MÉTODOS DE BÚSQUEDA

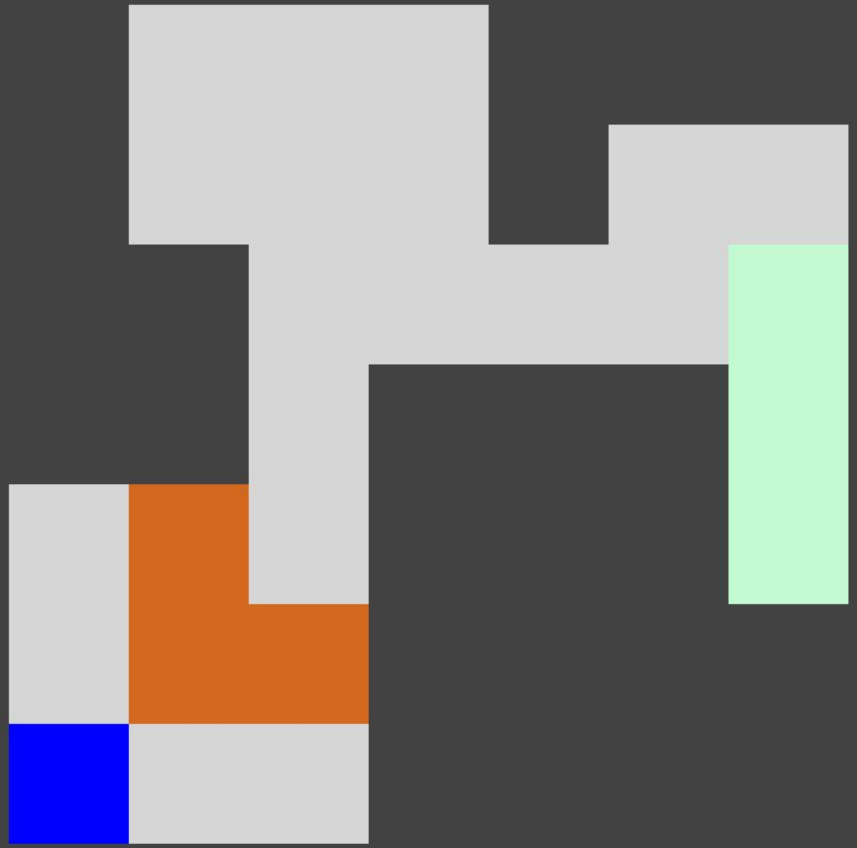
DESINFORMADOS E INFORMADOS

TABLA DE CONTENIDOS

01

Introducción

Moves: 0



SOKOBAN

Colores:

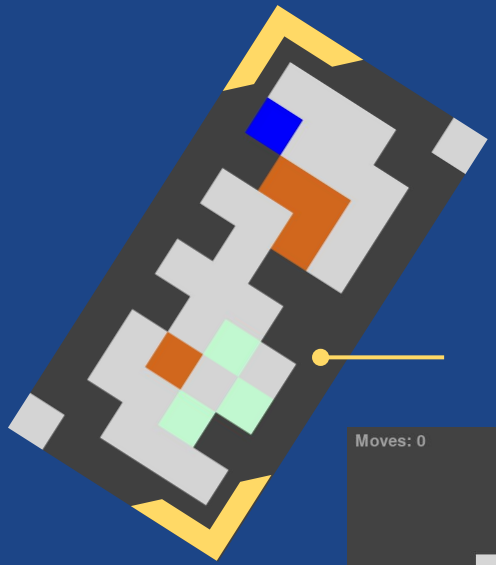
Jugador → Azul

Cajas → Naranja

Paredes → Gris oscuro

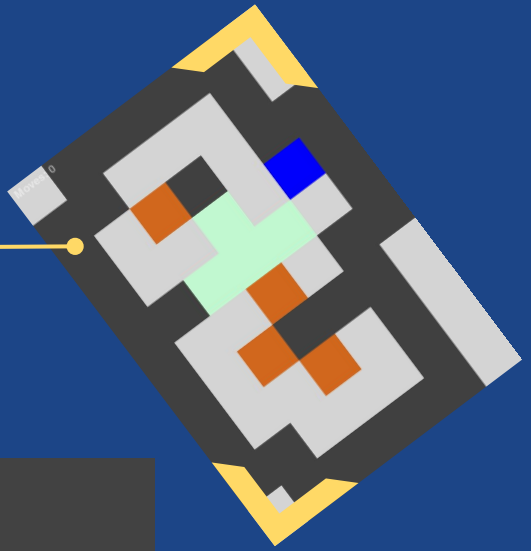
Objetivos → Verde agua

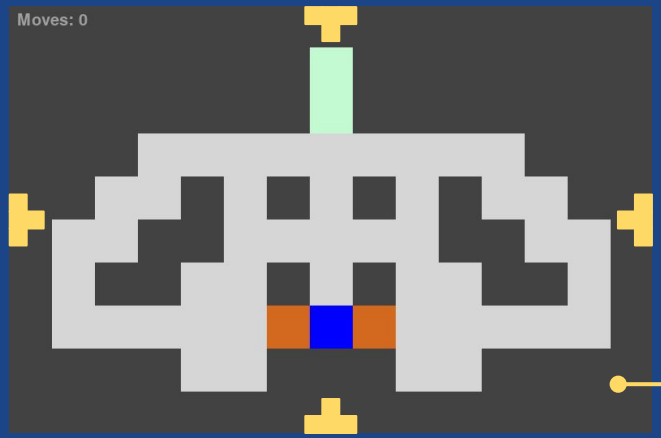
Niveles



PUZZLE_13

PUZZLE_16

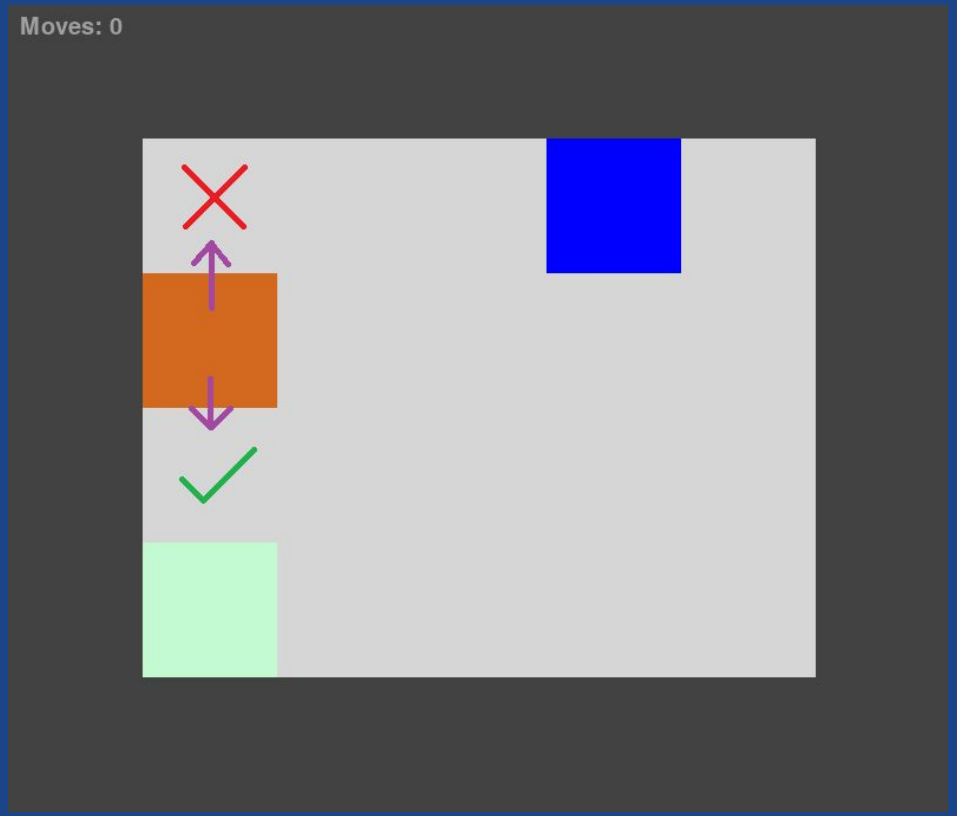




SOKO_01

Tecnología y herramientas

Optimizacion



02

Métodos de búsqueda desinformados

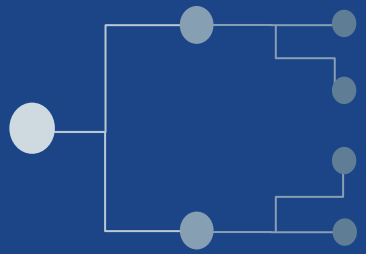
Busqueda Desinformada

BFS

- Completo (para finitos nodos)
- Óptimo (con costo uniforme)
- Expande menor profundidad
- Requiere mucha memoria
- Lento

DFS

- No es óptimo
- Expande mayor profundidad
- Menor complejidad espacial
- Más rápida que BFS por lo general



IDDFS

- Puede ser utilizado para obtener el camino óptimo
- Realiza DLS pero la profundidad varía

Resultados

Algoritmo	Tiempo	Profundidad	Nodos Expandidos	Nodos Frontera
BFS	1,4490	78	30377	18
DFS	0,4518	614	11647	243
IDDFS (10)	2,5821	78	40965	144

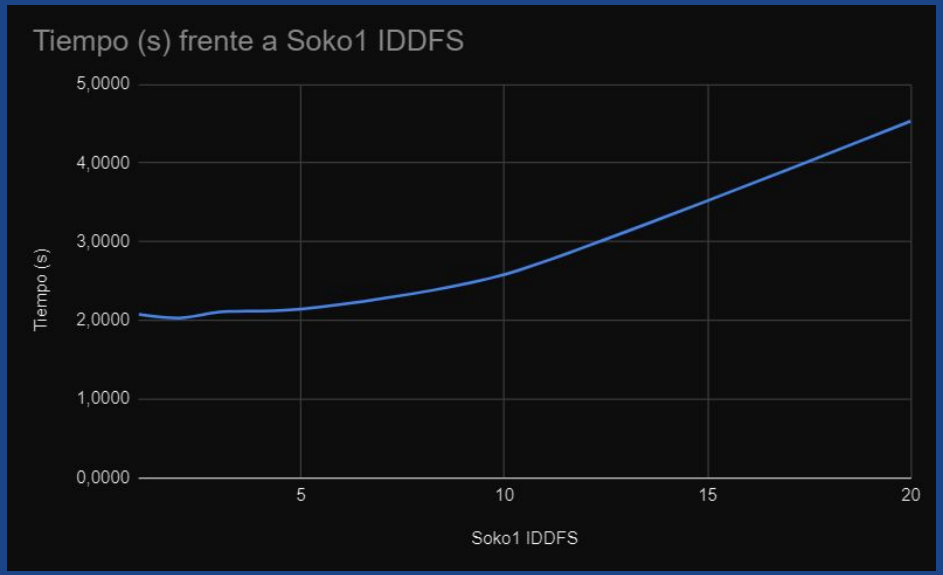
Mediciones llevadas a cabo usando el mapa mencionado como **soko_1**

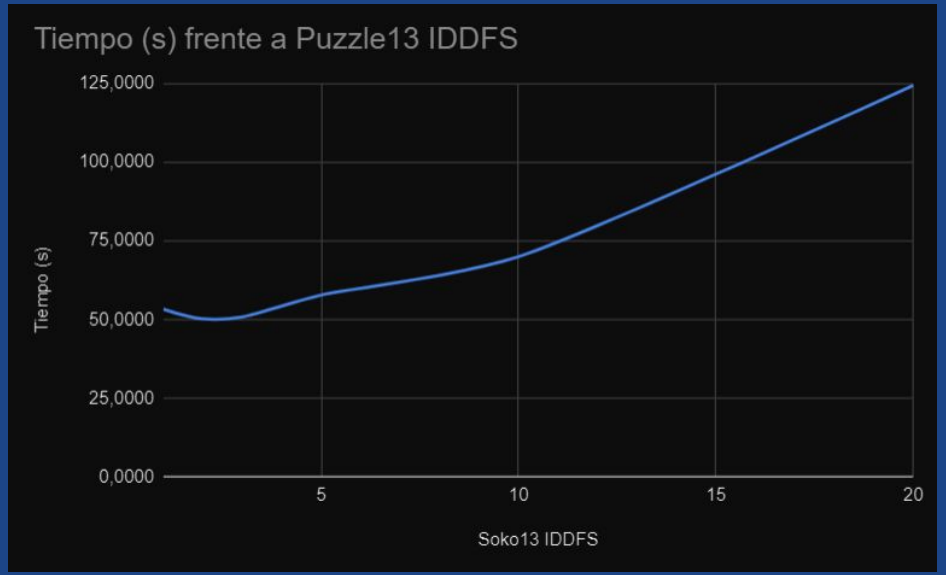
Resultados

Soko1 IDDFS	Tiempo	Profundidad	Nodos Expandidos	Nodos Frontera
1	2,0750	78	30380	18
2	2,0293	78	30382	18
3	2,1069	78	31617	14
5	2,1420	78	33322	36
10	2,5821	78	40965	144
20	4,5317	78	65343	939

Puzzle13 IDDFS	Tiempo	Profundidad	Nodos Expandidos	Nodos Frontera
1	53,2809	64	480690	6551
2	50,2016	64	486741	6241
3	50,8141	64	496211	8544
5	57,7783	64	516861	12819
10	69,9668	64	682427	20513
20	124,3688	64	1239856	34417

Datos


Tiempo (s) frente a Soko1 IDDFS
5,0000
4,0000
3,0000
2,0000
1,0000
0,0000
Tiempo (s)
5
10
15
20
Soko1 IDDFS


Tiempo (s) frente a Puzzle13 IDDFS
125,0000
100,0000
75,0000
50,0000
25,0000
0,0000
Tiempo (s)
5
10
15
20
Soko13 IDDFS

03

Heurísticas

Implementaciones

Heurística #1

Manhattan entre el jugador y la caja más cercana, sumado a la distancia de esta caja con el objetivo más cercano a la caja

Heurística #2

Manhattan entre el jugador y la caja más cercana, sumado a las distancias entre cada caja y su objetivo más cercano

Heurística #3

Cantidad de cajas fuera de objetivos

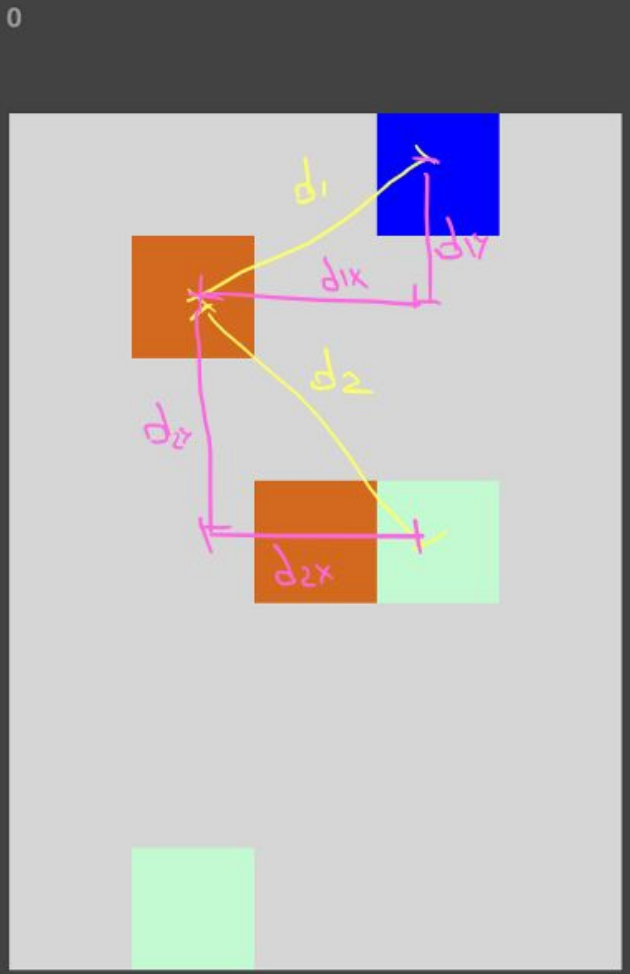
Implementaciones

Heurística #4

Movimientos mínimos (1 paso por dirección y par caja-objetivo) necesario para llevar las cajas a los objetivos

Heurística #5

Distancia entre el centro de masa de las cajas y el centro de masa de los objetivos

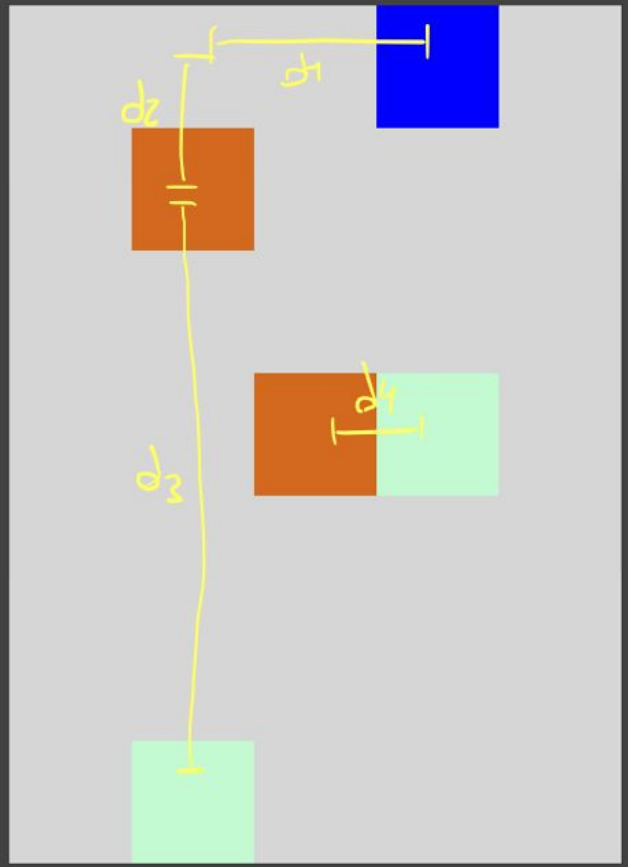


Heurística 1

Primero el jugador encuentra la distancia a la caja más cerca (d1) y la distancia al objetivo más cerca desde esa caja (d2)

Se cumple que d1x+d1y es la distancia de manhattan entre el jugador y la caja más cercana (la menor cantidad de pasos en un caso ideal y sin obstáculos) y d2x+d2y es la distancia de manhattan entre la caja y el objetivo más cercano.

La heurística devuelve como valor d1x+d1y+d2x+d2y que es siempre menor a la mínima cantidad de pasos que se tendrían que realizar para mover la caja más cercana al objetivo más cercano.



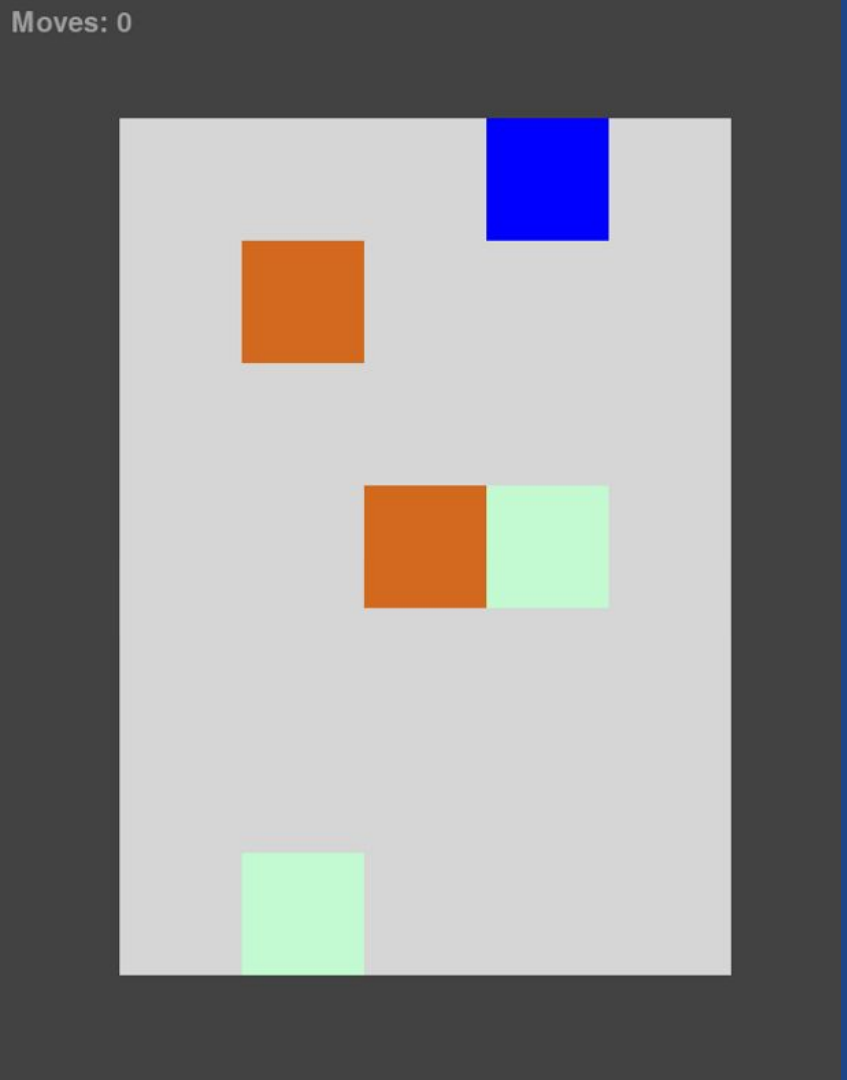
Heurística 2

Primero se calcula la distancia entre cada caja y su objetivo más cercano, si hay conflictos entre cajas se le da prioridad a la caja más cercana a ese objetivo.

Luego se suma la distancia entre el jugador y la caja que tenga más cerca y además las distancias entre cada caja y su respectivo objetivo más cercano.

En este caso la heurística devuelve d1+d2+d3+d4

Este número siempre es más chico que la cantidad total de movimientos que hace falta hacer para que cada caja quede en un goal.

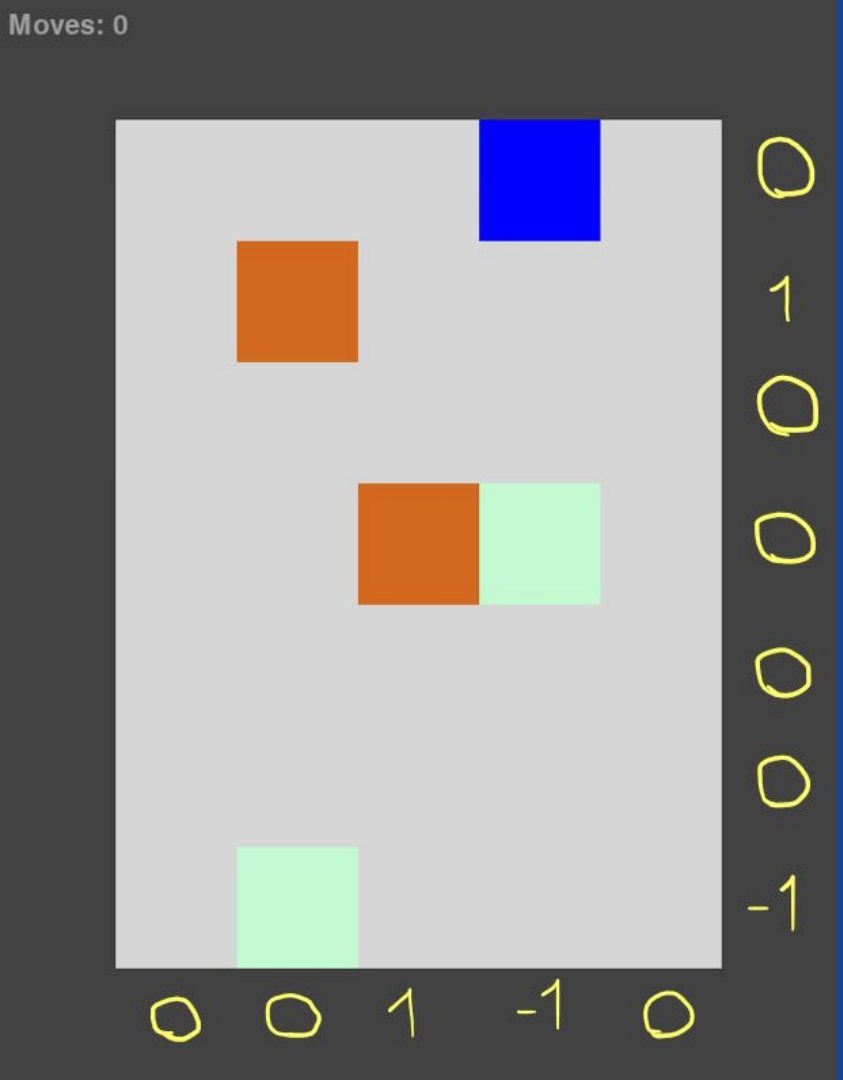


Heurística 3

Esta heurística tiene en cuenta cuántas cajas no se encuentran ya en algún objetivo.

En este caso hay 2 cajas y 2 objetivo, y como no hay ninguna caja en un objetivo, hay 2 cajas que no están en algún objetivo.

La heurística devuelve el número de cajas que no se encuentran en objetivos que siempre será menor a la mínima cantidad de movimientos que el jugador tendría que hacer para meter las cajas en dichos objetivos (en este caso como mínimo 2 movimientos).

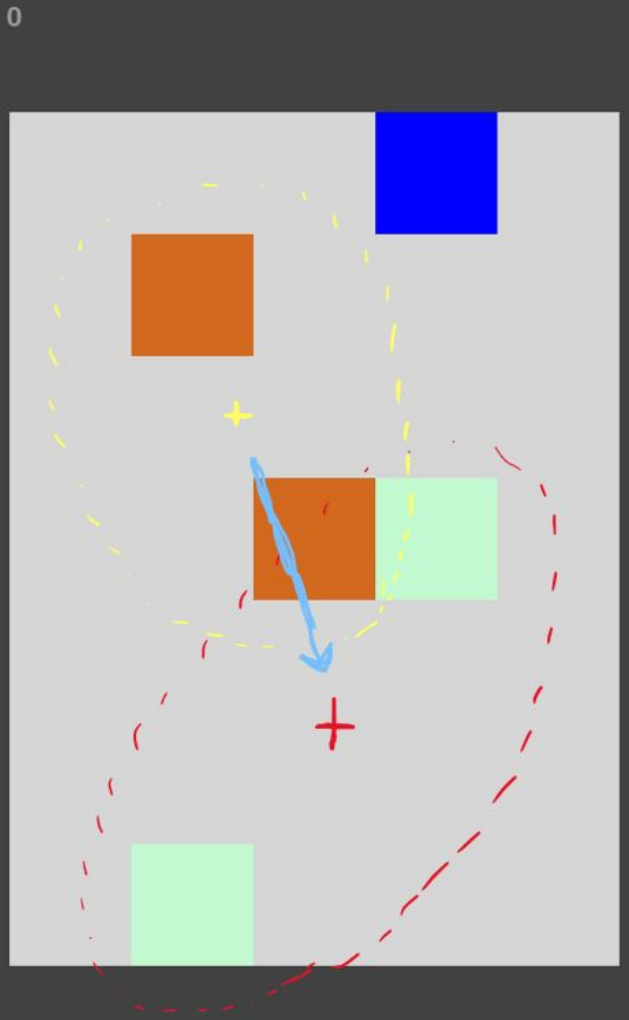


Heurística 4

Esta heurística analiza dos arrays, una para el eje Y y otra para el eje X. A cada lugar dentro del array se le suma 1 si hay una caja en esa coordenada y -1 si hay un objetivo.

La heurística devuelve la suma de los módulos de los distintos elementos del array (en este caso 4, 2 por cada array) y luego lo divide por dos, entonces devolverá en este caso 2.

Esto es menos que la menor cantidad de movimientos que el player tendría que hacer para posicionar a las cajas en los objetivos.



Heurística 5

Esta heurística tiene en cuenta la distancia entre el centro de masa de las cajas y el centro de masa de los goals.

Siempre devuelve esa distancia.

04

Métodos de búsqueda informados

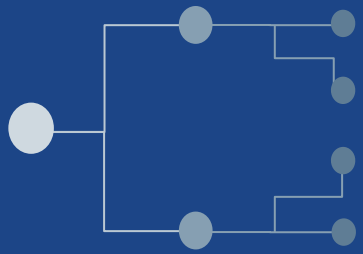
Busqueda Informada

GGS

- No es optima
- Expande el nodo que posea menor h(n)

A*

- Óptima y completa bajo ciertos casos
- F(n) = g(n) + h(n)
- Expande el nodo que posea menor F(n)



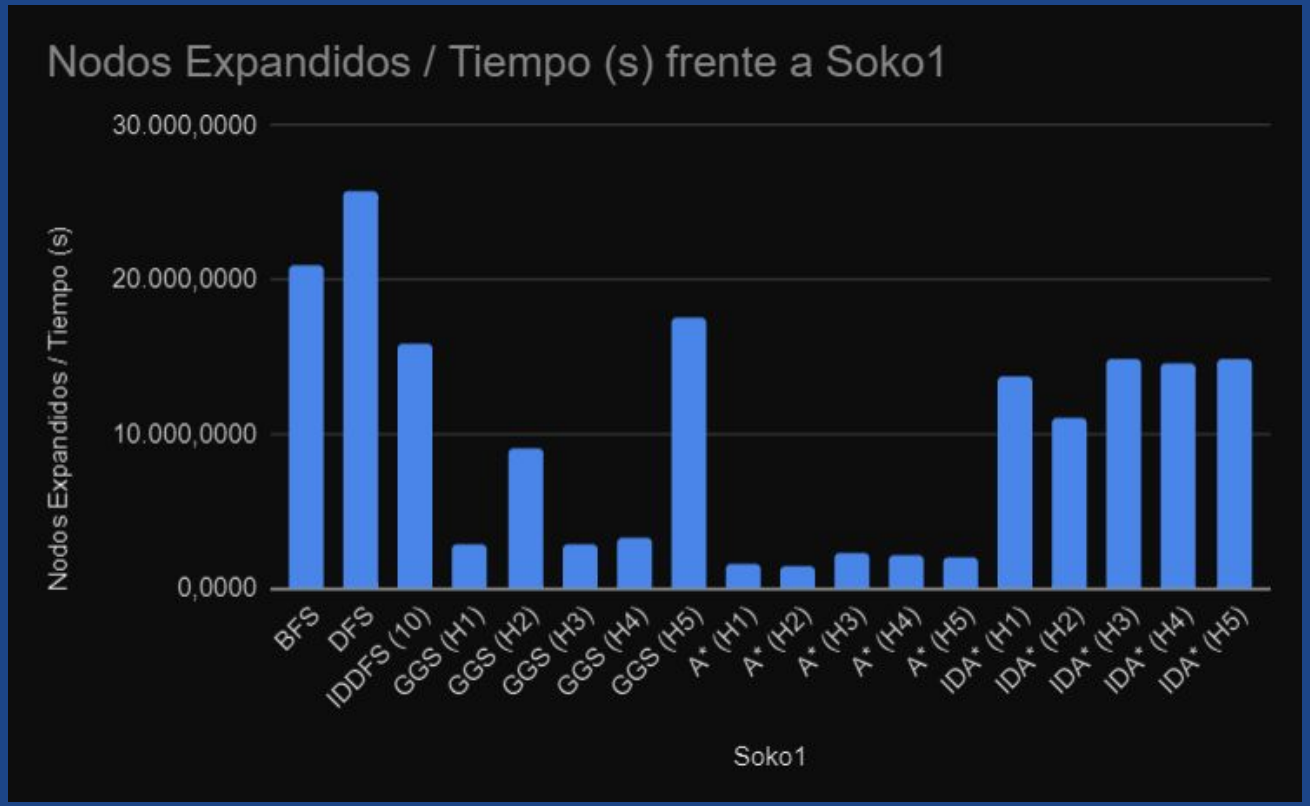
IDA*

- Óptima y completa bajo los mismos casos que A*
- Se realiza DFS hasta un Lim = F(n)

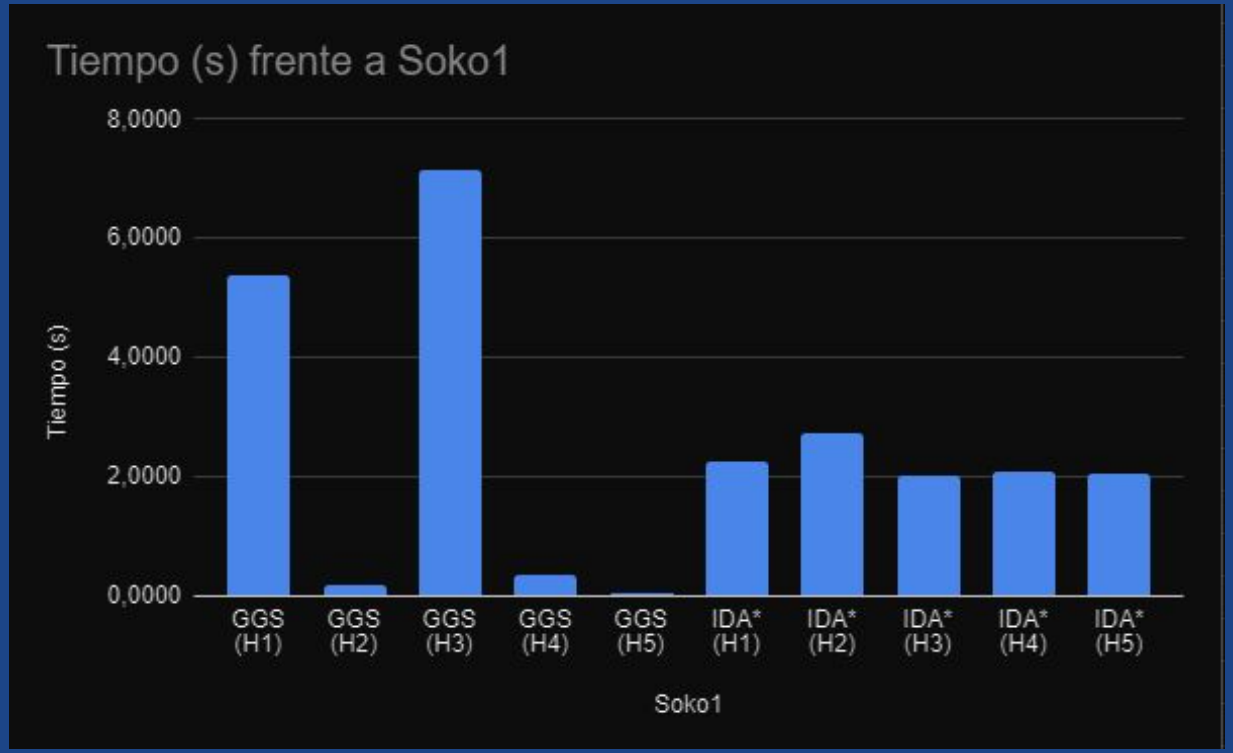
Busqueda Informada

Algoritmo	Tiempo	Profundidad	Nodos Expandidos	Nodos Frontera
GGS (H1)	5,3860	112	15519	2585
GGS (H2)	0,2020	86	1847	535
GGS (H3)	7,1429	82	20385	700
GGS (H4)	0,3427	96	3691	203
GGS (H5)	0,0490	90	739	79
A* (H1)	19,2750	78	31691	95
A* (H2)	21,1640	78	30483	154
A* (H3)	13,3410	78	30351	27
A* (H4)	14,1240	78	30317	33
A* (H5)	15,1440	78	30340	29
IDA* (H1)	2,2710	78	31269	77
IDA* (H2)	2,7330	78	30051	154
IDA* (H3)	2,0345	78	30347	28
IDA* (H4)	2,0736	78	30320	33
IDA* (H5)	2,0461	78	30352	25

Datos


Nodos Expandidos / Tiempo (s) frente a Soko1
30.000,0000
20.000,0000
10.000,0000
0,0000
Nodos Expandidos / Tiempo (s)
BFS
DFS
IDDFS (10)
GGS (H1)
GGS (H2)
GGS (H3)
GGS (H4)
GGS (H5)
A* (H1)
A* (H2)
A* (H3)
A* (H4)
A* (H5)
IDA* (H1)
IDA* (H2)
IDA* (H3)
IDA* (H4)
IDA* (H5)
Soko1

Datos


Tiempo (s) frente a Soko1
8,0000
6,0000
4,0000
2,0000
0,0000
Tiempo (s)
GGS (H1)
GGS (H2)
GGS (H3)
GGS (H4)
GGS (H5)
IDA* (H1)
IDA* (H2)
IDA* (H3)
IDA* (H4)
IDA* (H5)
Soko1

Soko_1 sin y con optimizaciones

Soko1	Tiempo (s)	Profundidad	Nodos Expandido:	Nodos Frontera
BFS	1,4490	78	30377	18
DFS	0,4518	614	11647	243
IDDFS (10)	2,5821	78	40965	144
GGS (H1)	5,3860	112	15519	2585
GGS (H2)	0,2020	86	1847	535
GGS (H3)	7,1429	82	20385	700
GGS (H4)	0,3427	96	3691	203
GGS (H5)	0,0490	90	739	79
A* (H1)	19,2750	78	31691	95
A* (H2)	21,1640	78	30483	154
A* (H3)	13,3410	78	30351	27
A* (H4)	14,1240	78	30317	33
A* (H5)	15,1440	78	30340	29
IDA* (H1)	2,2710	78	31269	77
IDA* (H2)	2,7330	78	30051	154
IDA* (H3)	2,0345	78	30347	28
IDA* (H4)	2,0736	78	30320	33
IDA* (H5)	2,0461	78	30352	25

Soko1	Tiempo (s)	Profundidad	Nodos Expandido:	Nodos Frontera
BFS	0,657958	78	15650	18
DFS	0,202003	614	2789	241
IDDFS (10)	1,337021	78	20360	92
GGS (H1)	1,800092	112	9518	1276
GGS (H2)	0,140034	86	1417	339
GGS (H3)	1,746978	82	10471	361
GGS (H4)	0,260026	96	2678	137
GGS (H5)	0,032026	90	592	72
A* (H1)	4,477993	78	16185	69
A* (H2)	5,990001	78	15696	92
A* (H3)	4,323001	78	15621	27
A* (H4)	4,425994	78	15591	31
A* (H5)	4,439	78	15615	27
IDA* (H1)	1,070954	78	15953	59
IDA* (H2)	1,420029	78	15503	87
IDA* (H3)	1,14397	78	15620	28
IDA* (H4)	1,362004	78	15594	32
IDA* (H5)	1,122038	78	15628	23

Puzzle_13 sin y con optimizaciones

Puzzle_13	Tiempo (s)	Profundidad	Nodos Expandido	Nodos Frontera
BFS	28,7545	64	480689	6551
DFS	33,287499	188	502155	128
IDDFS (10)	60,697999	64	682427	20513
GGS (H1)	16,636988	64	42019	8721
GGS (H2)	12,465501	102	38212	3788
GGS (H3)	1,983999	68	14763	1961
GGS (H4)	1,006998	92	7916	1193
GGS (H5)	6,719499	82	62874	1550
A* (H1)				
A* (H2)				
A* (H3)				
A* (H4)				
A* (H5)				
IDA* (H1)				
IDA* (H2)				
IDA* (H3)				
IDA* (H4)				
IDA* (H5)				

Puzzle_13	Tiempo (s)	Profundidad	Nodos Expandido	Nodos Frontera
BFS	1,2175	64	28561	307
DFS	1,01	188	19714	125
IDDFS (10)	2,676002	64	38661	806
GGS (H1)	0,977002	64	4163	789
GGS (H2)	0,821001	102	6316	616
GGS (H3)	0,260501	68	4156	306
GGS (H4)	0,309998	92	4306	374
GGS (H5)	0,623001	82	9708	253
A* (H1)	9,839998	64	27491	838
A* (H2)	9,687501	64	17759	1563
A* (H3)	10,500499	64	27176	470
A* (H4)	11,667001	64	26700	562
A* (H5)	11,538001	64	27301	486
IDA* (H1)	1,862001	64	27472	799
IDA* (H2)	2,5725	64	18552	1514
IDA* (H3)	1,646498	64	27356	465
IDA* (H4)	1,8625	64	27000	514
IDA* (H5)	1,749002	64	27394	481

05

Conclusiones

Informados vs Desinformados

Pudimos observar que los métodos desinformados pudieron llegar a soluciones más rápido que algunos informados, pero esto se debe a que también procesan más nodos por segundo, puesto que ejecutan menor cantidad de cálculos por cada nodo.

Además, los métodos informados, si bien tardan más, muchas veces terminan procesando menos nodos que los procesados por métodos desinformados.

GGS, A* e IDA*

En ocasiones notamos que GGS conseguía una solución más rápidamente que los otros métodos informados, pero esto dependía de la heurística.

Por otra parte, pudimos comprobar que A* devuelve la solución óptima con una heurística admisible, y notamos que la versión superadora (IDA*) normalmente consigue el caso óptimo en menor tiempo.

¿Qué método de búsqueda usar?

A la hora de elegir un método de búsqueda, no existe una manera perfecta que se adecúe a todo problema.

Elegir un método informado o uno desinformado, dependerá de la facilidad de analizar “mejores casos” contra “peores casos”.
Por otra parte, dependerá del tamaño de posibilidades a analizar en cada paso, así como la geometría del ambiente del problema.

Lo que sí podemos afirmar, es que se cuenta con algoritmos que permiten llegar (quizás no en el tiempo ideal) a la solución ideal.

¡Gracias Totales!

Baiges Matias	59076	mbaiges@itba.edu.ar
Bilevich Andres	59108	abilevich@itba.edu.ar
Margossian Gabriel	59130	gmargossian@itba.edu.ar