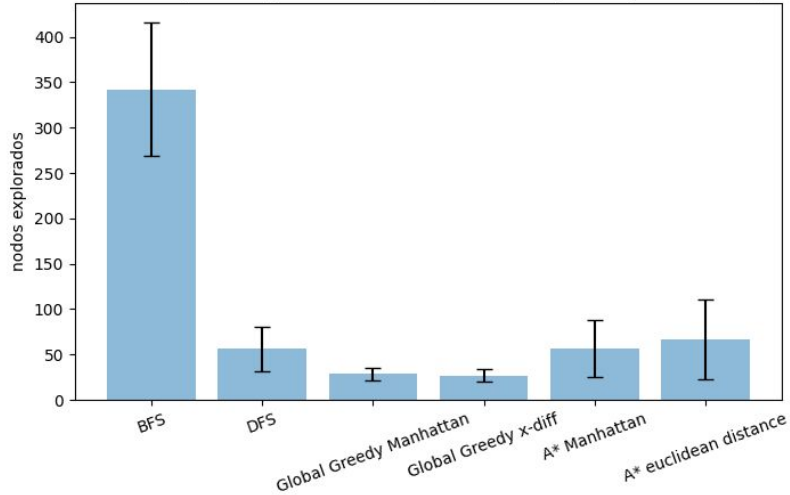


Resultados (L=5, N=2, O=0.15)

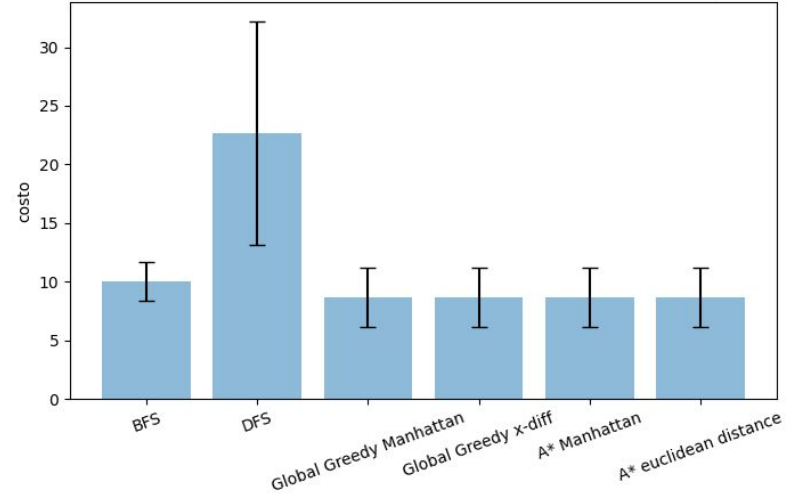


Resultados (L=5, N=2, O=0.15)

Nodos explorados / método de búsqueda - L = 5, N = 2 (3 iteraciones)

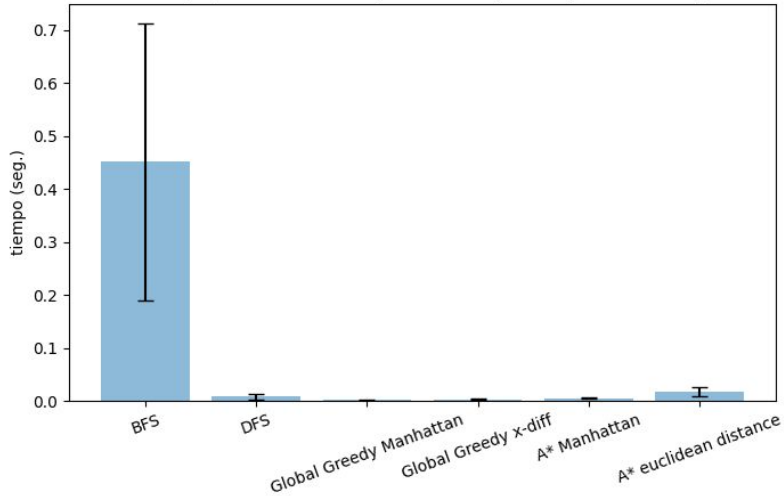


Costo total / método de búsqueda - L = 5, N = 2 (3 iteraciones)

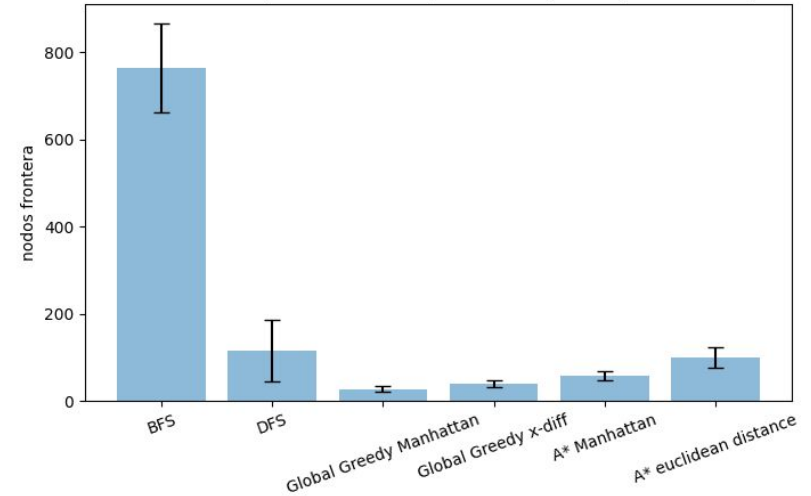


Resultados (L=5, N=3, O=0.15)

Tiempo / método de búsqueda - L = 5, N = 3 (3 iteraciones)

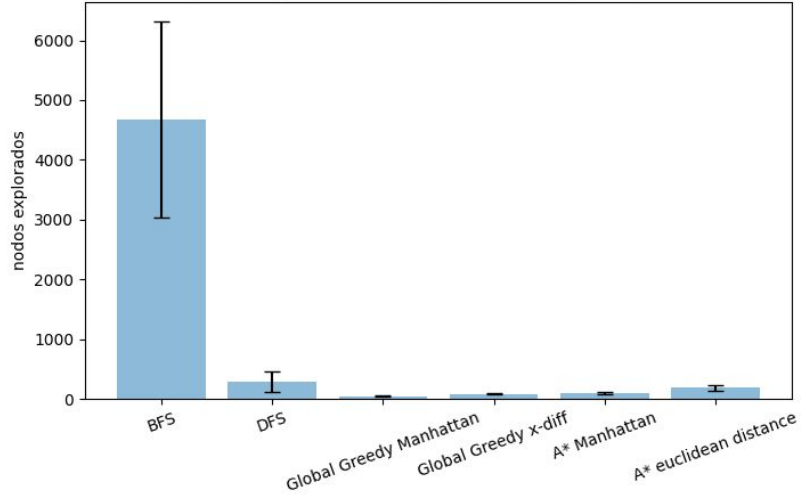


Nodos frontera / método de búsqueda - L = 5, N = 3 (3 iteraciones)

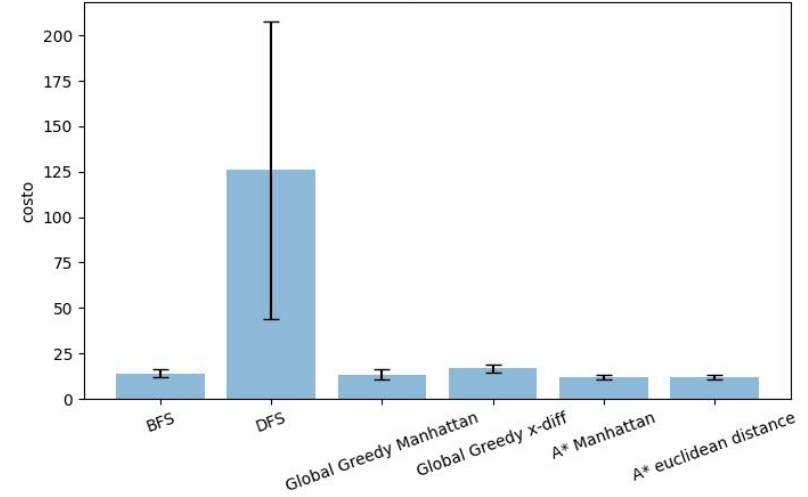


Resultados (L=5, N=3, O=0.15)

Nodos explorados / método de búsqueda - L = 5, N = 3 (3 iteraciones)

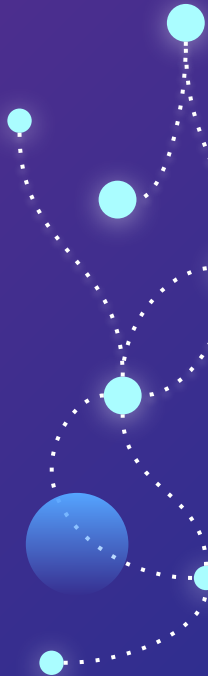


Costo total / método de búsqueda - L = 5, N = 3 (3 iteraciones)



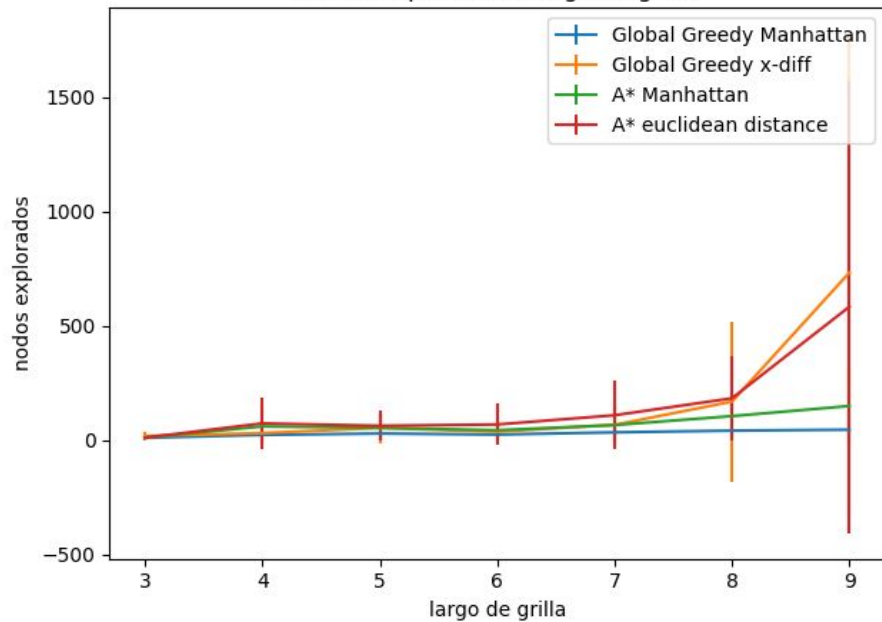
Resultados (métodos informados vs. desinformados)

- Para los siguientes gráficos se tomó un promedio de los siguientes datos:
 - $N = 1, 2, 3$
 - $L = 1, 3, 7, 9$
 - $O = 0.15, 0.20$

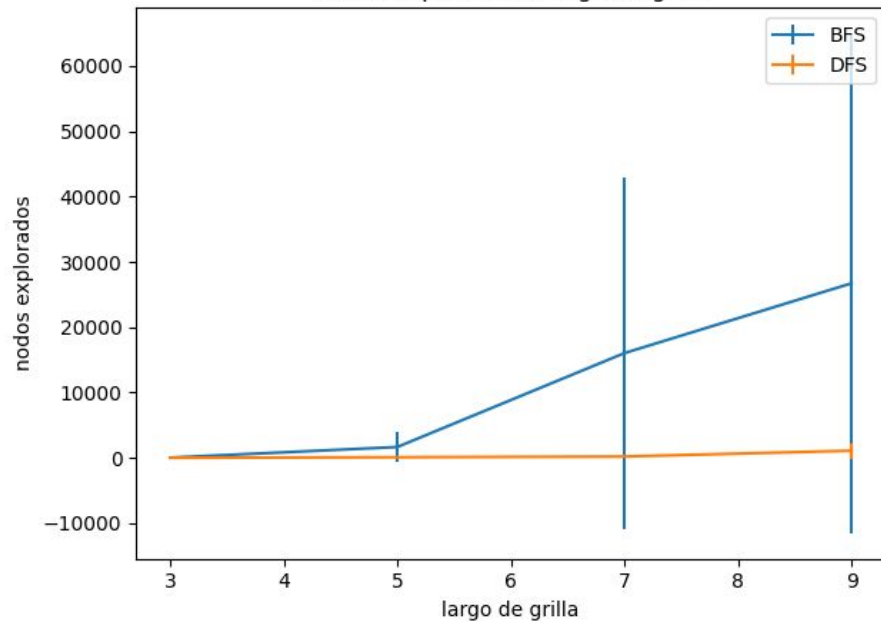


Resultados

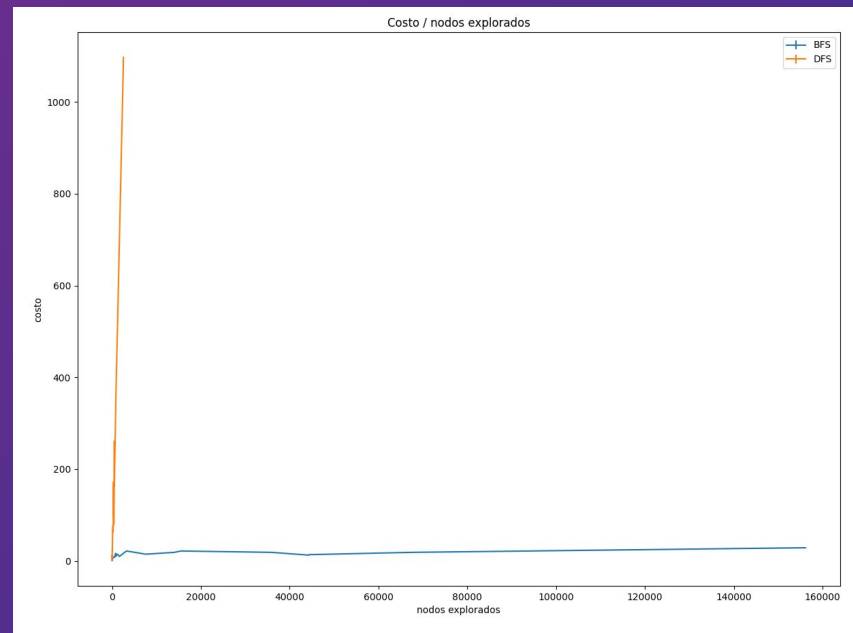
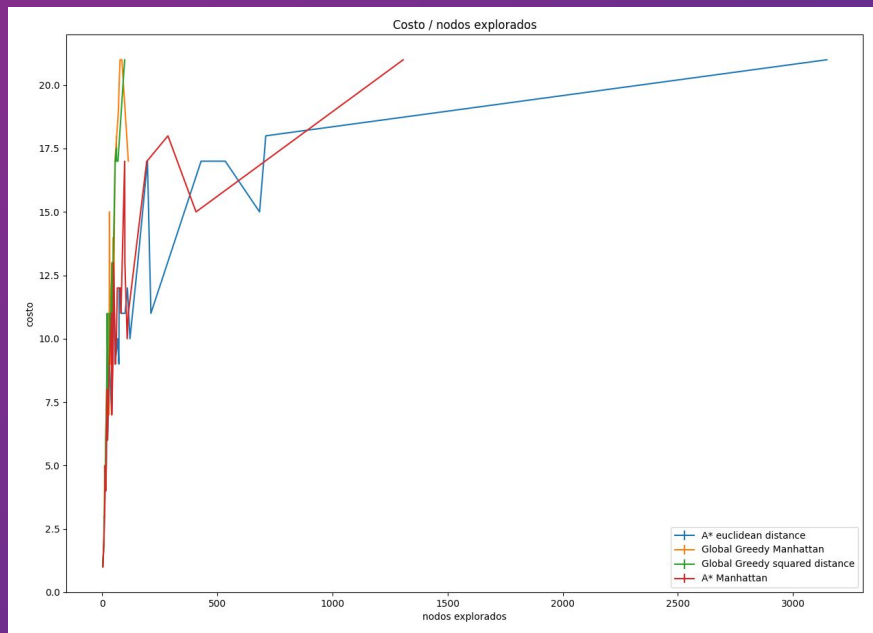
Nodos explorados / largo de grilla



Nodos explorados / largo de grilla



Resultados



Conclusiones

- 1 El uso de las heurísticas elegidas reduce considerablemente el costo total y en varios casos, los nodos explorados
- 2 Si no estamos buscando la solución óptima y queremos velocidad, Global Greedy es más rápido y explora menos nodos que A*
- 3 A partir de los gráficos obtenidos, podemos confirmar que A* siempre llega a la solución óptima (utilizando una heurística admisible)



FIN