

TP1: Métodos de búsqueda

Grupo 2

Tomás Álvarez Escalante (60127)

Alejo Francisco Caeiro (60692)

Lucas Ferreiro (61595)

Román Gómez Kiss (61003)

Tabla de contenidos

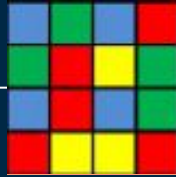


5	7	3
8	2	
1	6	4

01

8-Puzzle

Estructura de estado, métodos de búsqueda y heurísticas



02

Fill Zone

Estructura de estado, métodos de búsqueda y heurísticas



03

Conclusiones

Datos y gráficos de eficiencia y tiempos

01

5	7	3
8	2	0
1	6	4

8-Puzzle

1	2	3
8	0	4
7	6	5

Estructura de estado

Configuración inicial

- Matriz de 3x3 que representa un tablero con casillas.
- Cada casilla contiene un valor entre [0,8].
- La casilla vacía o "empty" se representa con el valor 0
- La casilla a mover o "selected" se representa con un valor distinto de 0.
- Existen 9! posibles estados.



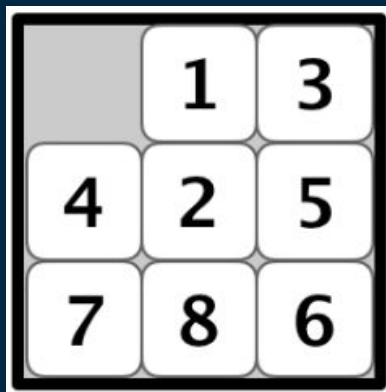
	1	3
4	2	5
7	8	6

Estructura de estado

Acciones válidas

La pieza seleccionada podrá moverse solo si es adyacente a "empty". Luego "selected" tomará el lugar de "empty" y viceversa.

Todas las acciones tienen costo 1.



Estructura de estado

Condición de finalización

El estado actual se considera final cuando las casillas estén ordenadas de la siguiente manera:

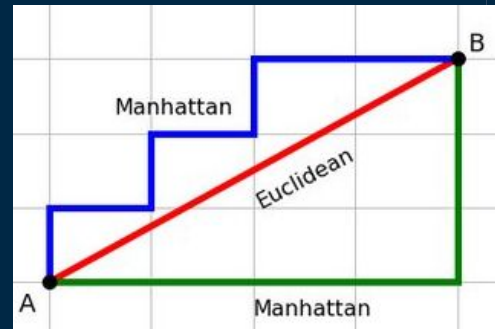
1	2	3
8	0	4
7	6	5

Heurística 1

Distancia de Manhattan

$h(n)$ es la suma de todas las distancias de Manhattan desde una casilla que no esté en su posición correcta (sin contar "empty"). La distancia de Manhattan se calcula sumando la distancia horizontal y la distancia vertical por separado hasta la posición correcta en el estado objetivo.

Podemos asegurar que $h(n) < \text{costo de la solución}$ porque en 8-puzzle los movimientos válidos son solo horizontal y vertical, al igual que con Manhattan. Por lo tanto es **admisible**



Heurística 1

Ejemplo

0	2	3
8	1	6
7	4	5

Las casillas con el valor 1, 4 y 6 se encuentran en una posición incorrecta. Sabiendo que $Man(1)=2$, $Man(6)=2$, $Man(4)=2$, entonces $h(n)=2+2+2=6$.

Usando movimientos válidos podemos ver que necesitaremos obligatoriamente al menos 6 movimientos.

Heurística 2

Cantidad de casillas mal posicionadas

La función heurística $h(n)$ devuelve la cantidad de casilleros que NO se encuentran en la posición correcta.

Podemos asegurar que $h(n) < \text{costo de la solución}$ porque siempre se deberá mover al menos $h(n)$ casillas. Por lo tanto es **admisibile**.

Heurística 2

Ejemplo

5	7	3
8	2	
1	6	4

Las casillas con el valor 3, 6 y 8 se encuentran en su posición correcta, entonces $h(n) = 8 - 3 = 5$.

Usando movimientos válidos podemos ver que necesitaremos realizar mínimamente 5 movimientos.

Métodos de búsqueda

Conocemos el estado objetivo y
planteamos heurísticas admisibles



Métodos de búsqueda informados

Greedy Search



Elegimos el nodo con menor $h(n)$

A* Search



Elegimos el nodo con menor $f(n)=h(n)+g(n)$

Método elegido

Como $h(n)$ es admisible, consistente, costo > 0 y
nodos sucesores finitos



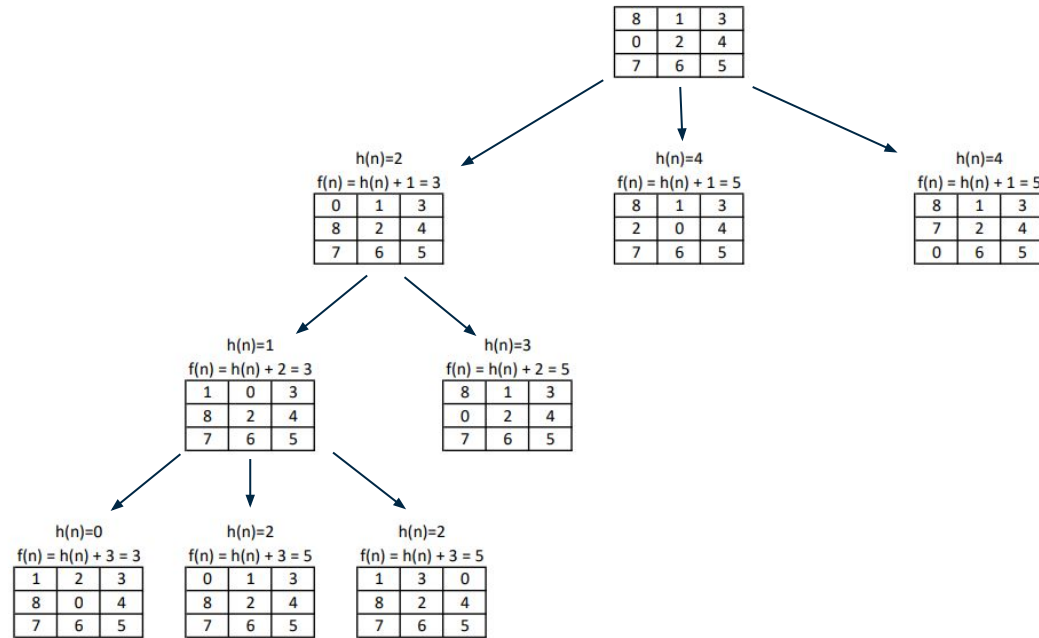
**A* con la heurística
"Distancia de Manhattan"**



Solución óptima y completa

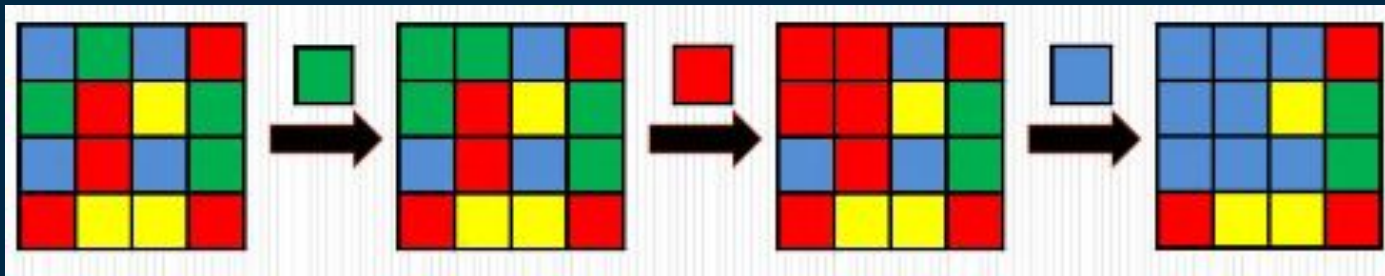
Además A* es eficiente en complejidad temporal y espacial si
se usa una Priority Queue para almacenar los nodos.

Ejemplo del método elegido



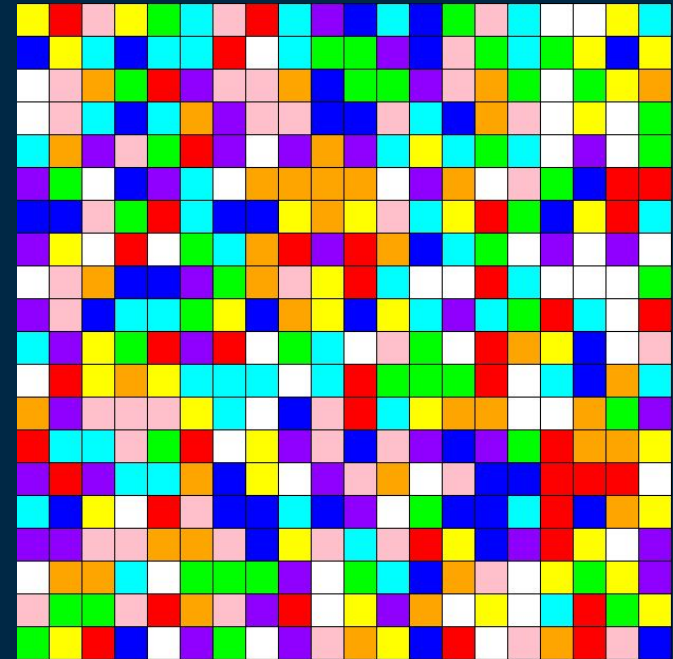
02

Fill Zone



¿Es Fill-Zone un problema bien definido?

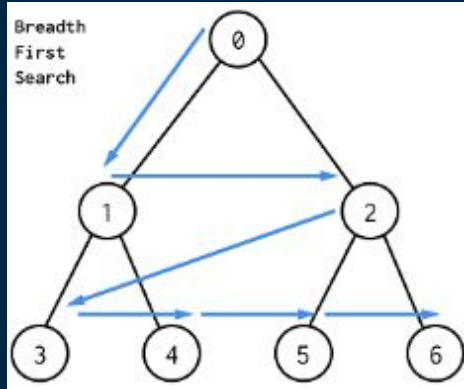
- **Estado inicial:** matriz de $N \times N$ aleatoria y conocida con M colores.
- **Conjunto de posibles acciones:** cambiar a cualquier color disponible en el tablero.
- **Modelo de transición:** elegir un color para unir las casillas adyacentes del mismo color.
- **Función de costo:** siempre 1, pues no hay acciones más costosas que otras.
- **Condición de solución:** matriz completa, es decir, todas las casillas de un mismo color.



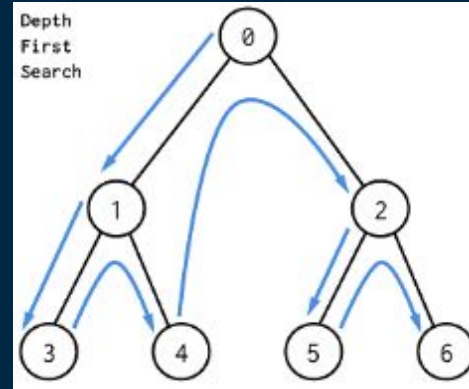
20x20 y 9 colores

Métodos de búsqueda desinformados

BFS (Breadth First Search)

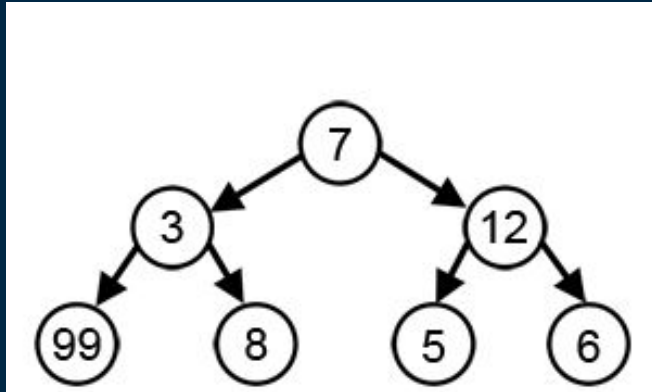


DFS (Depth First Search)

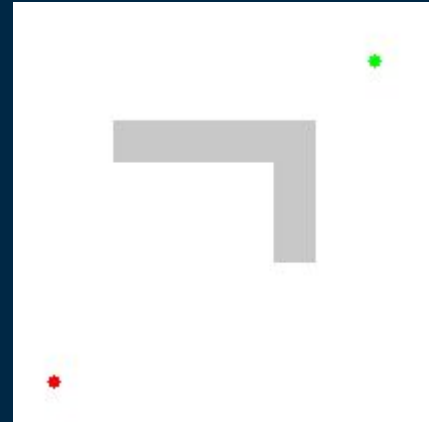


Métodos de búsqueda informados

Local Greedy Search



A*



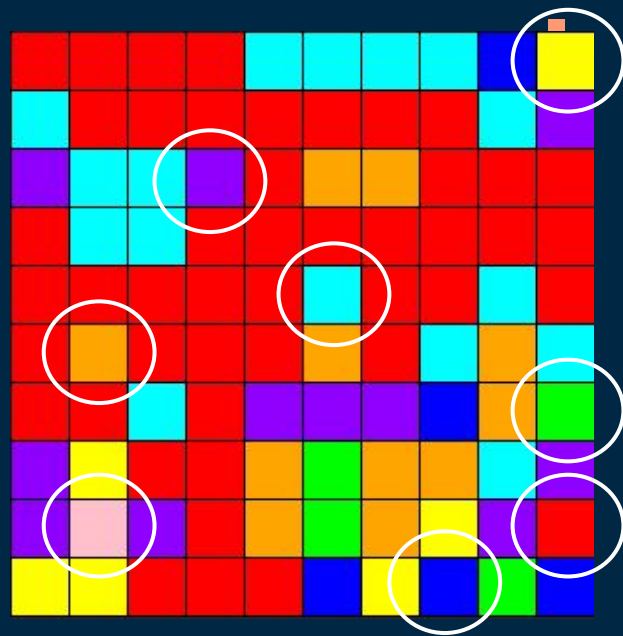
Heurística 1

Colores restantes

$h(n)$ se calcula en base a la cantidad de colores diferentes no controlados por el jugador.

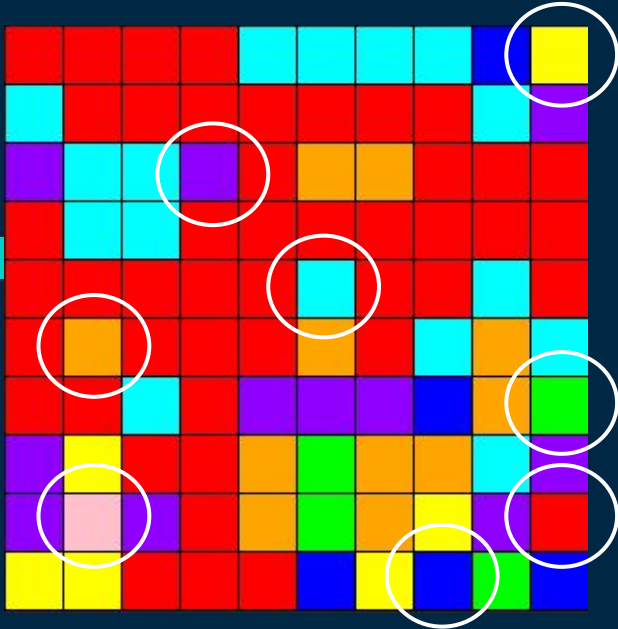
Podemos asegurar que $h(n) < \text{costo de la solución}$ porque se debe cambiar al menos una vez por cada color no controlado para completar el tablero.

Por lo tanto, $h(n)$ es **admisible**.



Heurística 1

Ejemplo

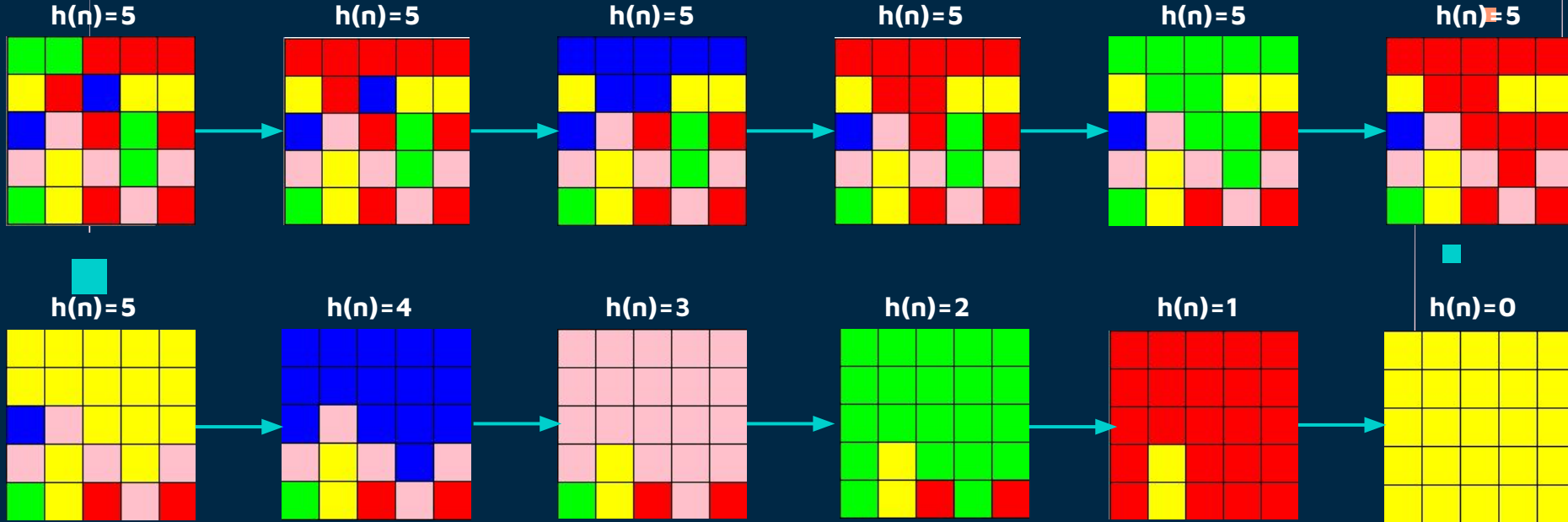


$h(n)=8$, pues son 8 los colores no controlados por el jugador.

Es trivial ver que se necesitan al menos 8 cambios de color para completar el juego.

Heurística 1

Ejemplo de seguimiento para un tablero de 5x5 con 5 colores



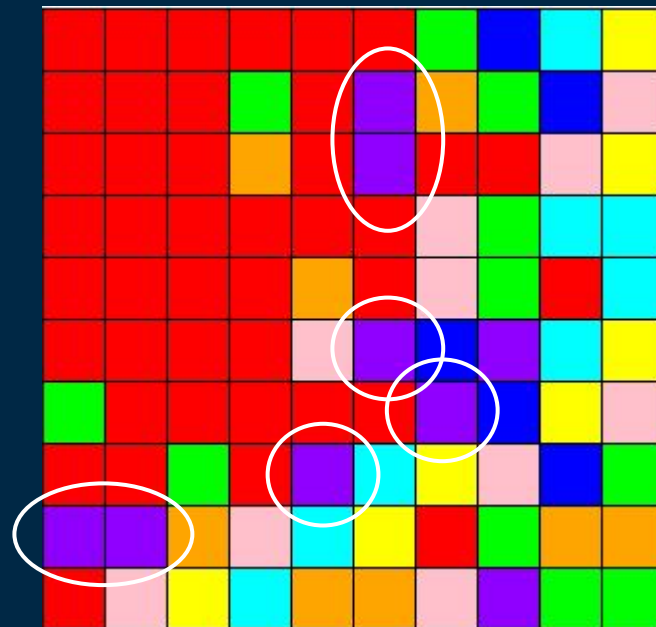
Costo real = 11

Heurística 2

Cantidad de celdas restantes

$h(n)$ se calcula en base al movimiento que permite reducir la mayor cantidad de celdas no controladas por el jugador para completar el tablero.

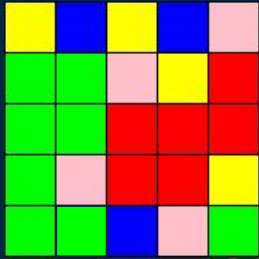
Como veremos en el ejemplo siguiente, $h(n)$ **no es admisible**, ya que sobreestima el costo real de la solución y no permite estimar la cantidad de pasos restantes.



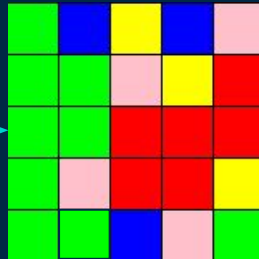
Heurística 2

Ejemplo de seguimiento para un tablero de 5x5 con 5 colores

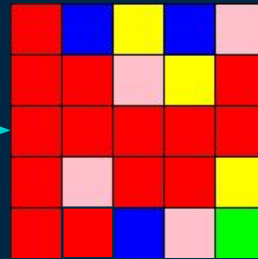
$h(n)=24$



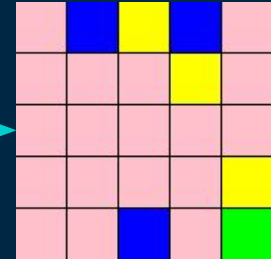
$h(n)=17$



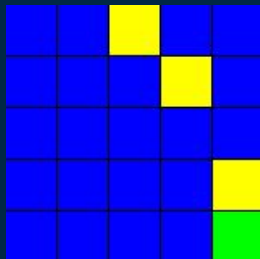
$h(n)=11$



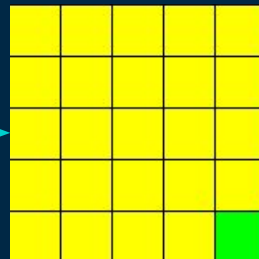
$h(n)=7$



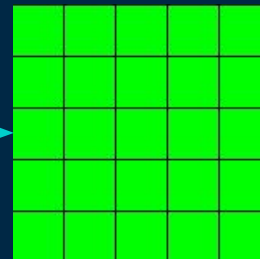
$h(n)=4$



$h(n)=1$



$h(n)=0$



Costo real = 6

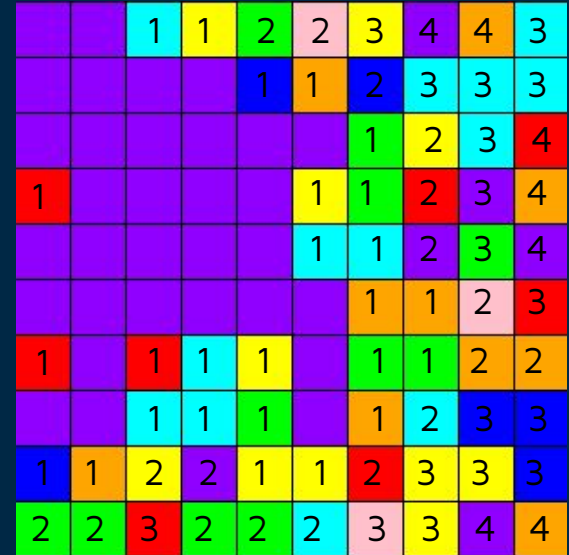
Heurística 3

Distancia de Dijkstra

$h(e)$ se calcula en base a la distancia Dijkstra para llegar a la celda más alejada del “bloque principal” controlado por el jugador.

Podemos asegurar que $h(n) < \text{costo de la solución}$ porque se deben realizar al menos $h(n)$ pasos para alcanzar la celda más alejada y completar el tablero.

Por lo tanto $h(e)$ es **admisible**.



Heurística 3

Ejemplo

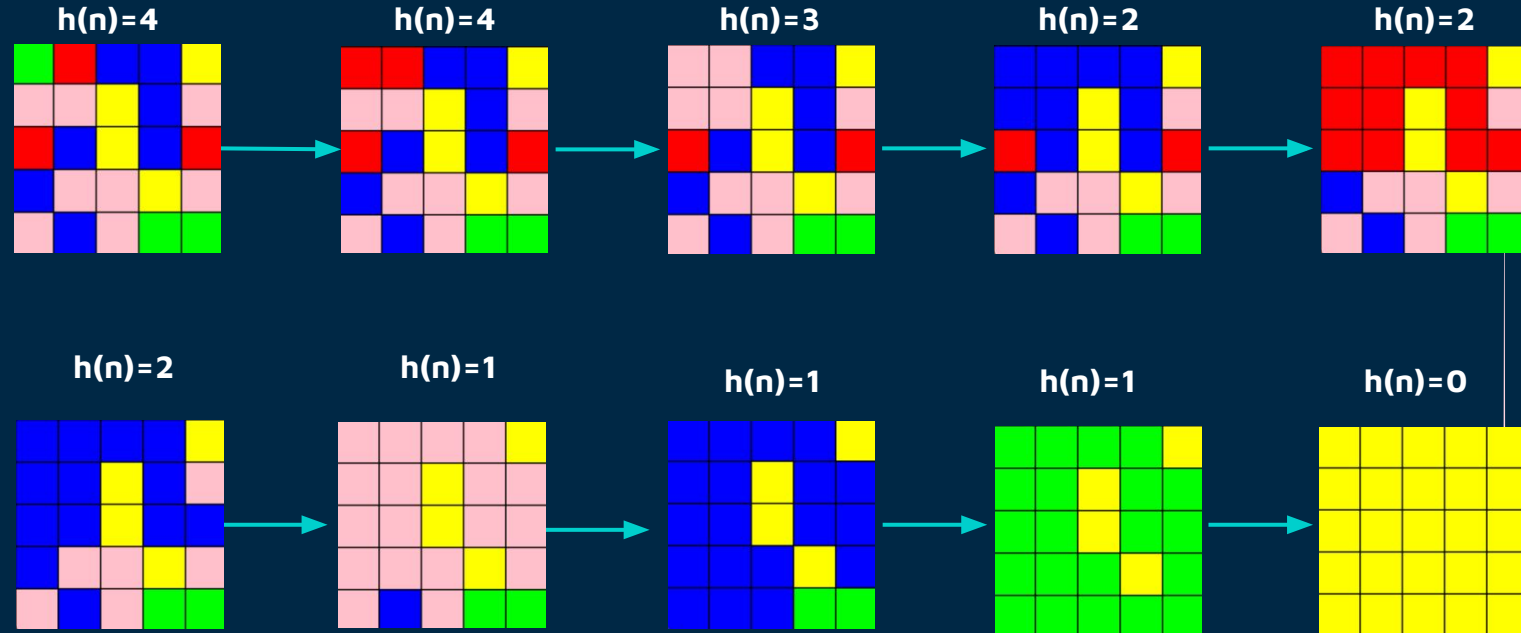
		1	1	2	2	3	4	4	3
				1	1	2	3	3	3
						1	2	3	4
1					1	1	2	3	4
					1	1	2	3	4
						1	1	2	3
1		1	1	1		1	1	2	2
		1	1	1		1	2	3	3
1	1	2	2	1	1	2	3	3	3
2	2	3	2	2	2	3	3	4	4

El valor de cada casilla corresponde a la mínima distancia desde el color predominante hasta dicha casilla. Por ejemplo para el color naranja en la posición de abajo a la derecha bastaría hacer: O -> A -> B -> O.

Por lo tanto en este nodo $h(n)$ sería 4. Pues es la mínima distancia a la casilla más alejada.

Heurística 3

Ejemplo de seguimiento para un tablero de 5x5 con 5 colores




Costo real = 9



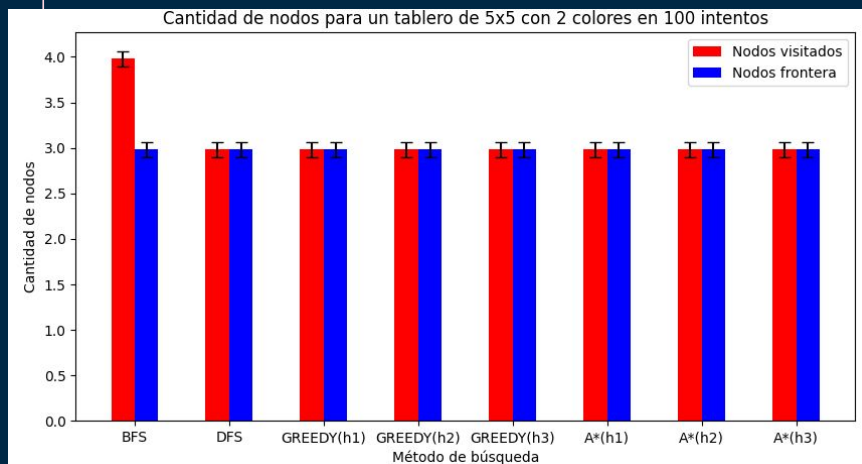
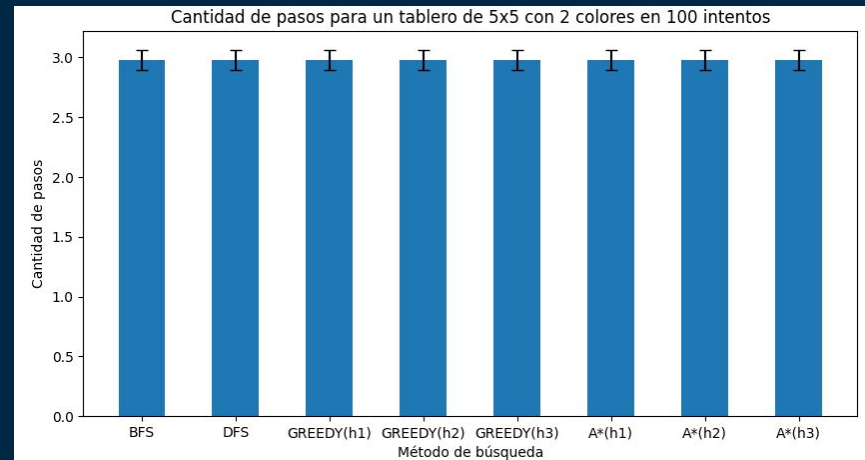
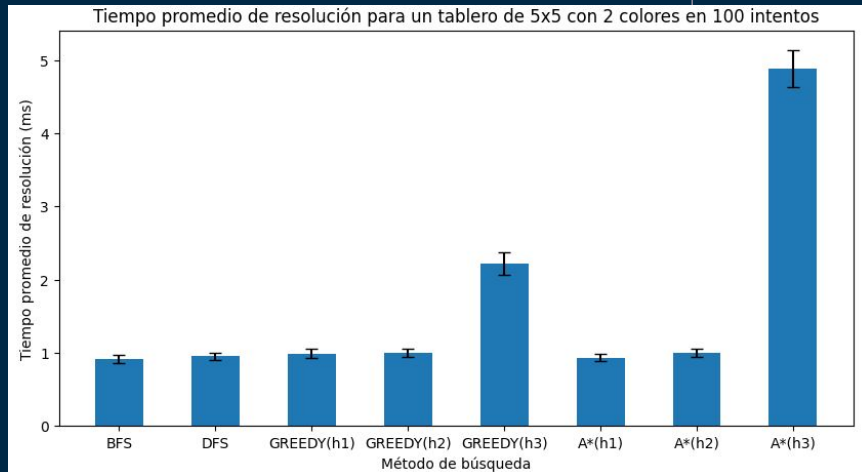
03

Conclusiones

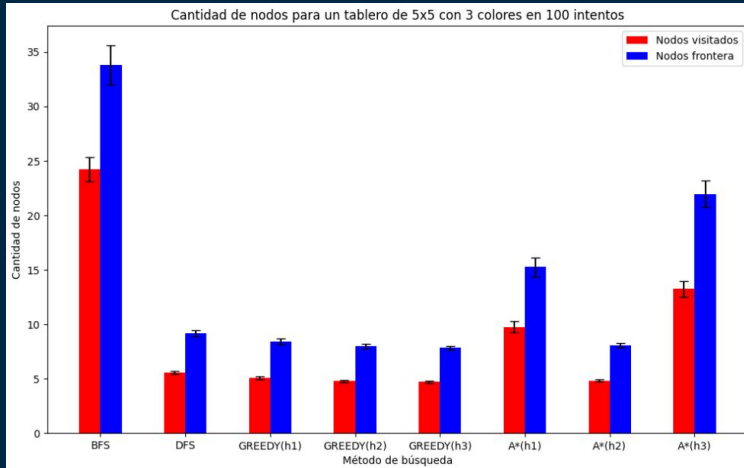
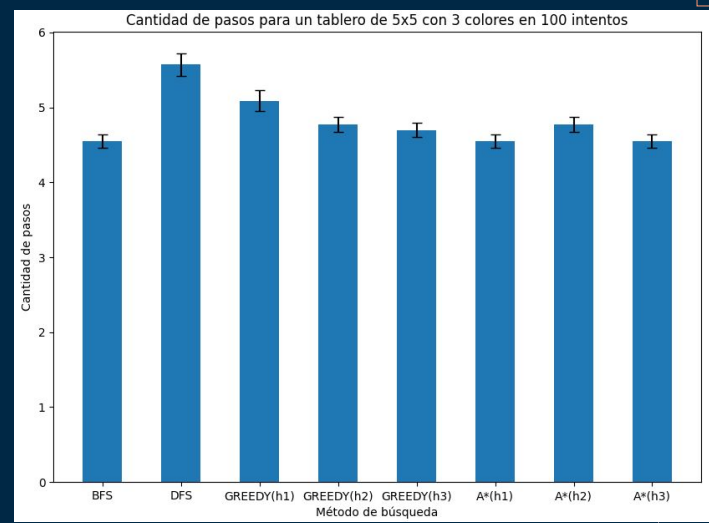
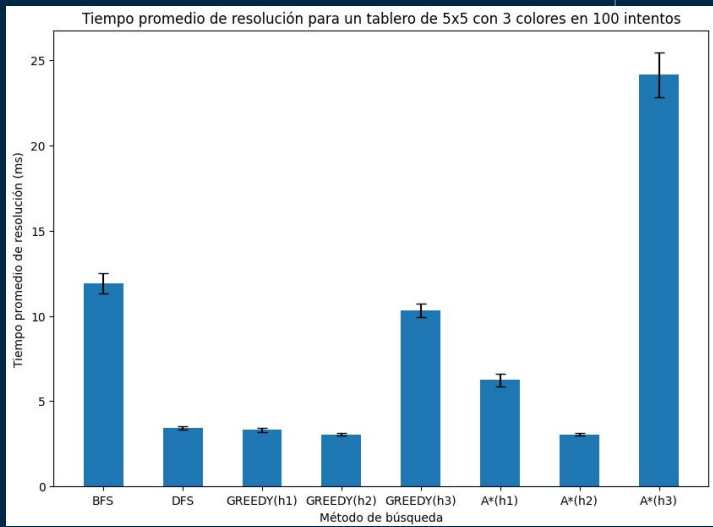


The background is a solid dark blue. It features several decorative elements: a vertical white line on the left with a small cyan square at its base; a vertical white line in the upper center with a small orange square at its base; a vertical white line on the right with a small cyan square at its base; and a vertical white line on the far right. Scattered throughout are small squares in cyan, orange, and pink. The text is centered in a white, sans-serif font.

Tamaño 5x5 y
variando colores

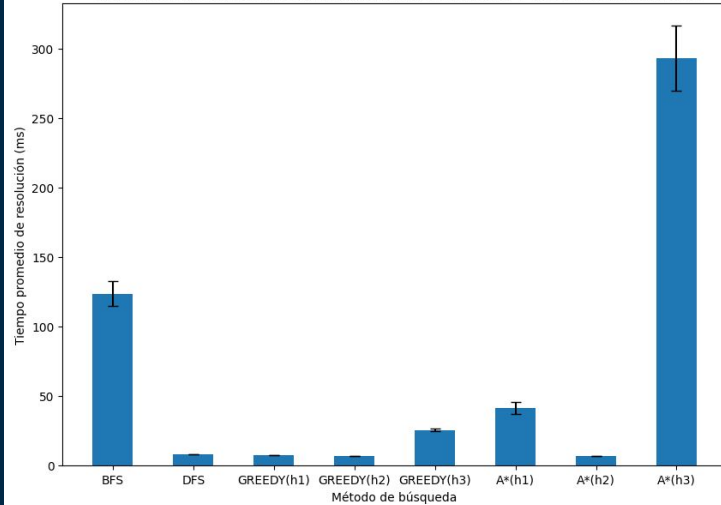


	Tiempo [ms]	Pasos	Nodos visitados	Nodos frontera
BFS	0,90 ± 0,06	3	4	3
DFS	0,95 ± 0,05	3	3	3
Greedy (h1)	1,00 ± 0,06	3	3	3
Greedy (h2)	1,00 ± 0,06	3	3	3
Greedy (h3)	2,20 ± 0,15	3	3	3
A* (h1)	0,95 ± 0,05	3	3	3
A* (h2)	1,00 ± 0,06	3	3	3
A* (h3)	4,90 ± 0,25	3	3	3

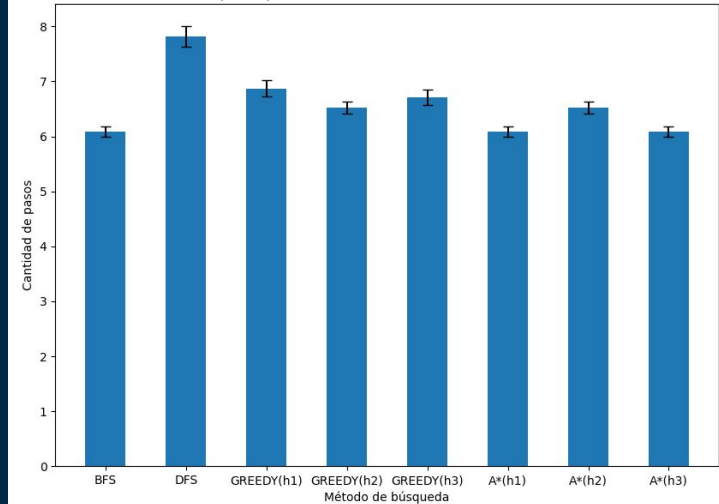


	Tiempo [ms]	Pasos	Nodos visitados	Nodos frontera
BFS	11,90 ± 0,60	4 ± 1	24 ± 1	34 ± 1
DFS	3,40 ± 0,10	5 ± 1	5 ± 1	9 ± 1
Greedy (h1)	3,30 ± 0,10	5 ± 1	5 ± 1	8 ± 1
Greedy (h2)	3,05 ± 0,10	4 ± 1	4 ± 1	8 ± 1
Greedy (h3)	10,30 ± 0,40	4 ± 1	4 ± 1	8 ± 1
A* (h1)	6,25 ± 0,40	4 ± 1	9 ± 1	15 ± 1
A* (h2)	3,05 ± 0,10	4 ± 1	4 ± 1	8 ± 1
A* (h3)	24,10 ± 1,30	4 ± 1	13 ± 1	22 ± 1

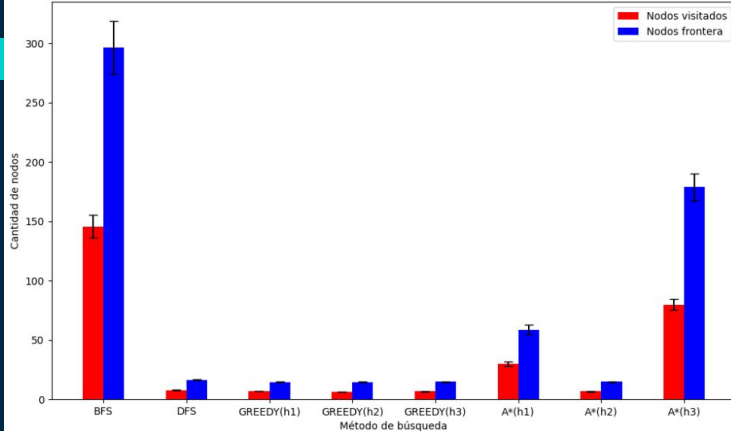
Tiempo promedio de resolución para un tablero de 5x5 con 4 colores en 100 intentos



Cantidad de pasos para un tablero de 5x5 con 4 colores en 100 intentos

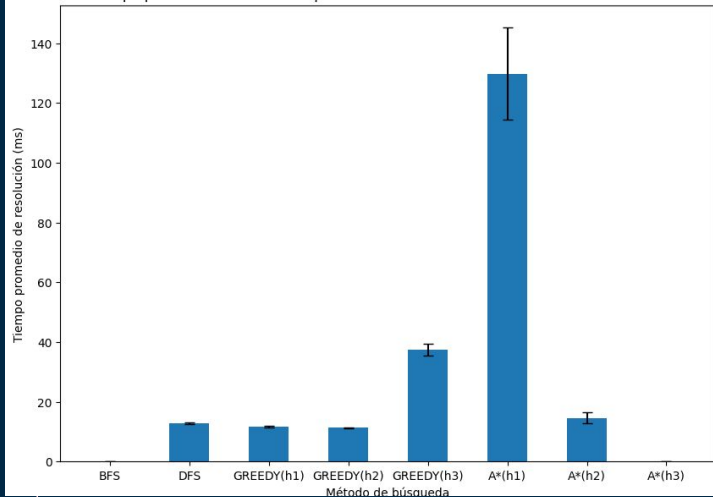


Cantidad de nodos para un tablero de 5x5 con 4 colores en 100 intentos

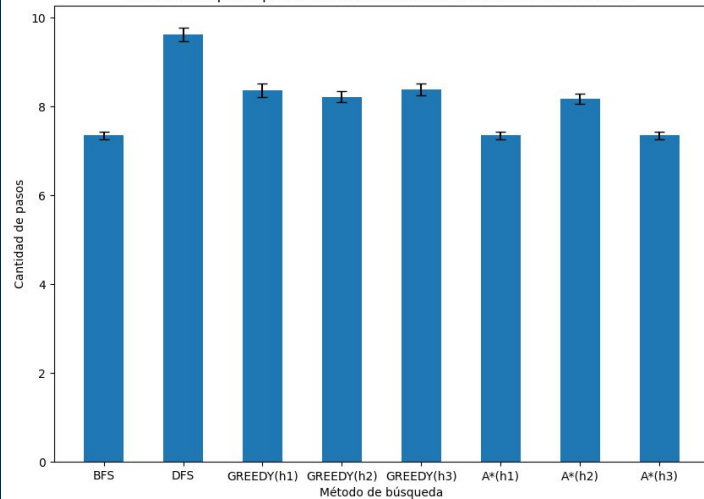


	Tiempo [ms]	Pasos	Nodos visitados	Nodos frontera
BFS	123,60 ± 8,80	6 ± 1	146 ± 10	297 ± 22
DFS	8,05 ± 0,20	8 ± 1	8 ± 1	16 ± 1
Greedy (h1)	7,45 ± 0,20	7 ± 1	7 ± 1	14 ± 1
Greedy (h2)	6,80 ± 0,15	7 ± 1	7 ± 1	14 ± 1
Greedy (h3)	25,55 ± 0,90	7 ± 1	7 ± 1	15 ± 1
A* (h1)	41,55 ± 4,50	6 ± 1	30 ± 2	59 ± 4
A* (h2)	7,00 ± 0,20	7 ± 1	7 ± 1	15 ± 1
A* (h3)	295,00 ± 23,50	6 ± 1	80 ± 5	179 ± 11

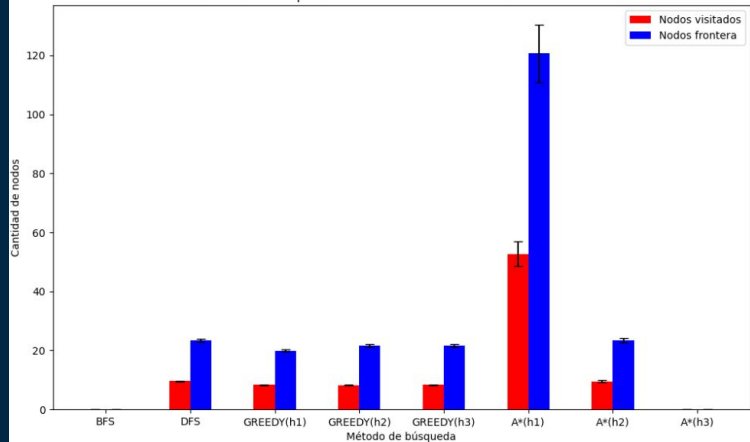
Tiempo promedio de resolución para un tablero de 5x5 con 5 colores en 100 intentos



Cantidad de pasos para un tablero de 5x5 con 5 colores en 100 intentos

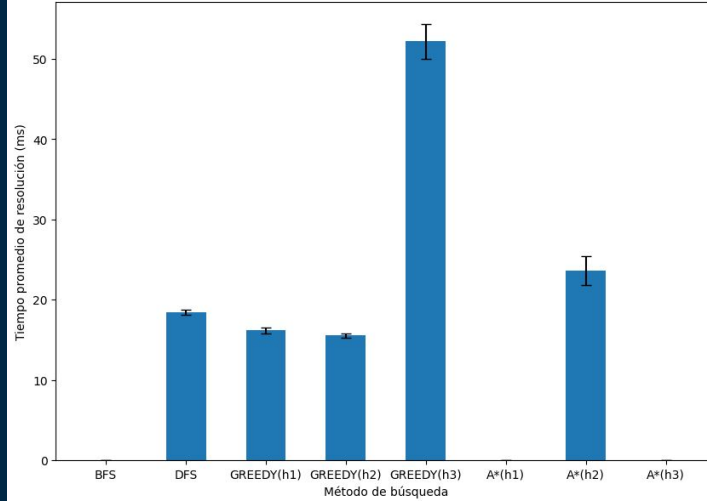


Cantidad de nodos para un tablero de 5x5 con 5 colores en 100 intentos

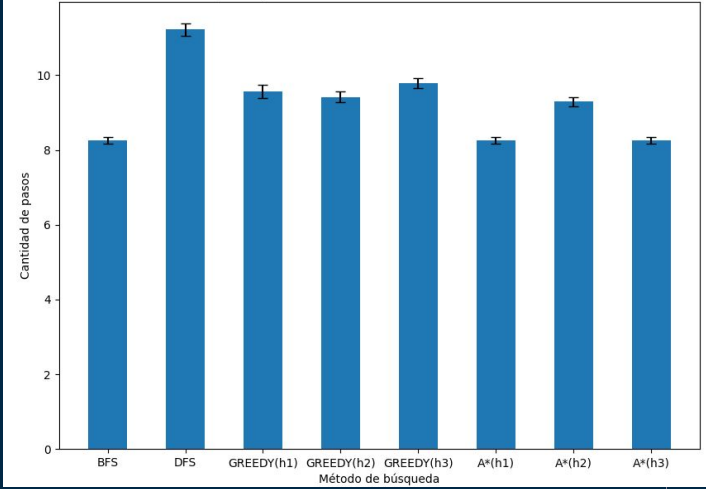


	Tiempo [ms]	Pasos	Nodos visitados	Nodos frontera
BFS	715,10 ± 45,10	7 ± 1	574 ± 34	1464 ± 95
DFS	12,85 ± 0,25	10 ± 1	10 ± 1	24 ± 1
Greedy (h1)	11,60 ± 0,20	8 ± 1	8 ± 1	20 ± 1
Greedy (h2)	11,35 ± 0,20	8 ± 1	8 ± 1	22 ± 1
Greedy (h3)	37,35 ± 1,95	8 ± 1	8 ± 1	22 ± 1
A* (h1)	129,80 ± 15,50	7 ± 1	53 ± 4	121 ± 10
A* (h2)	14,50 ± 1,80	8 ± 1	9 ± 1	23 ± 1
A* (h3)	2595,00 ± 289,00	7 ± 1	333 ± 22	934 ± 65

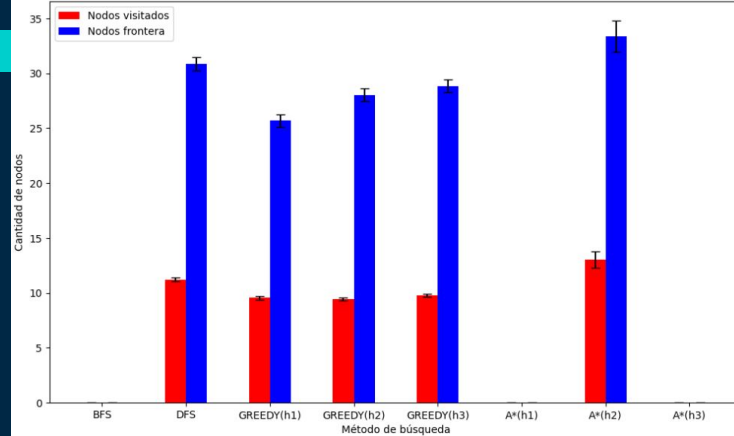
Tiempo promedio de resolución para un tablero de 5x5 con 6 colores en 100 intentos



Cantidad de pasos para un tablero de 5x5 con 6 colores en 100 intentos



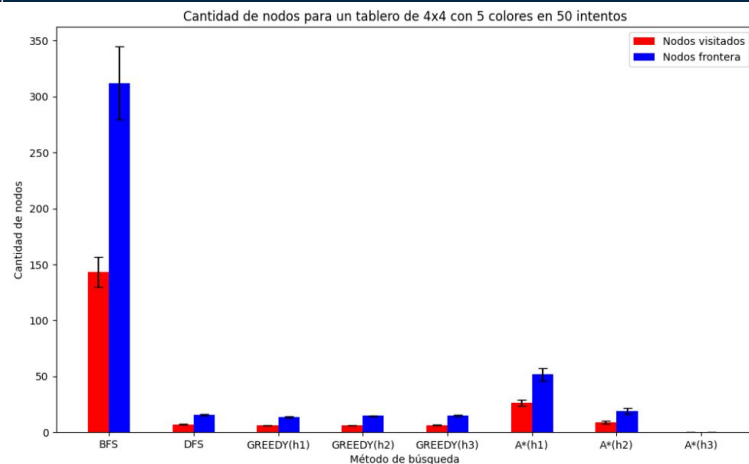
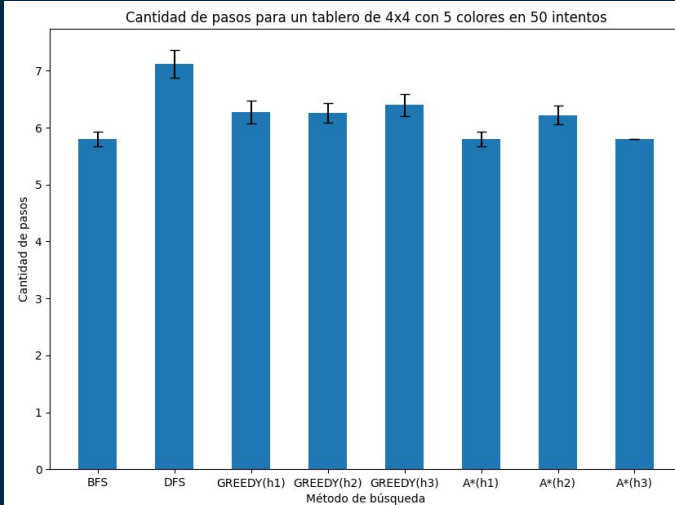
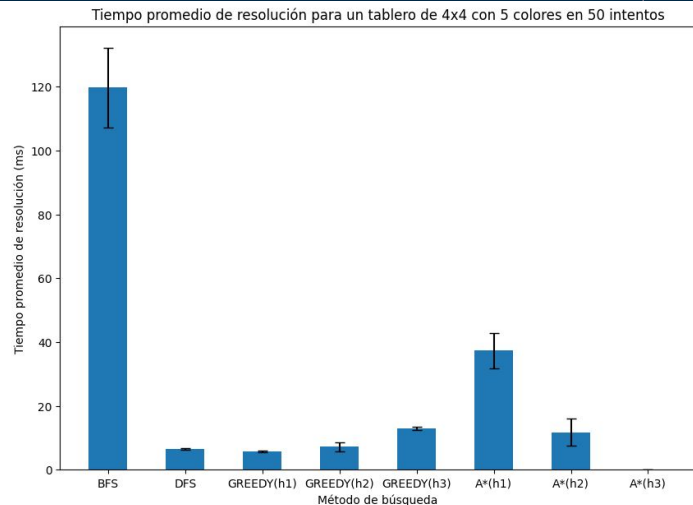
Cantidad de nodos para un tablero de 5x5 con 6 colores en 100 intentos



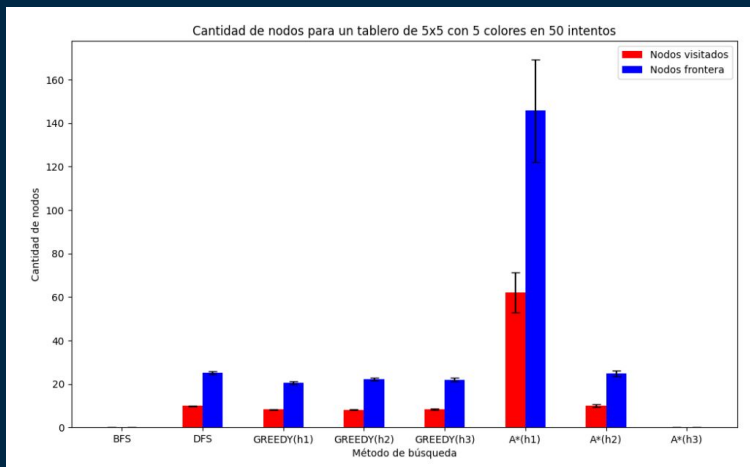
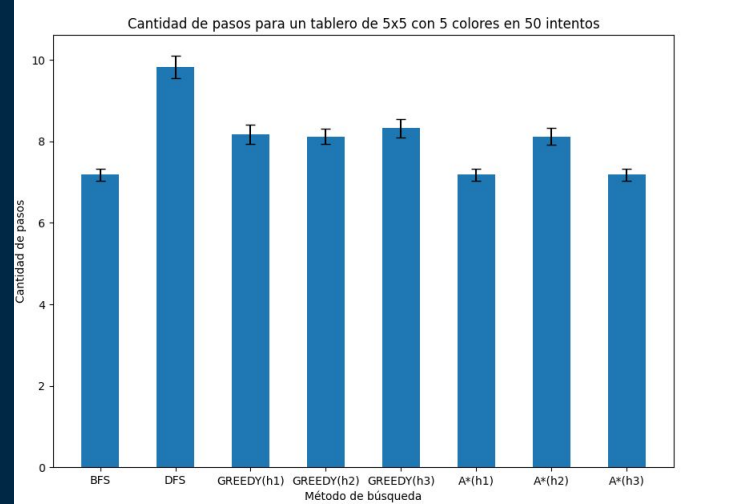
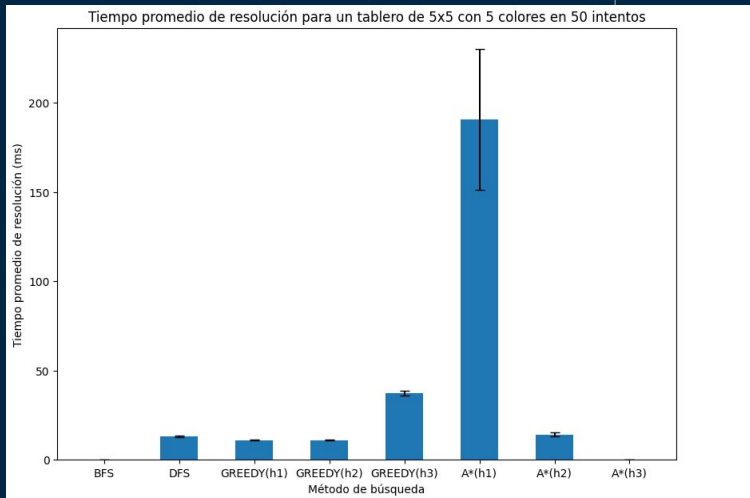
	Tiempo [ms]	Pasos	Nodos visitados	Nodos frontera
BFS	3537,00 ± 305,00	8 ± 1	1942 ± 144	5960 ± 475
DFS	18,50 ± 0,30	11 ± 1	11 ± 1	31 ± 1
Greedy (h1)	16,20 ± 0,40	10 ± 1	10 ± 1	26 ± 1
Greedy (h2)	15,50 ± 0,30	9 ± 1	9 ± 1	28 ± 1
Greedy (h3)	52,20 ± 2,20	10 ± 1	10 ± 1	29 ± 1
A* (h1)	708,00 ± 240,00	8 ± 1	89 ± 10	228 ± 28
A* (h2)	23,60 ± 1,80	9 ± 1	13 ± 1	33 ± 1
A* (h3)	339935,00 ± 174100,00	8 ± 1	1215 ± 110	4035 ± 378

The background is a solid dark blue. It features several decorative elements: a vertical white line on the left with a small cyan square at its base; a vertical white line near the top center with a small orange square at its base; a vertical white line on the right with a small cyan square at its base; a vertical white line on the far right; and several small squares in cyan, orange, and pink scattered across the upper and right portions of the frame.

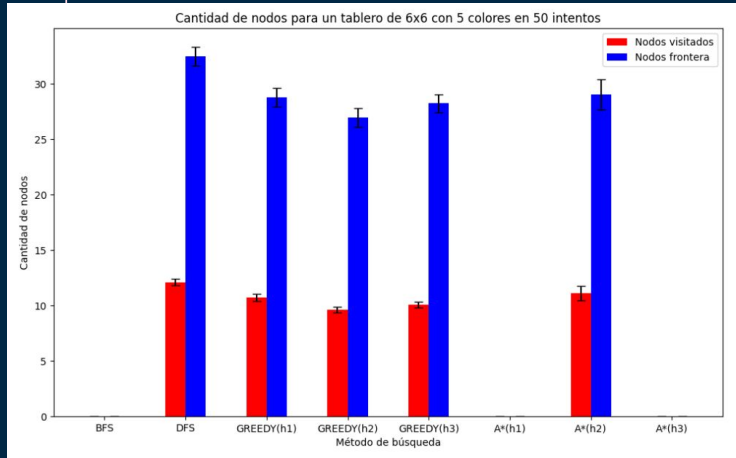
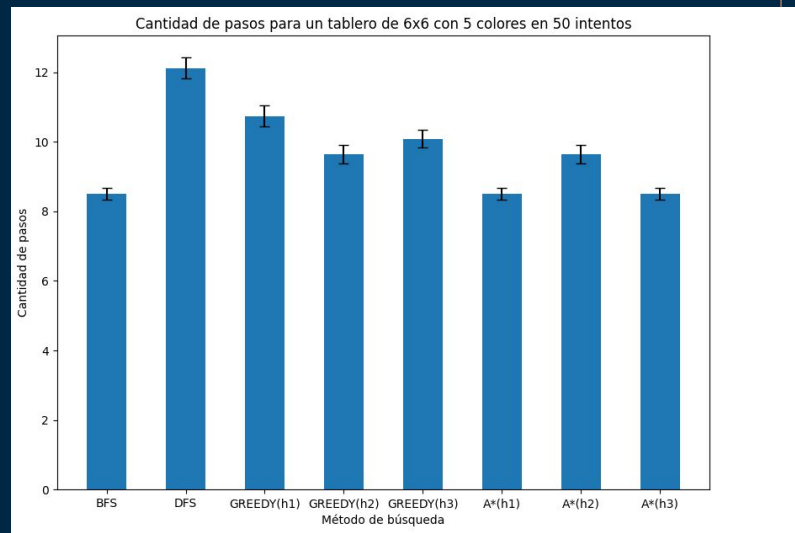
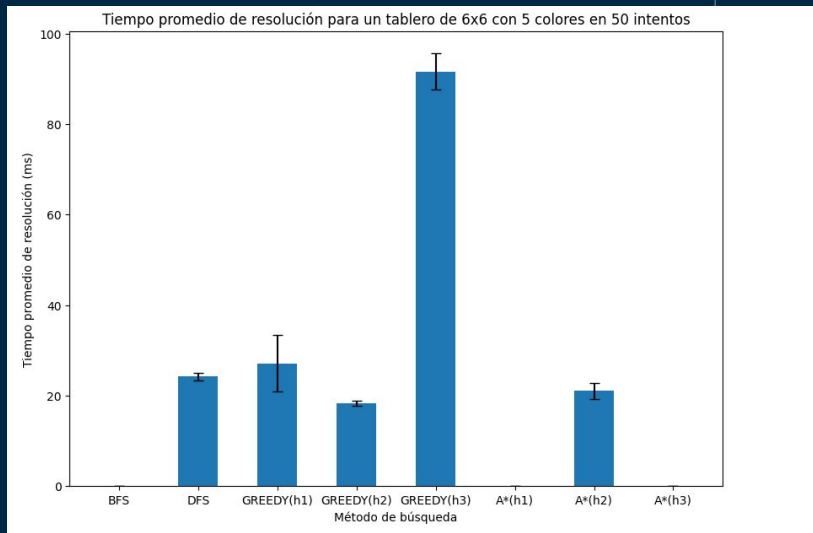
5 colores y
variando tamaño



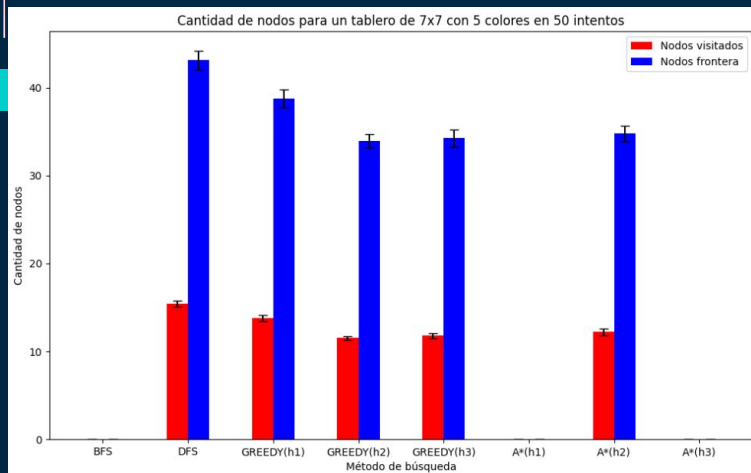
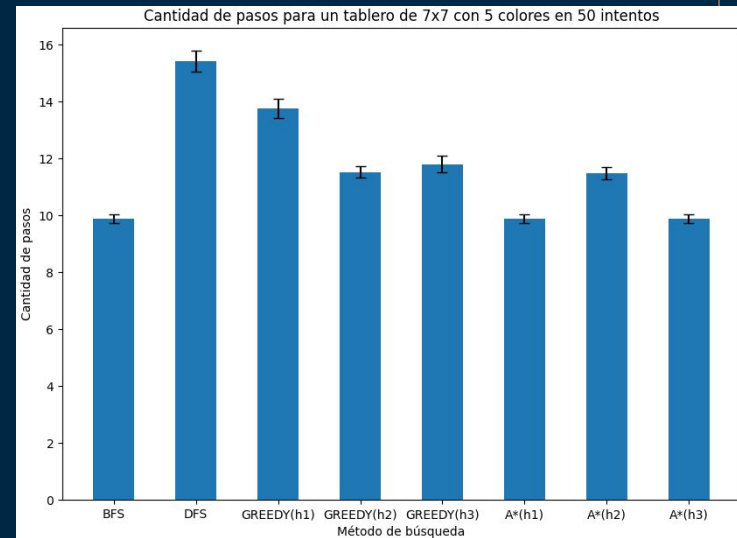
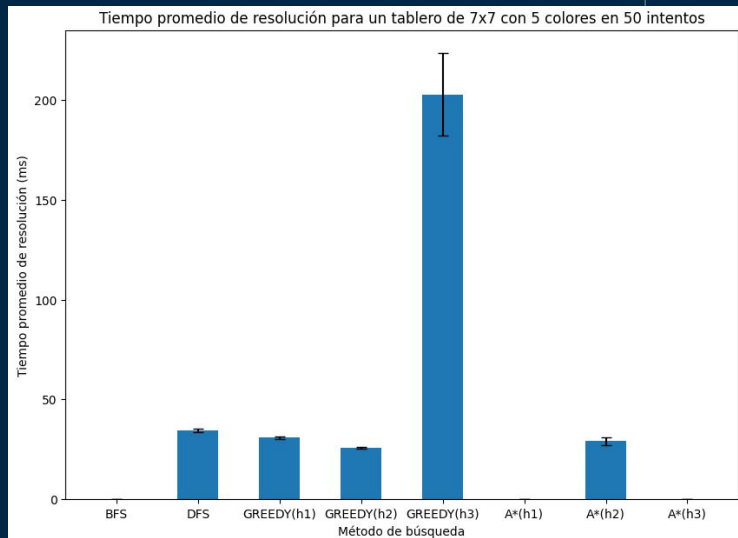
	Tiempo [ms]	Pasos	Nodos visitados	Nodos frontera
BFS	119,70 ± 12,50	6 ± 1	144 ± 14	312 ± 32
DFS	6,50 ± 0,25	7 ± 1	7 ± 1	16 ± 1
Greedy (h1)	5,70 ± 0,20	7 ± 1	7 ± 1	14 ± 1
Greedy (h2)	7,20 ± 1,50	7 ± 1	7 ± 1	15 ± 1
Greedy (h3)	12,90 ± 0,60	7 ± 1	7 ± 1	15 ± 1
A* (h1)	37,30 ± 5,40	6 ± 1	27 ± 3	52 ± 6
A* (h2)	11,70 ± 4,30	7 ± 1	9 ± 1	19 ± 3
A* (h3)	261,90 ± 39,70	6 ± 1	93 ± 9	228 ± 25



	Tiempo [ms]	Pasos	Nodos visitados	Nodos frontera
BFS	765 ± 84,40	7 ± 1	590 ± 57	1520 ± 158
DFS	13,20 ± 0,40	10 ± 1	10 ± 1	25 ± 1
Greedy (h1)	11 ± 0,40	8 ± 1	8 ± 1	21 ± 1
Greedy (h2)	11 ± 0,25	8 ± 1	8 ± 1	22 ± 1
Greedy (h3)	37,40 ± 1,40	8 ± 1	8 ± 1	22 ± 1
A* (h1)	190,6 ± 39,55	7 ± 1	62 ± 9	146 ± 24
A* (h2)	14,20 ± 1,25	8 ± 1	9 ± 1	25 ± 2
A* (h3)	5014,3 ± 1480	7 ± 1	355 ± 47	1000 ± 136



	Tiempo [ms]	Pasos	Nodos visitados	Nodos frontera
BFS	4495 ± 440	8 ± 1	2230 ± 193	6370 ± 578
DFS	24 ± 0,90	12 ± 1	12 ± 1	32 ± 1
Greedy (h1)	27 ± 6,30	11 ± 1	11 ± 1	29 ± 1
Greedy (h2)	18,40 ± 0,55	10 ± 1	10 ± 1	27 ± 1
Greedy (h3)	91,60 ± 4,05	9 ± 1	9 ± 1	28 ± 1
A* (h1)	565,70 ± 150,50	8 ± 1	126 ± 19	346 ± 53
A* (h2)	21 ± 1,70	10 ± 1	11 ± 1	29 ± 2
A* (h3)	42811 ± 14260	8 ± 1	1003 ± 105	3080 ± 324



	Tiempo [ms]	Pasos	Nodos visitados	Nodos frontera
BFS	25448,10 ± 4263,30	10 ± 1	8757 ± 1017	26856 ± 3215
DFS	34,50 ± 0,80	15 ± 1	15 ± 1	43 ± 1
Greedy (h1)	30,80 ± 0,70	14 ± 1	14 ± 1	39 ± 1
Greedy (h2)	25,60 ± 0,50	12 ± 1	12 ± 1	34 ± 1
Greedy (h3)	202,70 ± 20,80	12 ± 1	12 ± 1	34 ± 1
A* (h1)	3697,10 ± 1572,80	10 ± 1	330 ± 51	1056 ± 173
A* (h2)	29,00 ± 2,00	11 ± 1	12 ± 1	35 ± 1
A* (h3)	568461,80 ± 247468,90	10 ± 1	2775 ± 348	9069 ± 1160

Conclusiones de los gráficos

Primeramente, podemos notar que A^* con heurísticas admisibles logra obtener el mismo camino mínimo que BFS.

Sin embargo, en cuanto a tiempo y espacio, A^* es más eficiente que BFS, que al ser un método no informado, explora todos los posibles caminos a seguir. La única excepción en cuanto a tiempo es en la heurística de Dijkstra (h_3), ya que debe generarse el grafo antes de comenzar el algoritmo.

Además, como A^* descarta caminos tiene, por ende, un menor número de nodos visitados y nodos frontera que BFS.

También podemos ver que la heurística 2 (mayor número de celdas adyacentes) no es admisible ya que no devuelve el mismo promedio que las otras.

Conclusiones de los gráficos

En segunda instancia, podemos notar que tanto Local Greedy como DFS, se expanden de manera similar, por lo cual en ambos es lo mismo la cantidad de nodos visitados que los nodos solución.

Además, como el algoritmo GREEDY nos acerca en cada paso al mejor camino (ya que es informado), suele tener un promedio de pasos menor al DFS.

Conclusiones de los gráficos

Por último, haremos una comparación de las heurísticas elegidas, según su performance temporal y de la solución que arrojan.

h(2) es la más veloz (ya que prioriza adquirir la mayor cantidad de territorio en cada paso y nos lleva más rápido a la solución)

h(1) le sigue, ya que es un cálculo que se realiza iterando sobre el tablero

h(3) es la última ya que en cada instancia debe generar un nuevo grafo del tablero para decidir el siguiente paso.

En cuanto a solución, h(1) y h(3) son las mejores ya que arrojan el mismo resultado que BFS, ya que son admisibles.