(2024 2Q) 72.27 - Sistemas de Inteligencia Artificial - Comisión: S

Grupo 9

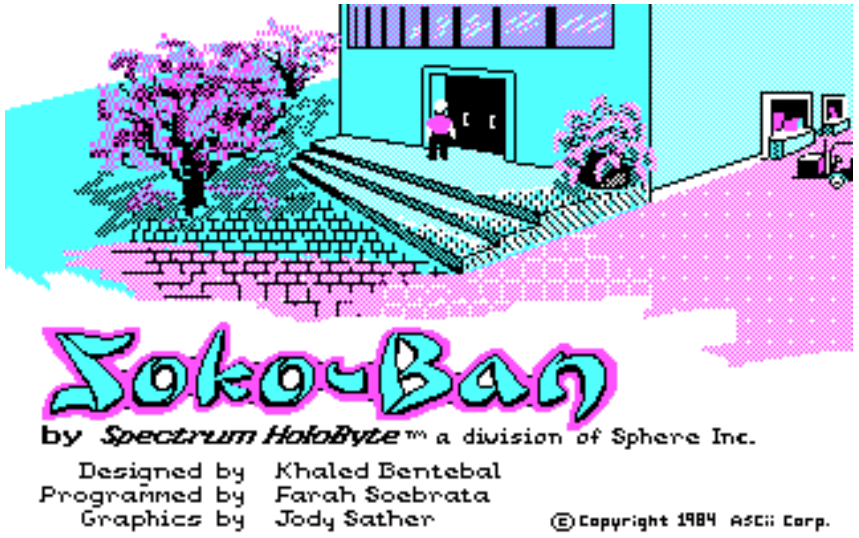
Gazzaneo Lautaro Nicolas

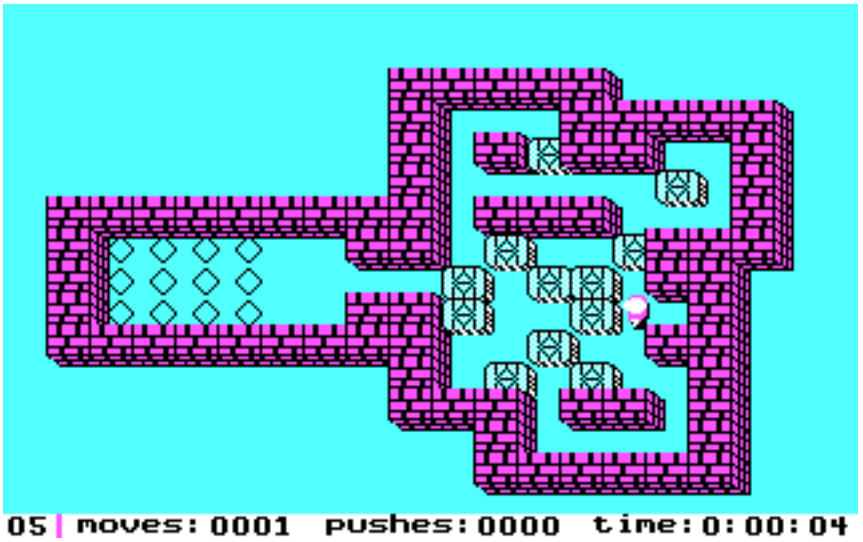
Pellegrini Jorge Orlando

Sinopluoglu Mustafa(no se pudo contactar)

倉庫番

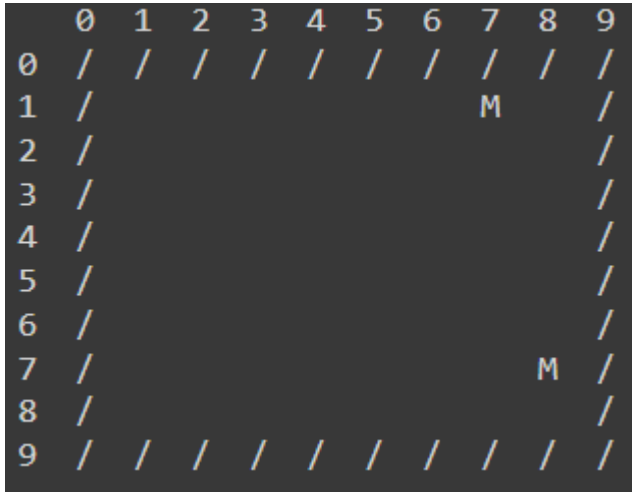
Encargado de almacén



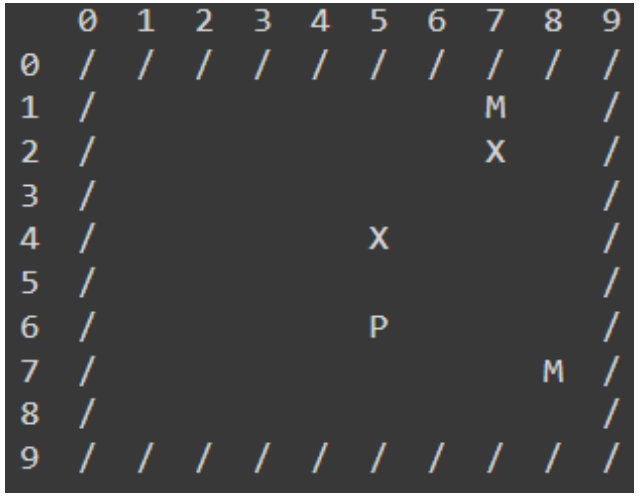


Representación del estado

Mapa Original

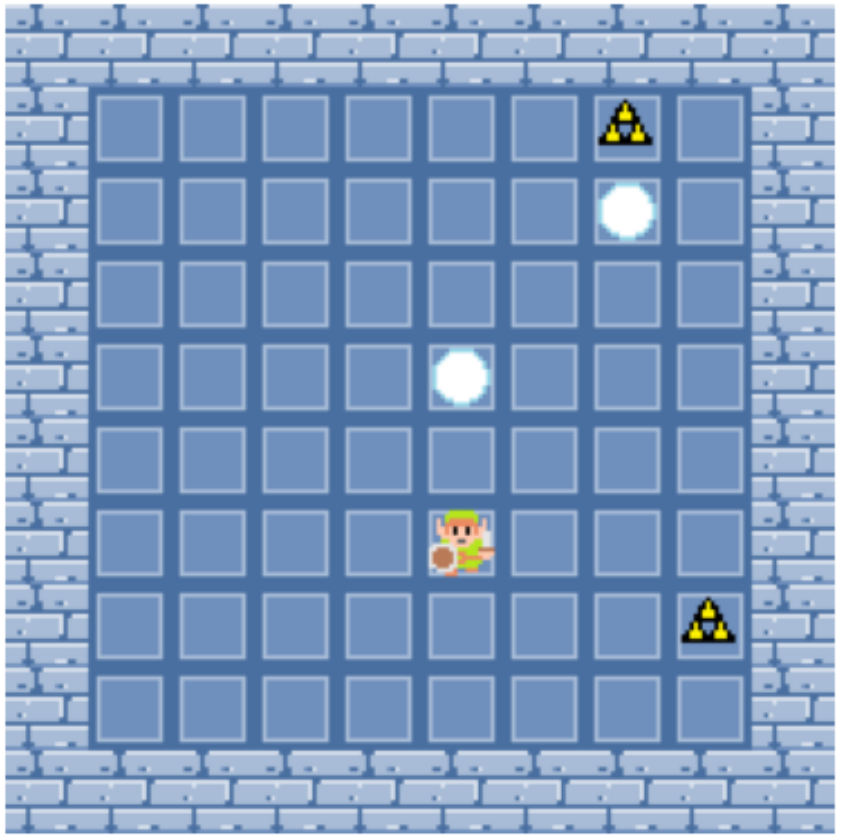


Mapa Símbolo

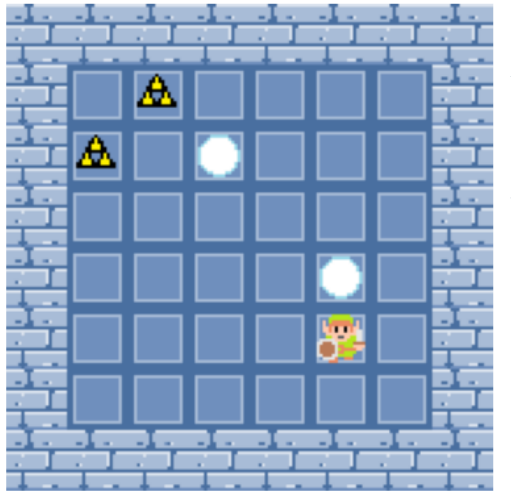


Representación gráfica del estado

```
   0 1 2 3 4 5 6 7 8 9
0  / / / / / / / / / /
1  /               M /
2  /               X /
3  /                 /
4  /         X       /
5  /                 /
6  /         P       /
7  /               M /
8  /                 /
9  / / / / / / / / / /
```

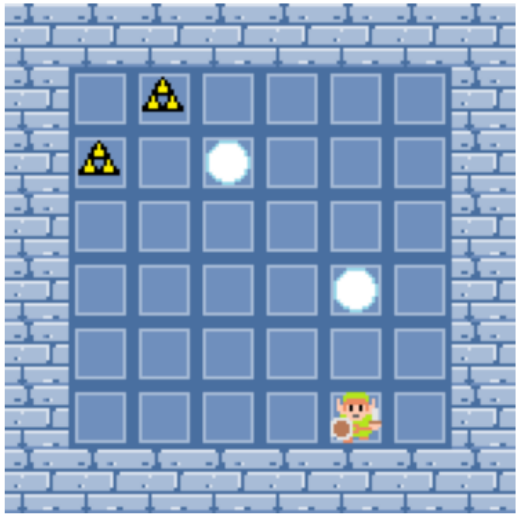


Acciones: siempre puede ejecutarse los 4 input

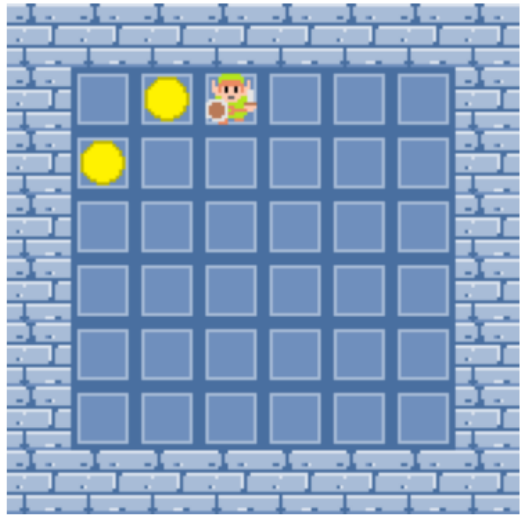


Se mueve el personaje y la caja

No se mueve el personaje

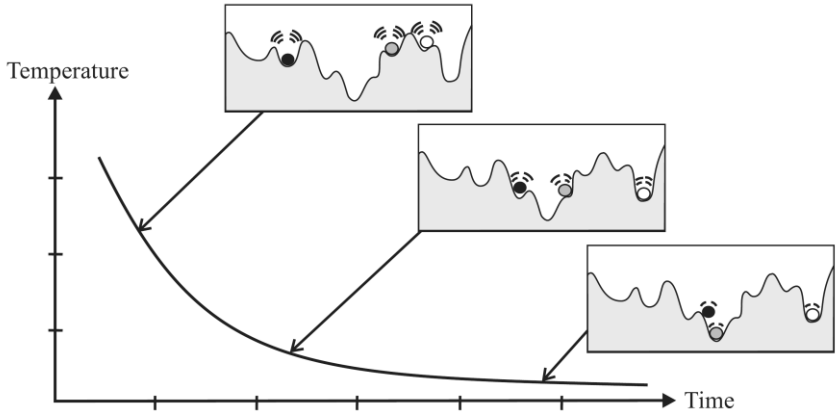


No se mueve el personaje

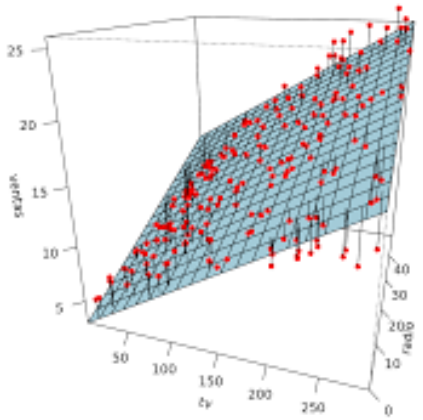


Condición ganadora: Posición cajas = posición marcas

Simulated annealing



Fuente:Ledesma, S., Ruiz, J., & Garcia, G. (2012). Simulated annealing evolution. Simulated Annealing-Advances, Applications and Hybridizations, 210-218.



Original:

$$P = p_0 e^{\frac{\Delta E}{T}}$$

ΔE : *Variación de la función objetivo*

T : *f*(iteraciones)

Se toma un movimiento al azar, se acepta si $\Delta E < 0$ o con una probabilidad *P*

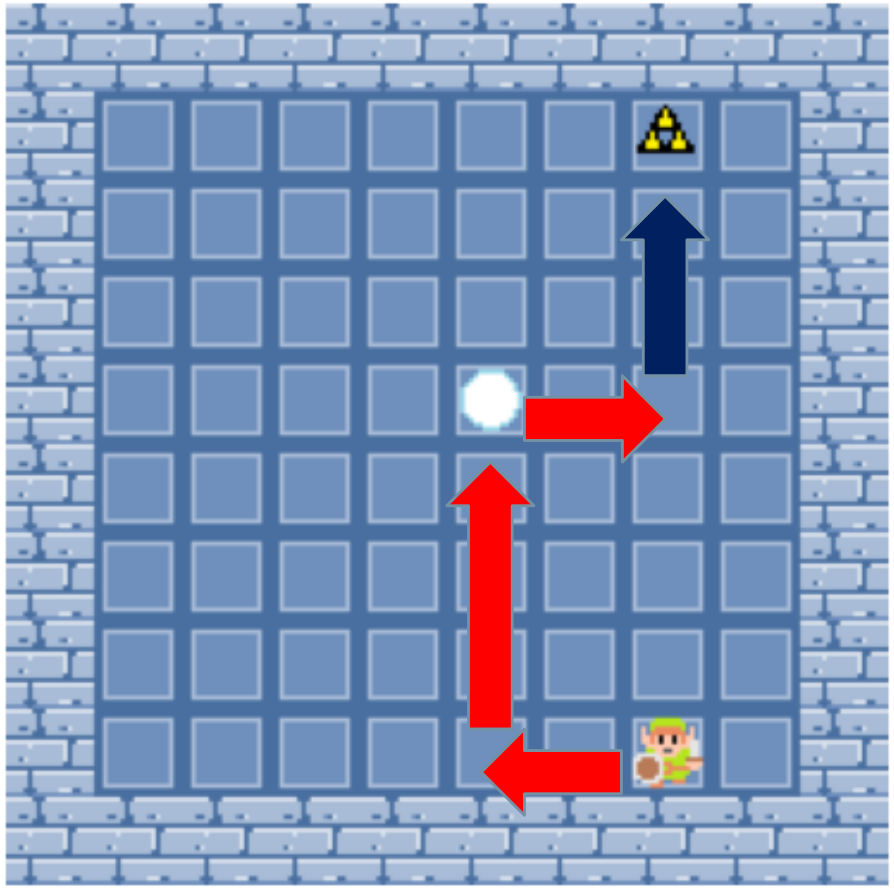
Propuesto:

$$P = p_0 e^{\frac{i}{tau}}$$

Se toma acepta siempre el movimiento que tiene el menor valor de la función objetivo. Con una probabilidad P se acepta el movimiento mas adverso (que mas aumenta la función objetivo o el que menos la disminuye).

Simulated Annealing - Función Objetivo

-Su minimización o maximización debe cumplir la condición ganadora
-Es preferible que el sistema refleje el grado de avance en la resolución del problema para que el agente disponga de la información necesaria encontrar la solución.

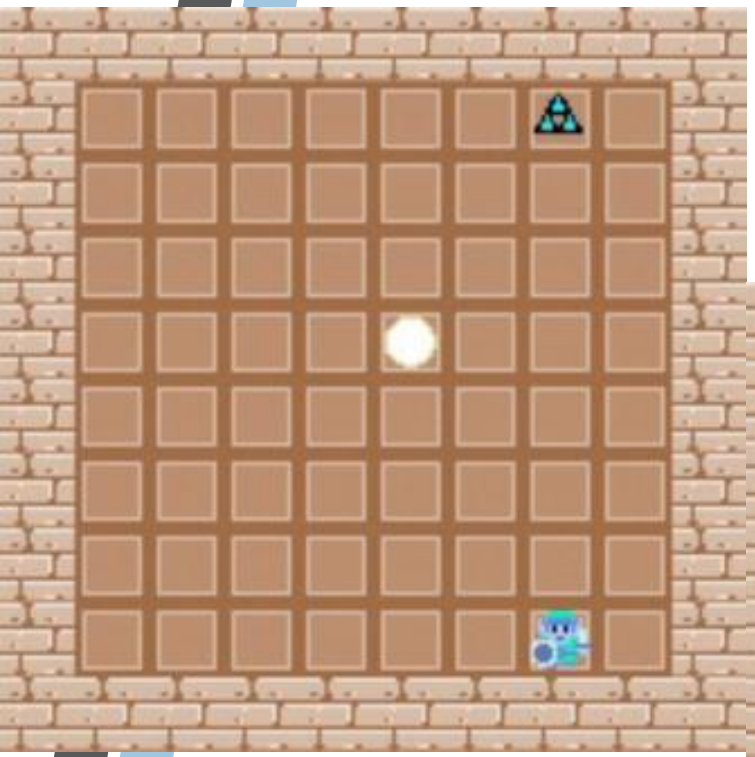


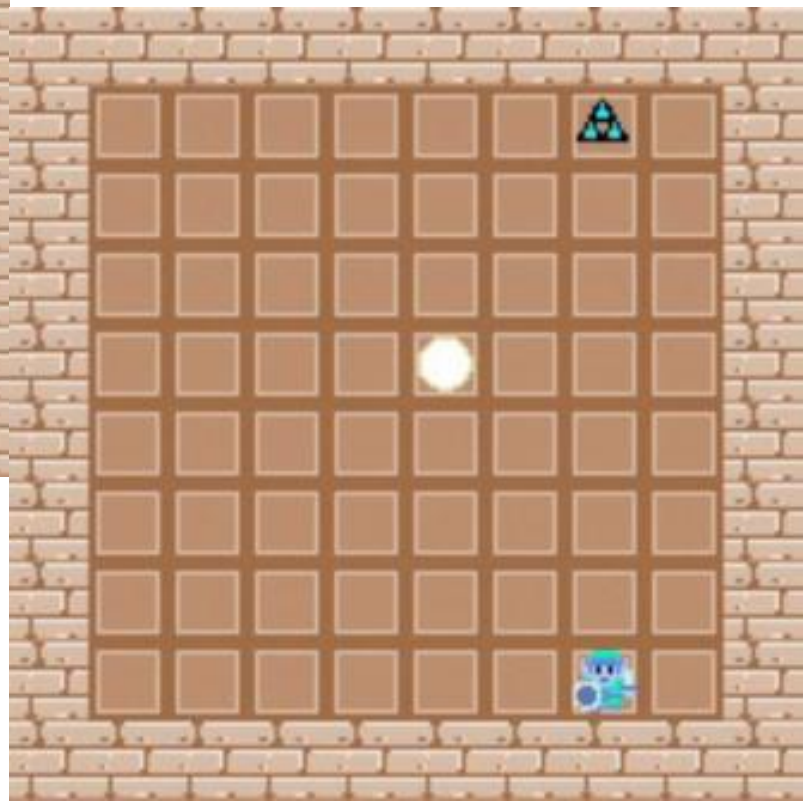
Función Objetivo

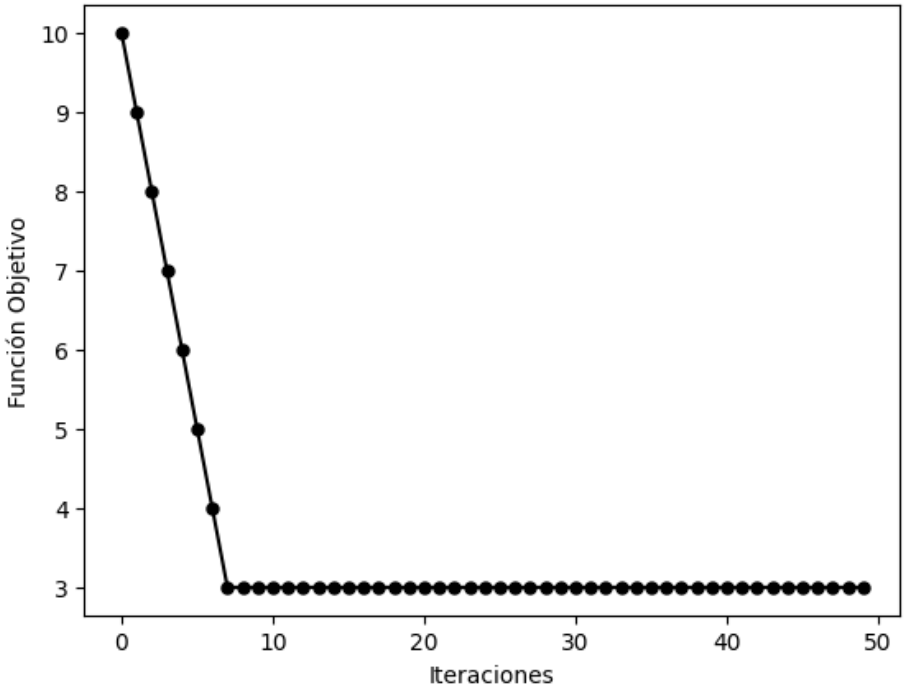
- Distancia manhattan Personaje-Caja
- Distancia manhattan Caja-Marca

$$r_i = |x - x_i| + |y - y_i|$$

Si solo se minimiza la función objetivo:

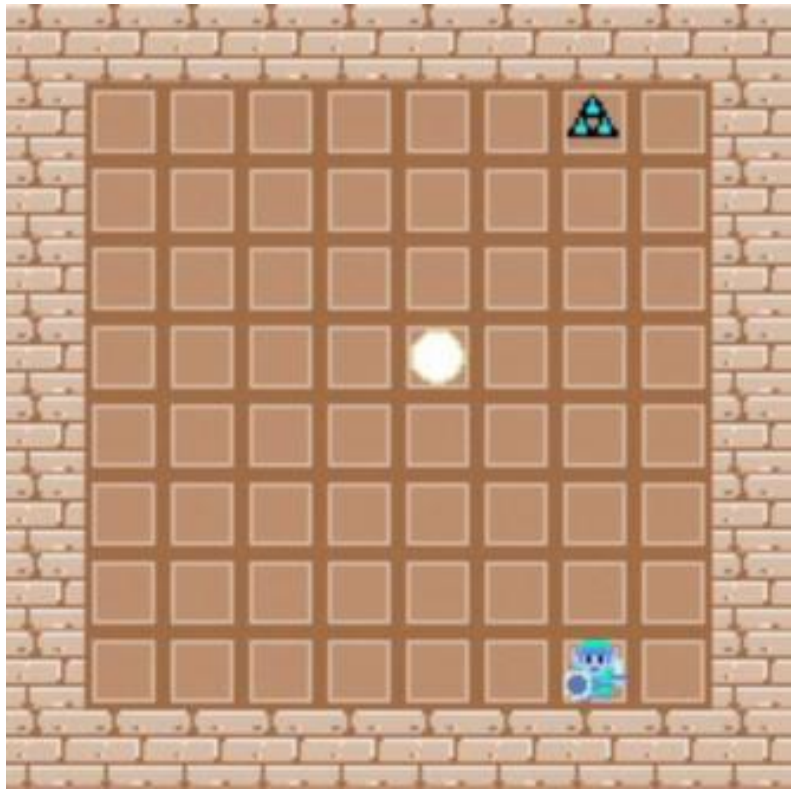


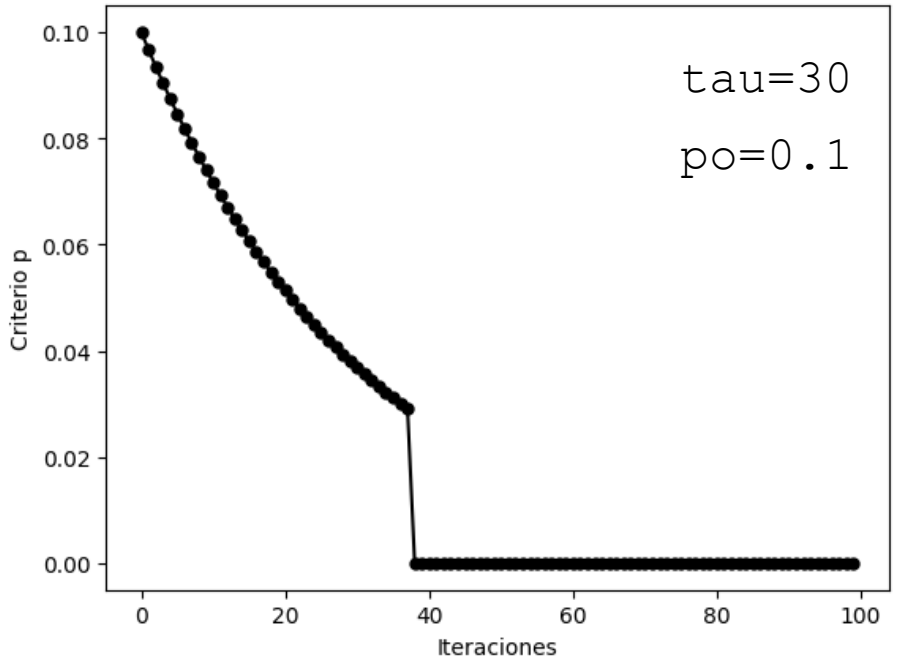


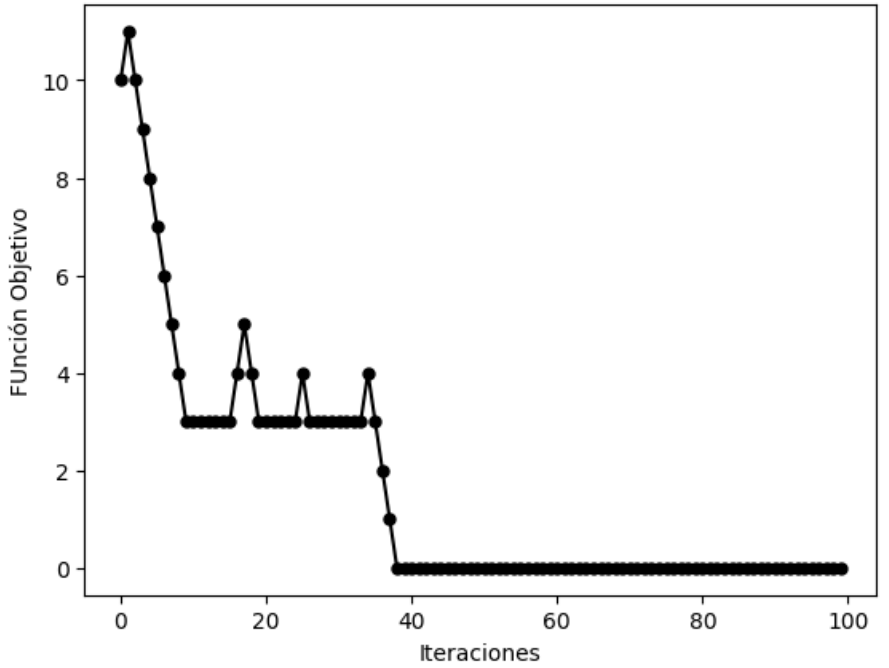


Implementación del método

0.5384 segundos

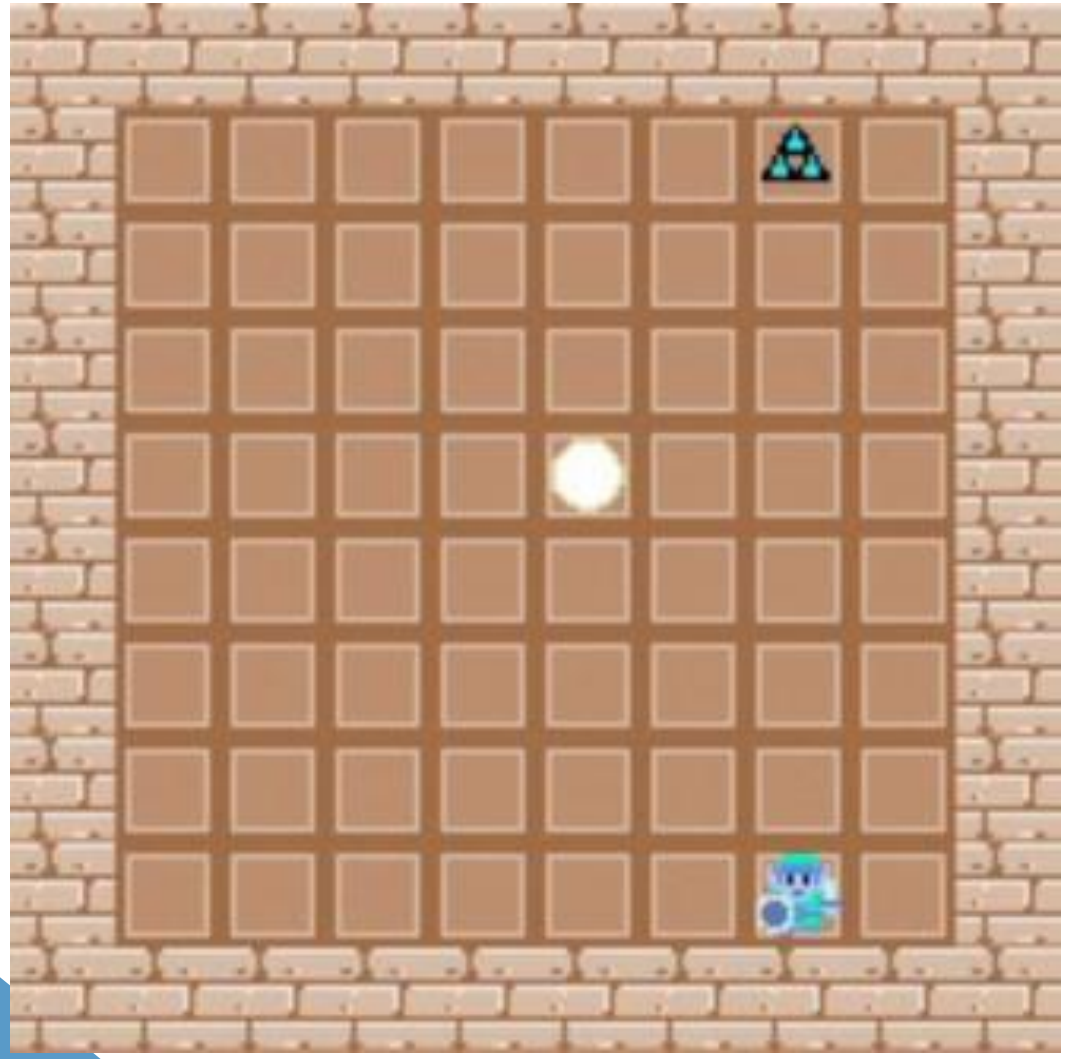


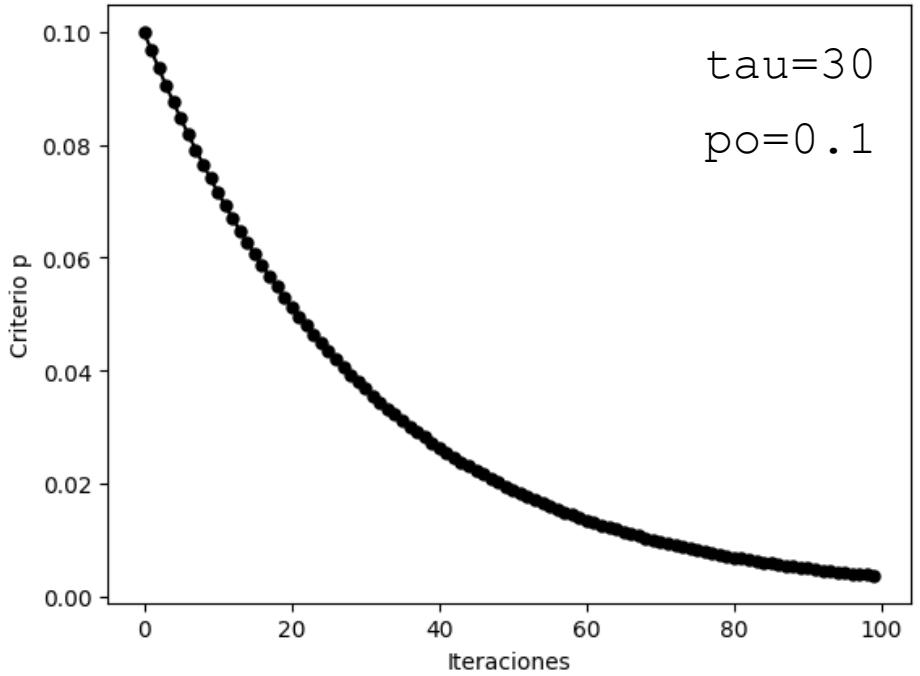


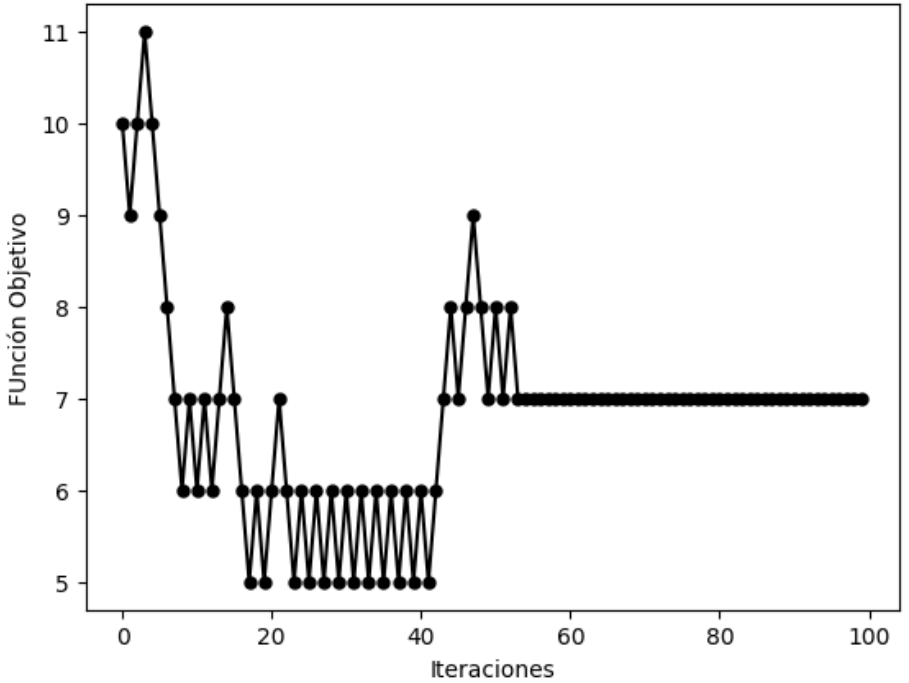


Intento fallido

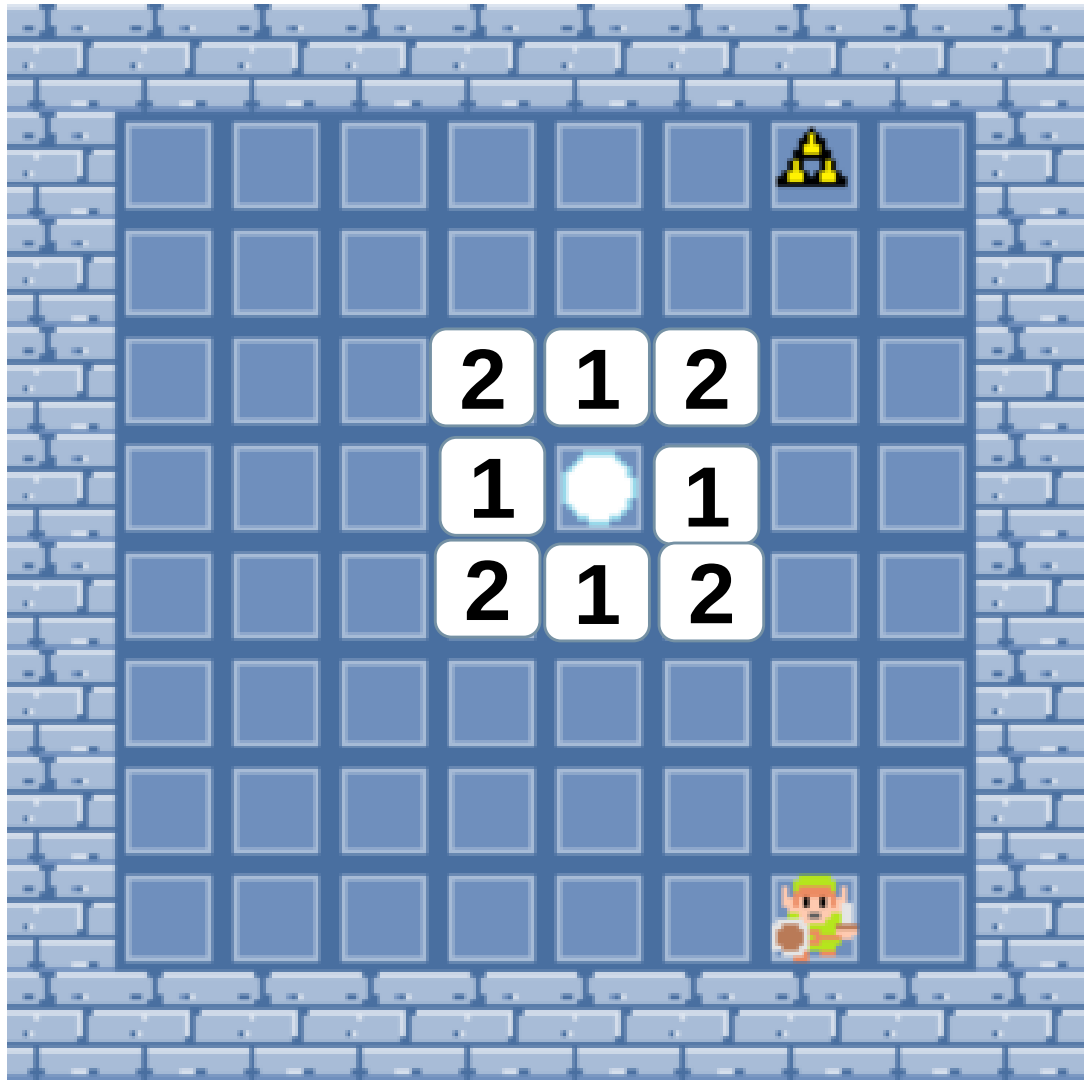
0.9474 segundos

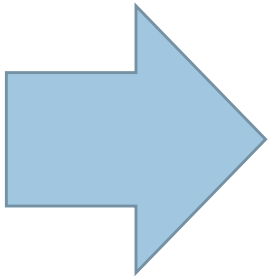


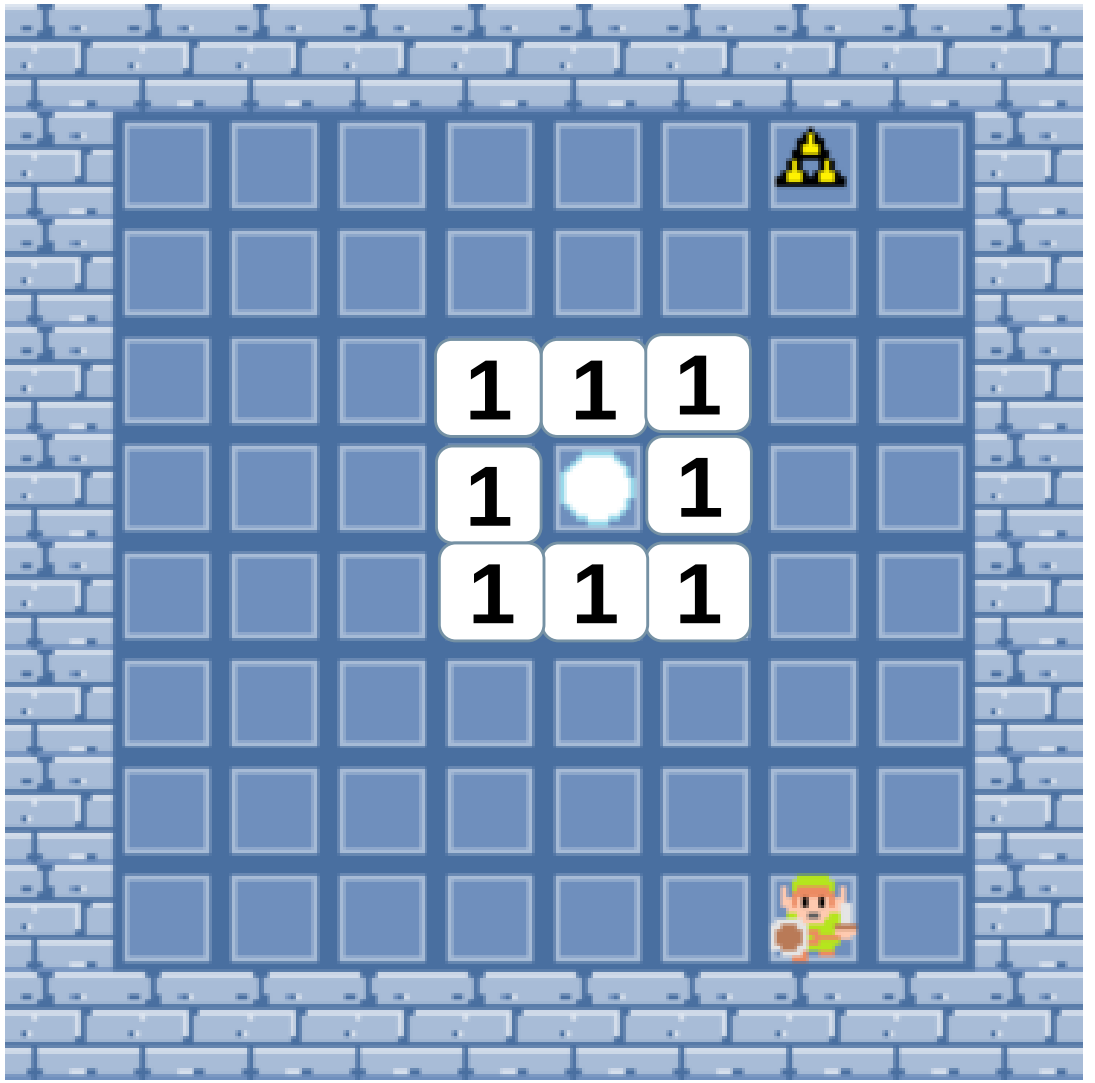




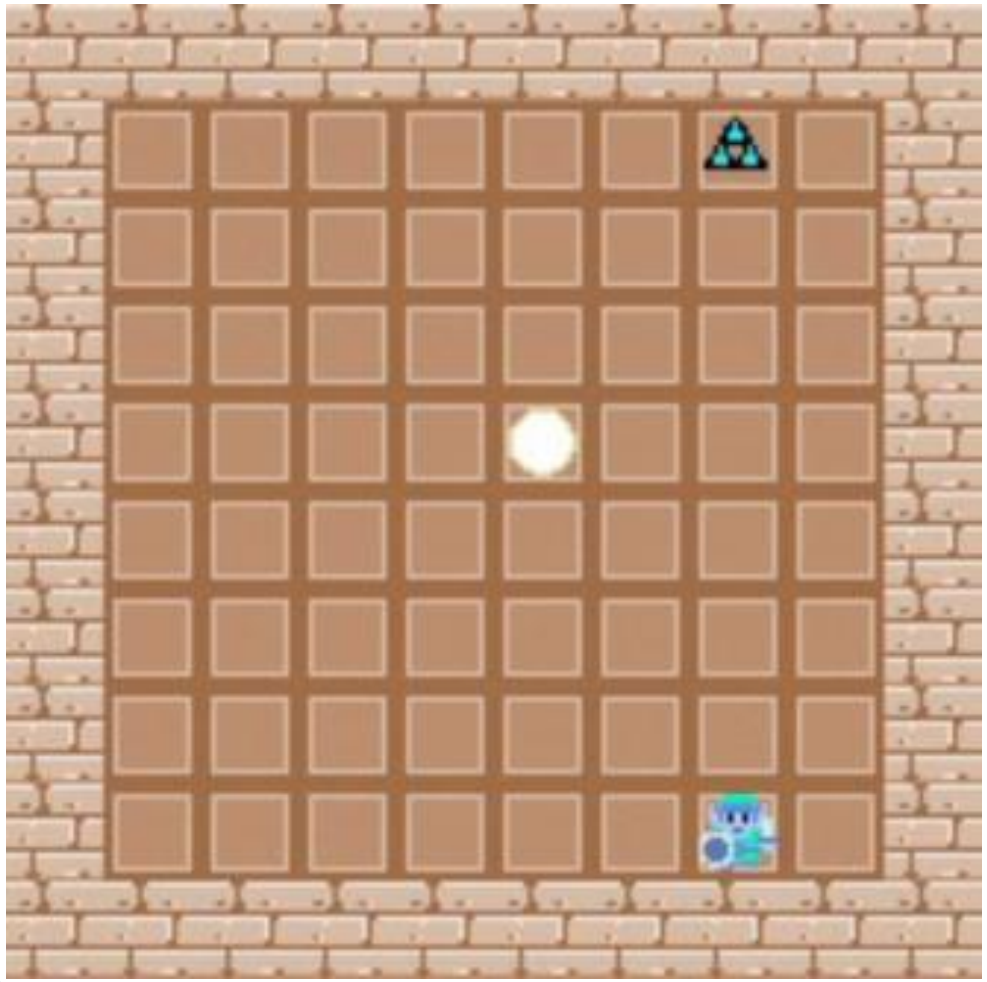
Nuevo criterio: diagonal cero

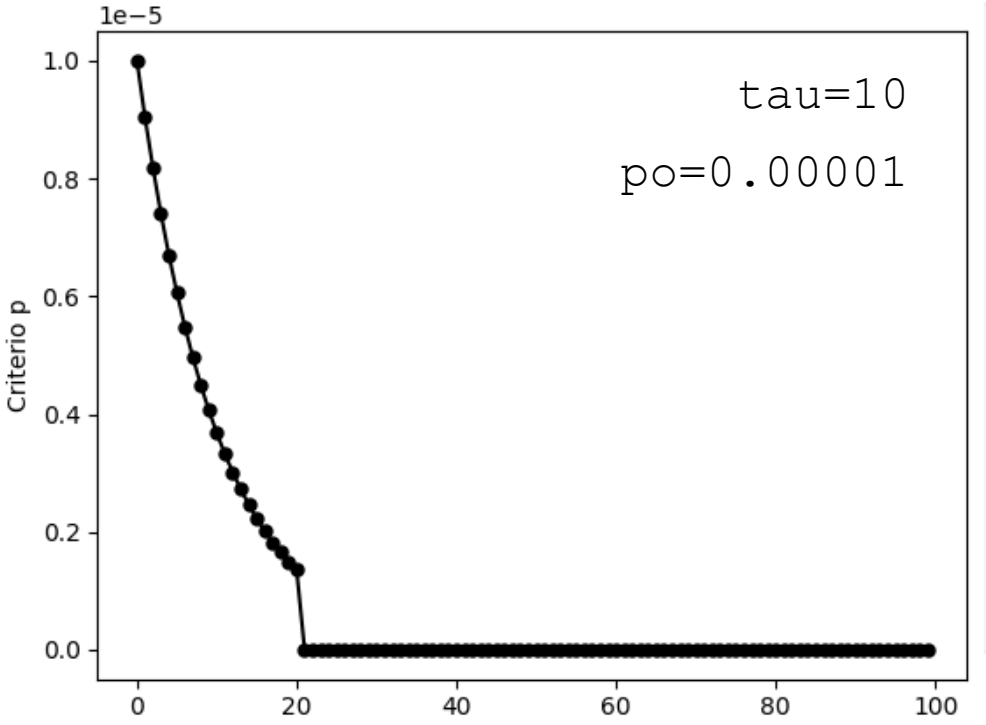


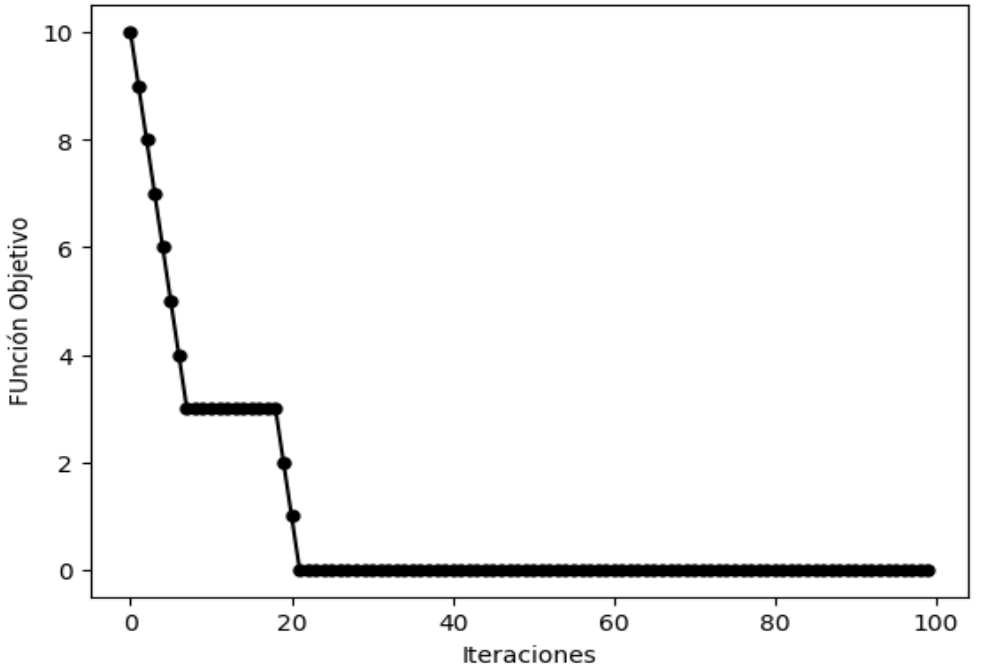




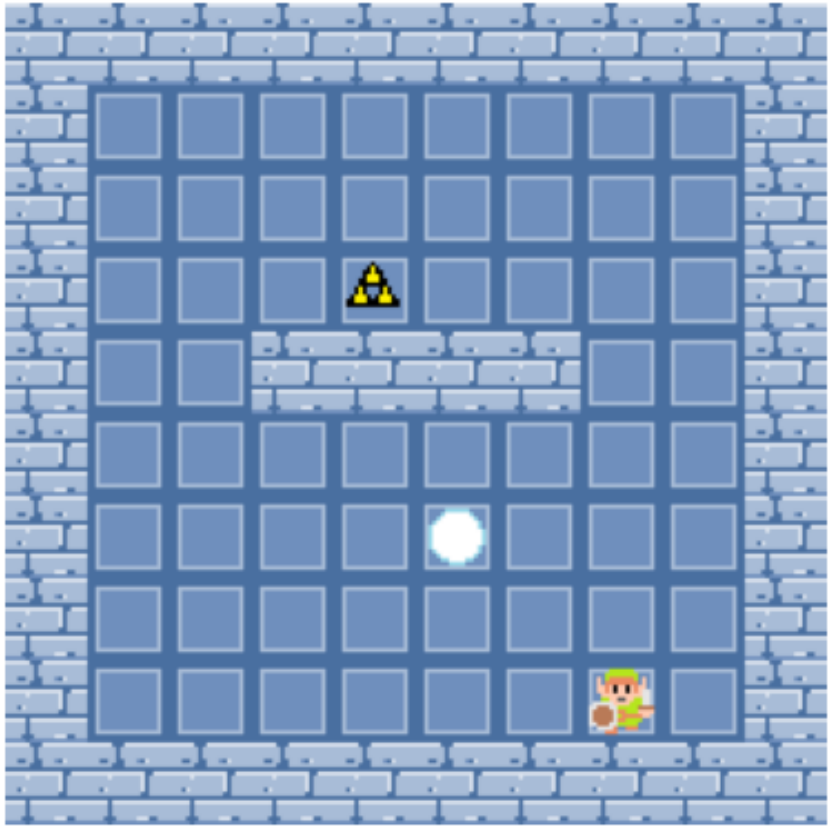
Implementación de diagonal cero

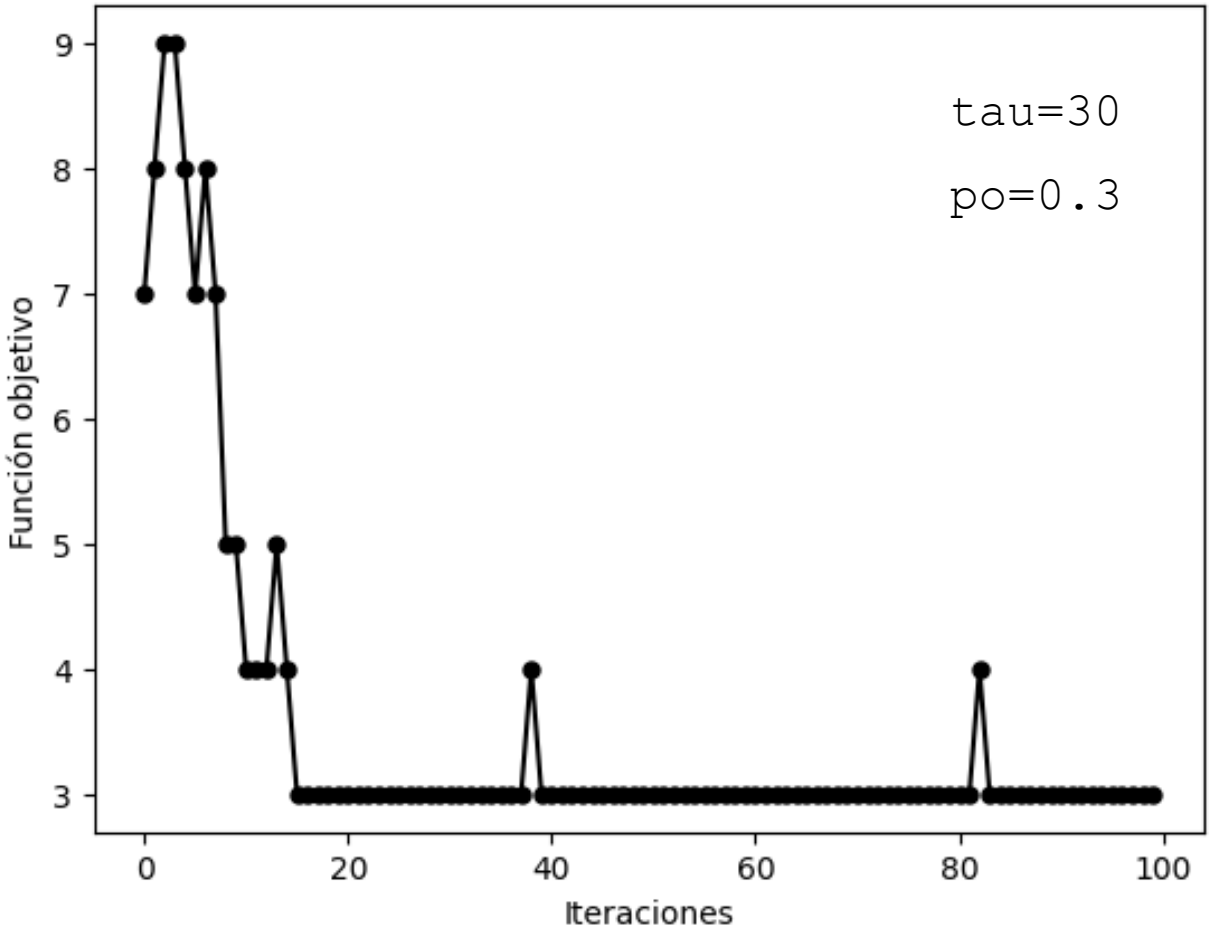




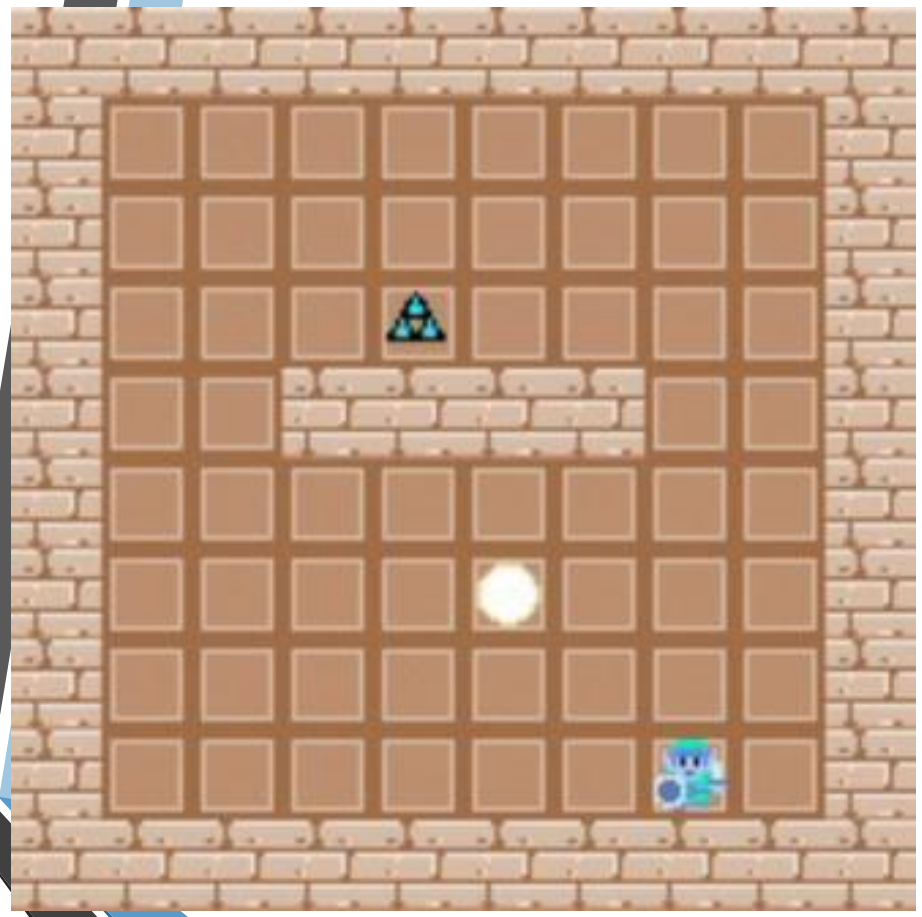


Nuevo mapa



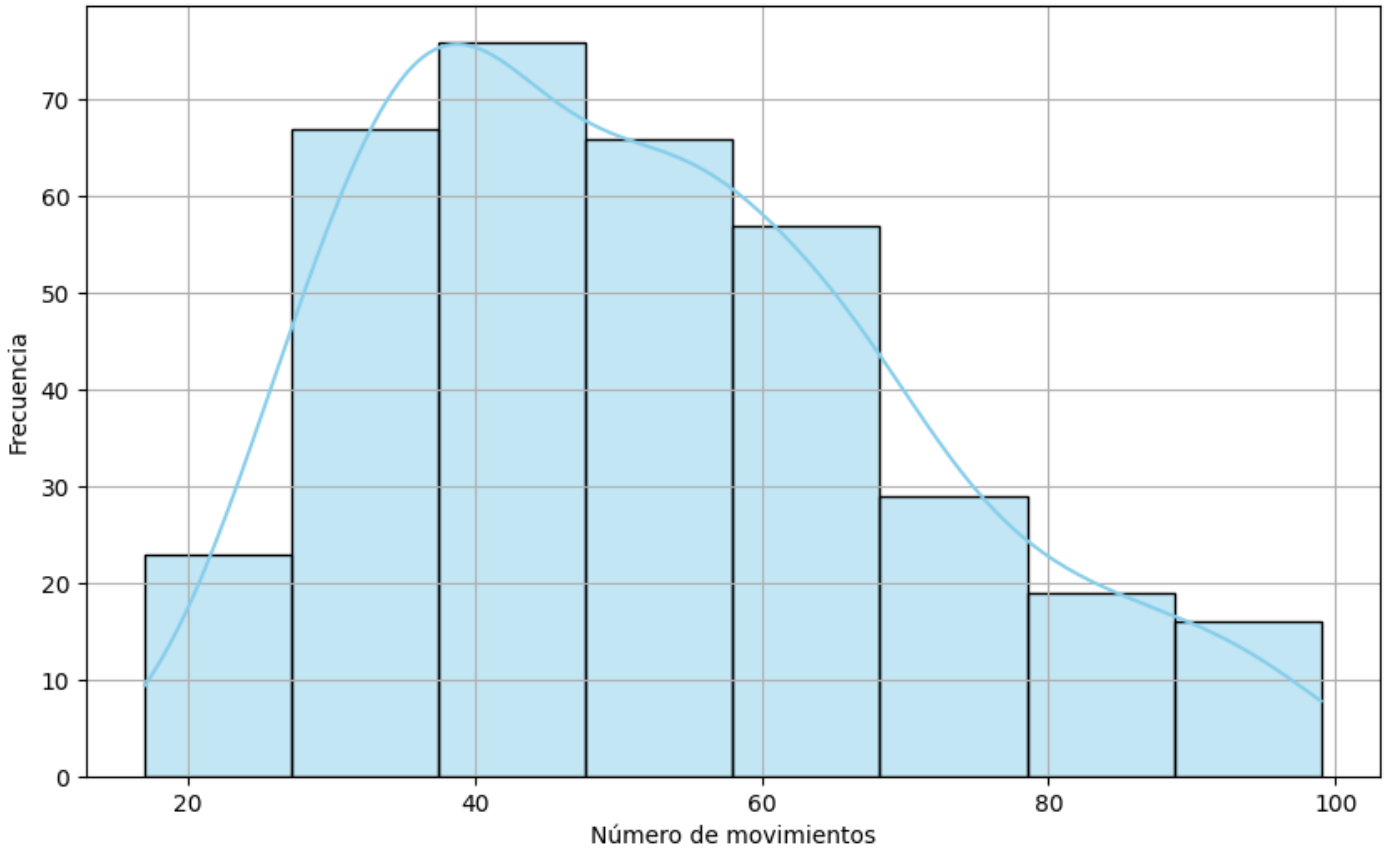

tau=30
po=0.3
Función objetivo
Iteraciones
9
8
7
6
5
4
3
0
20
40
60
80
100

Proceso iterativo



Comparación de diferentes parámetros

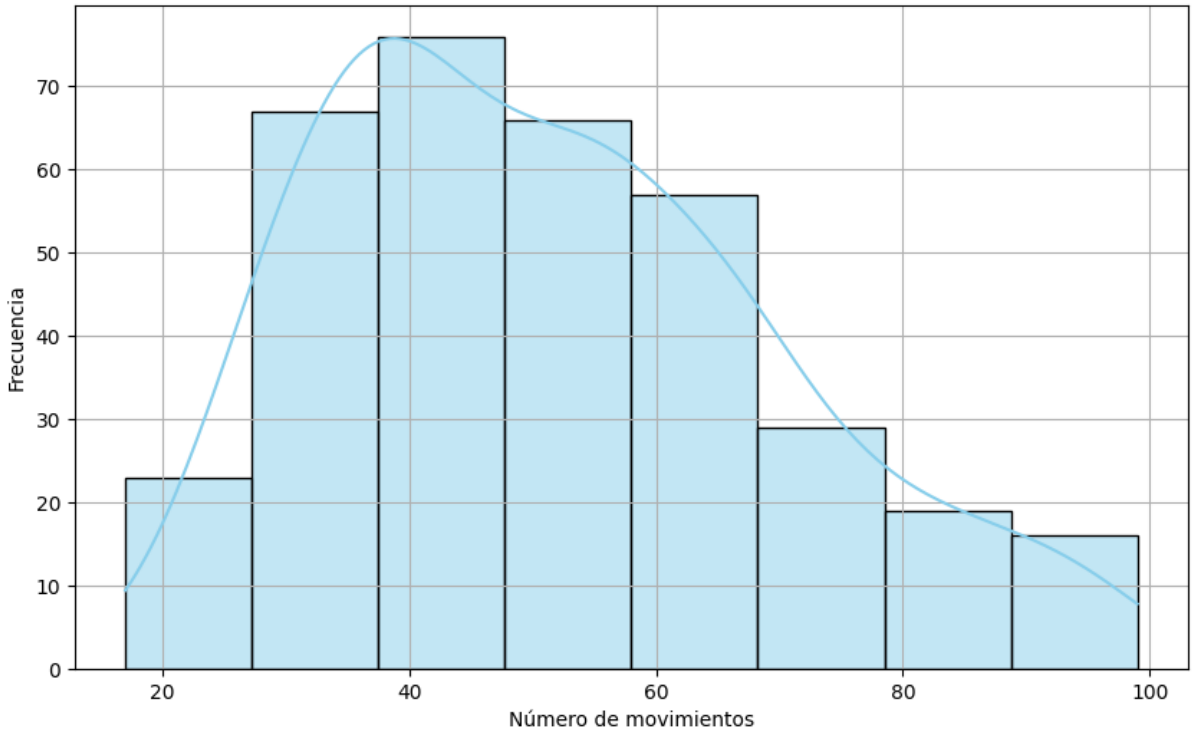
	po	tau	intentos	éxitos	Tasa de éxito	tiempo
Caso 1	0.3	30	5000	353	7%	3m3s
Caso 2	0.3	30	50	4	8%	2s
Caso 3	0.1	30	5000	74	1,5%	3m30s



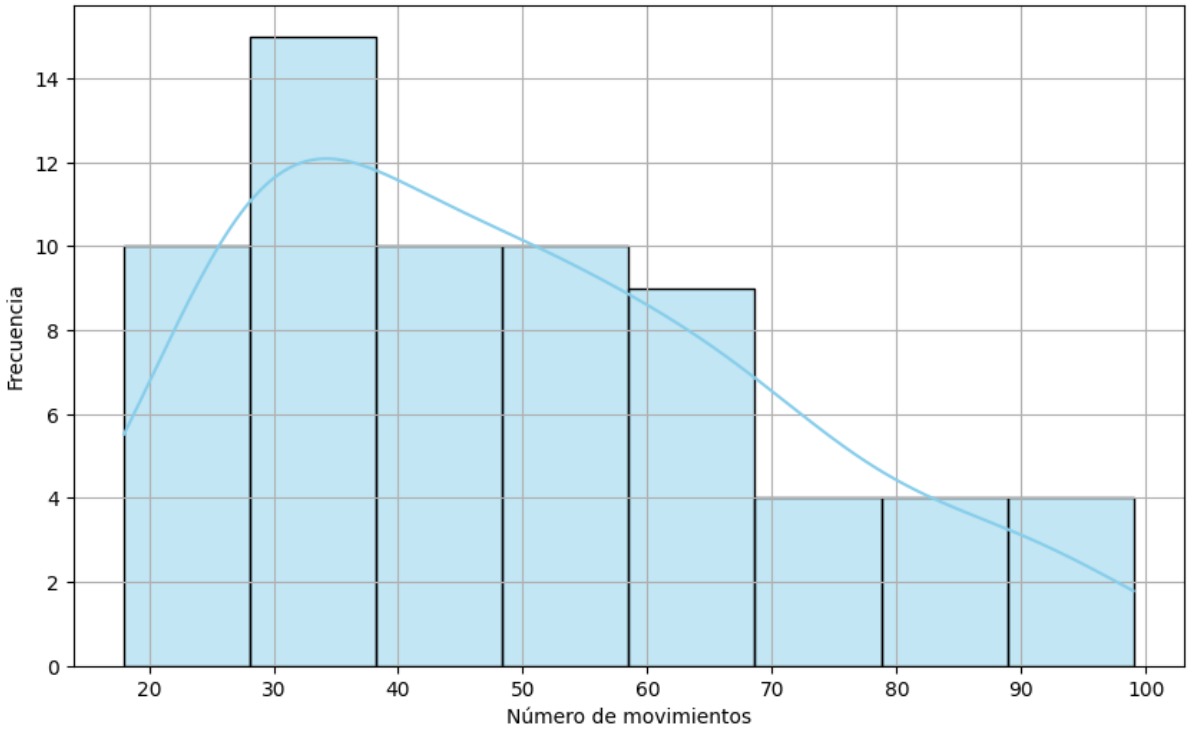
Histograma de éxitos. Caso 1 con diagonal cero

Comparación de diferentes parámetros

	po	tau	intentos	éxitos	Tasa de éxito	tiempo
Caso 1 con do	0.3	30	5000	353	7%	3m aprox.
Caso 1 sin do	0.3	30	5000	66	1,3%	3m aprox.



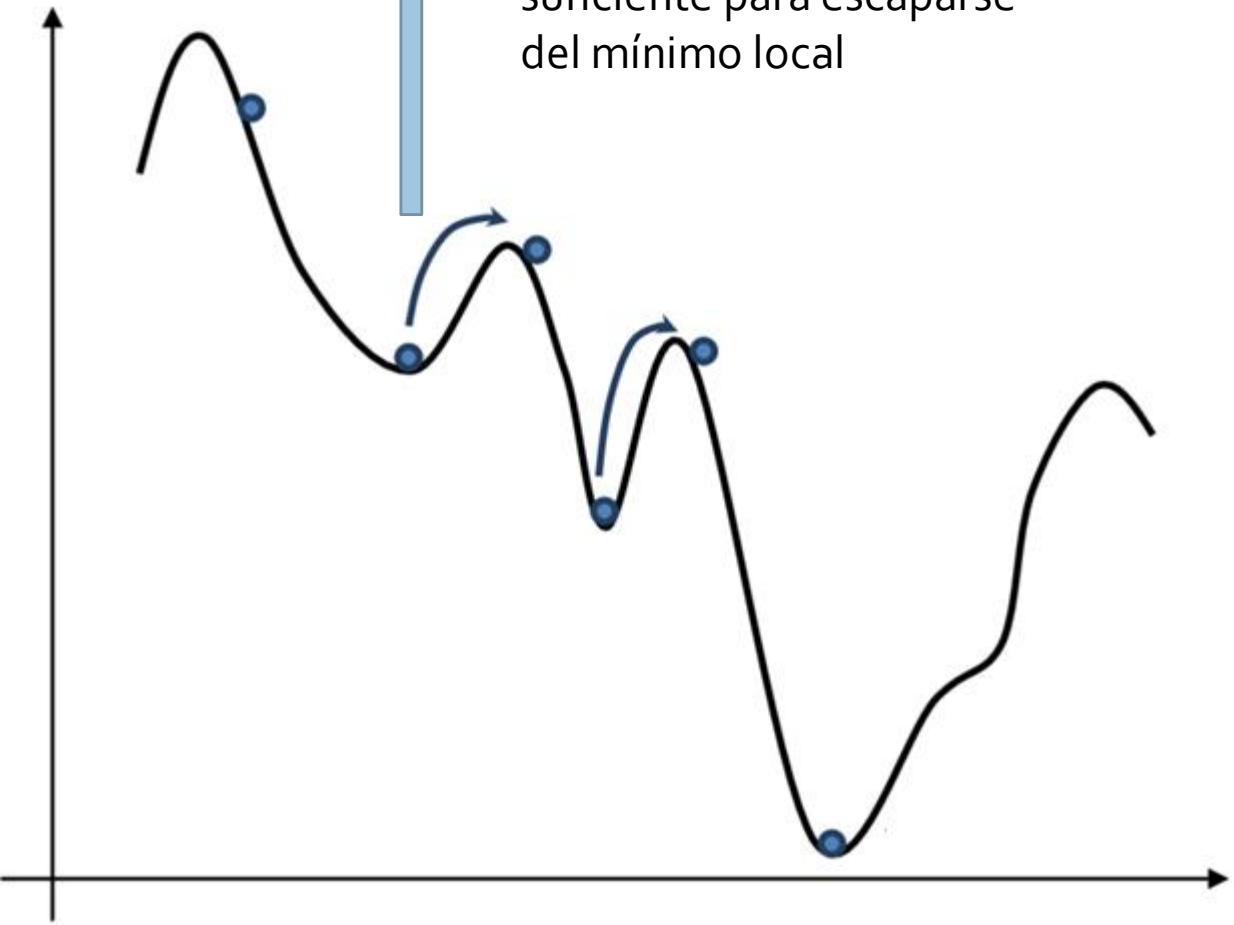
Histograma de éxitos. Caso 1 con diagonal cero



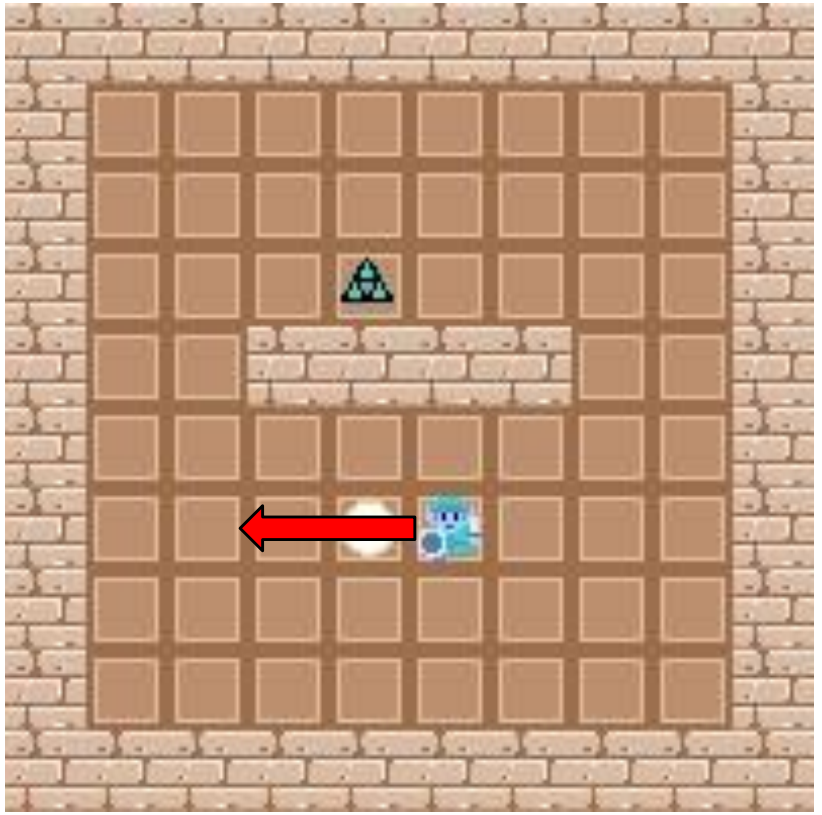
Histograma de éxitos. Caso 1 sin diagonal cero

Propuesta. No se implementó

Salto ineficiente, el peso de equivocarse no es suficiente para escaparse del mínimo local

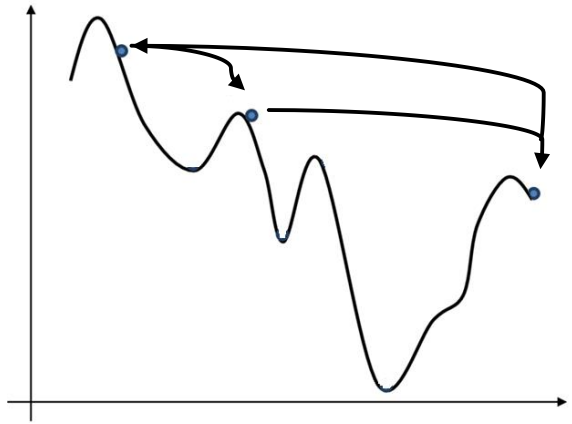


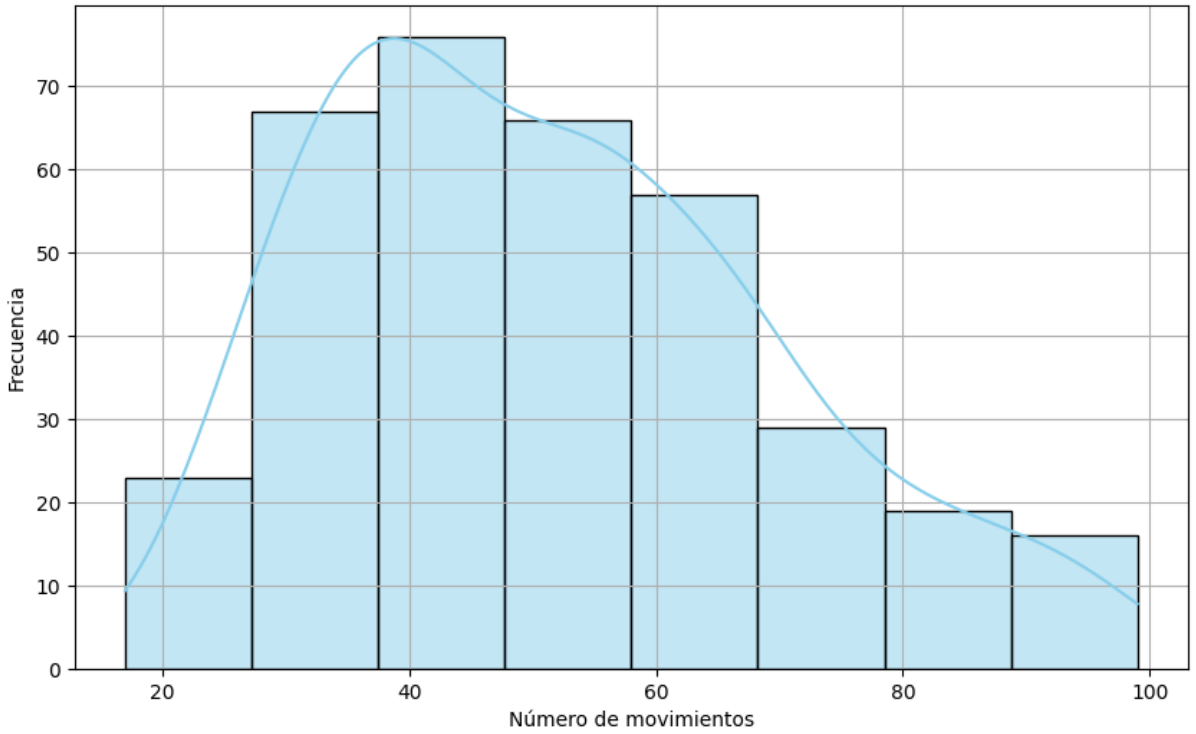
Solución, si se acepta un movimiento incorrecto, que corresponda a mas de un movimiento(se aumenta la energía de la partícula)



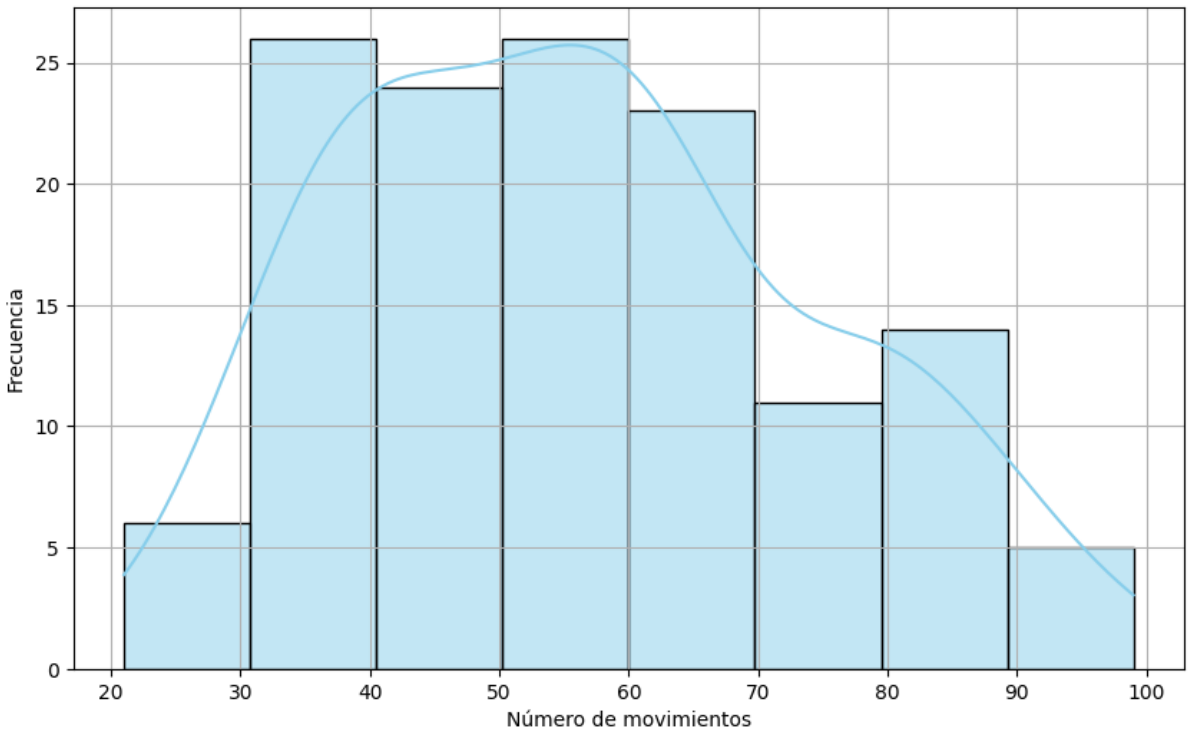
Comparación de diferentes parámetros

	po	tau	intentos	éxitos	Tasa de éxito	tiempo
Caso 1 con do	0.3	30	5000	353	7%	3m aprox.
Caso 1 con do	0.6	30	5000	135	2,7%	3m aprox.



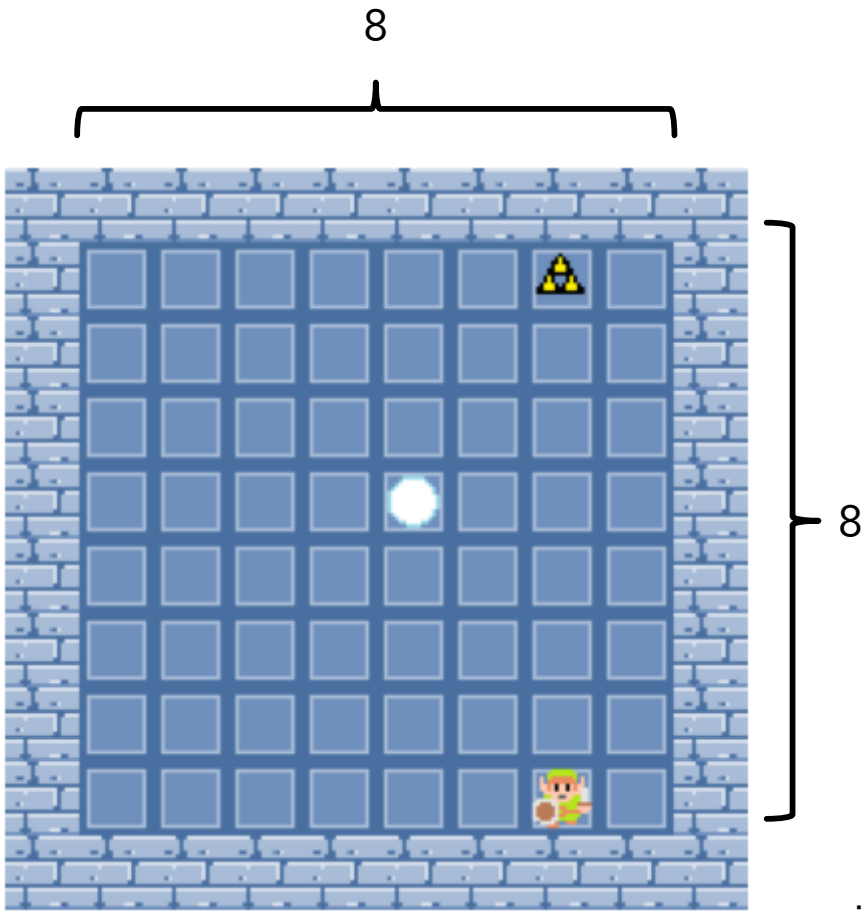


Histograma de éxitos. Caso 1 con do y po=0.3



Histograma de éxitos. Caso 1 con do y po=0.6

BFS



64 celdas=> 64x63=4032 posibilidades

```
set: visited

(4032 items) {(('/', '/', '/', '/', '/', ...), ('/', 'P', ' ', '
```

Condición ganadora
Índices cajas = índices marcas

Estado:
"Matriz símbolo"

- No se duplica los estados.

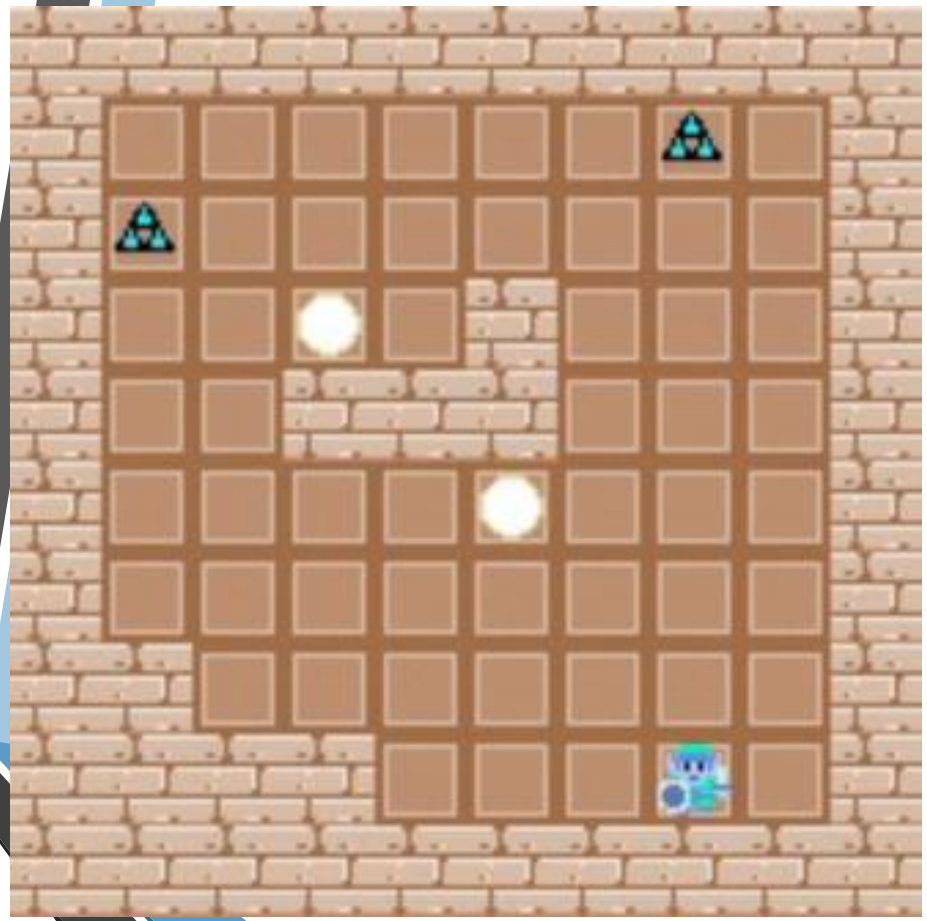
(First In, First Out)

implica que el estado alcanzado primero es el menos costoso

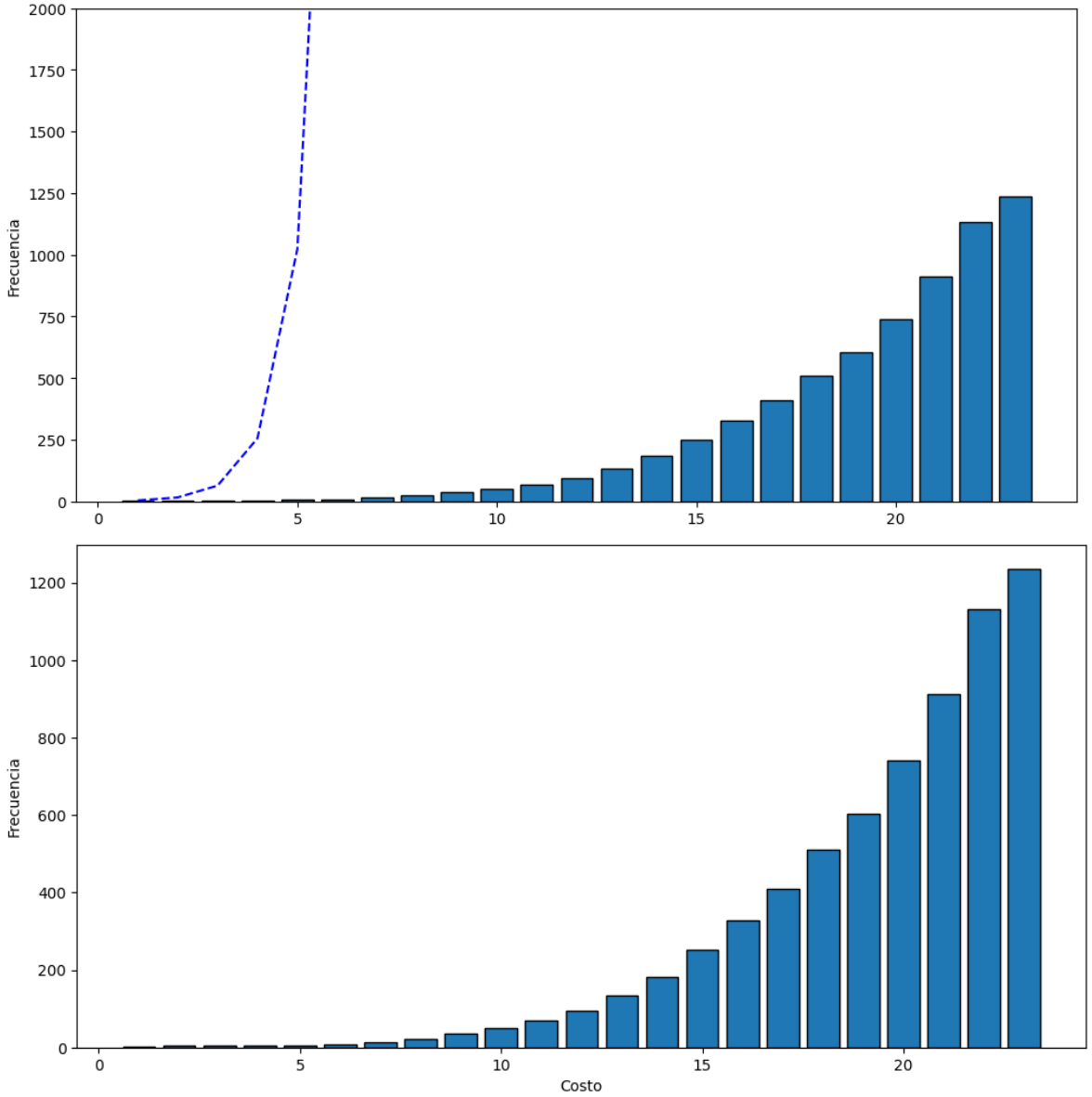
Tiempo: 3 segundos

23 movimientos

Completitud: encuentra solución si existe y el factor de ramificación es finito



Frecuencia de estados visitados en función del costo



Estados expandidos=5395

Estados totales=6760

Estados frontera=1365

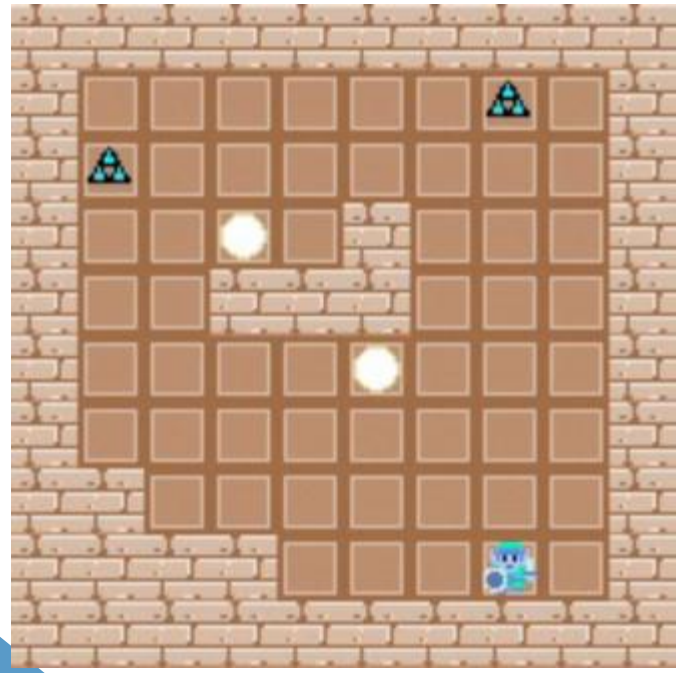
DFS

Estructura tipo pila

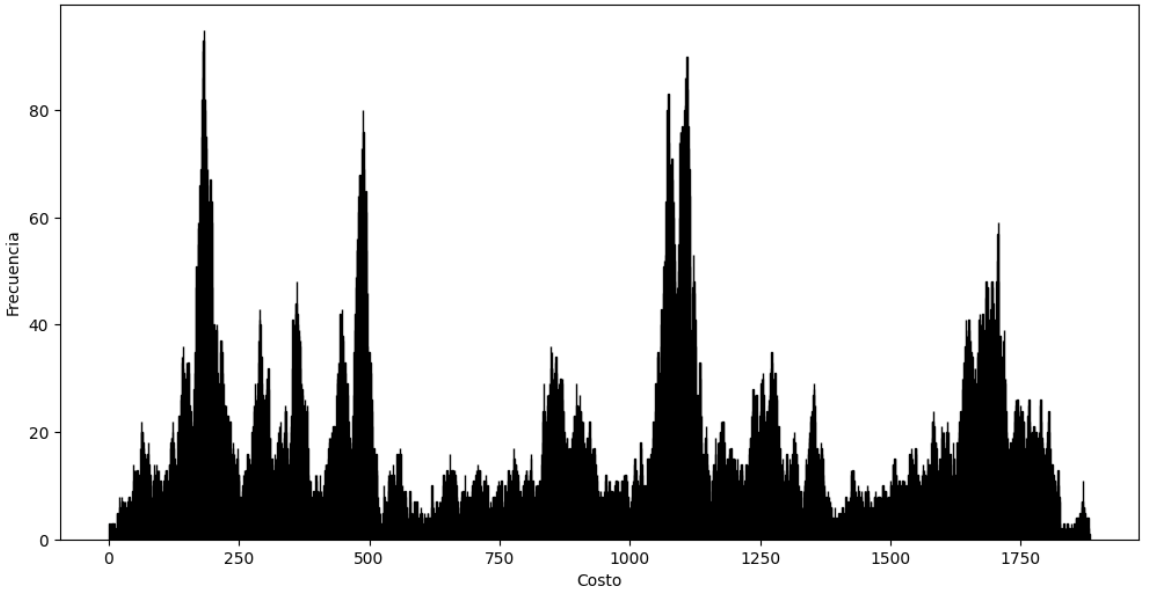
No garantiza el óptimo

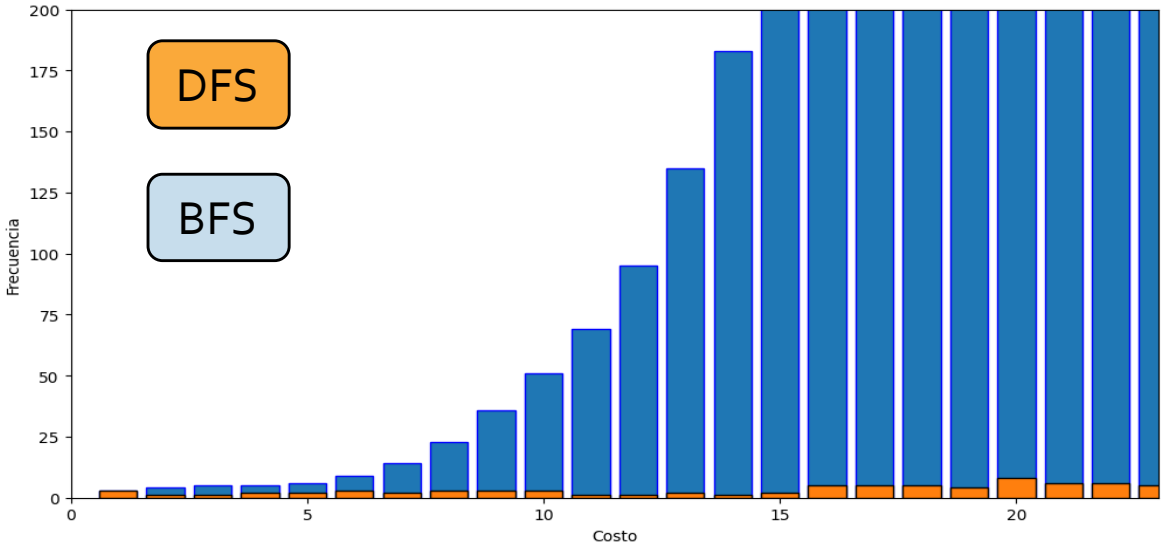
Tiempo: 27 segundos

Movimientos: 1873



Frecuencia de estados visitados en función del costo





Estados expandidos=33.371

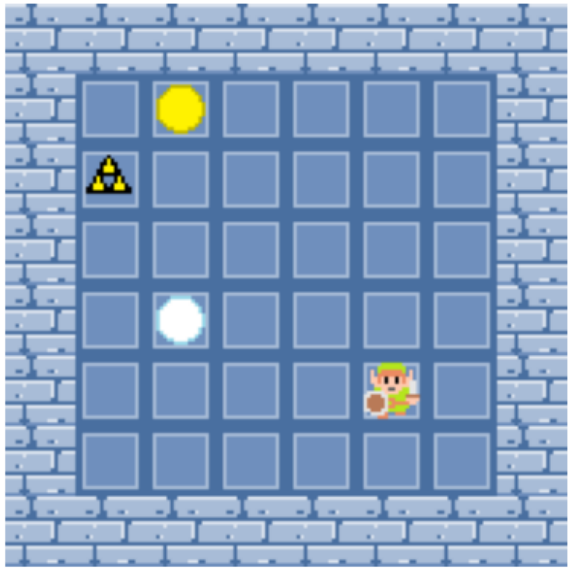
Estados totales=34.874

Estados frontera=1503

Heurísticas elegidas

(DM-PX) Distancia manhattan entre el personaje y la celda adyacente a la caja

$$h_i = \left|x_{caja\,i} - x_{personaje}\right| + \left|y_{caja\,i} - y_{personaje}\right| - 1$$



$h = 3$

Costo para acercarse a la caja=3

$h = 3$

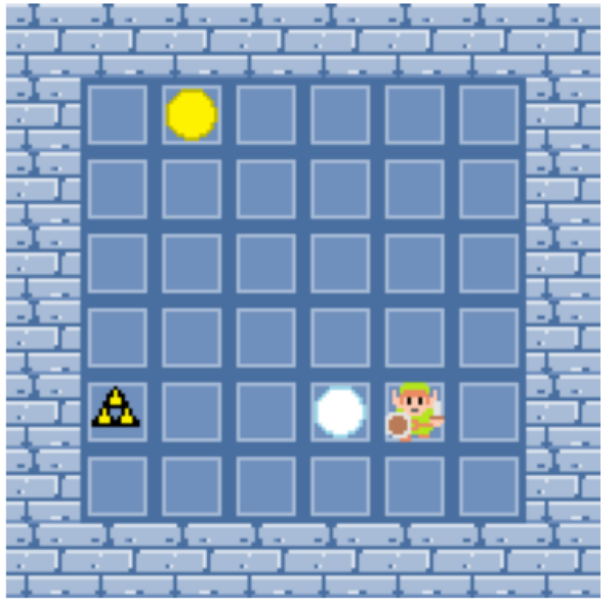
Costo para acercarse a la caja=5

Costo: cantidad de movimientos tomados

Heurísticas elegidas

(DM-PM) Distancia manhattan entre la caja y la marca

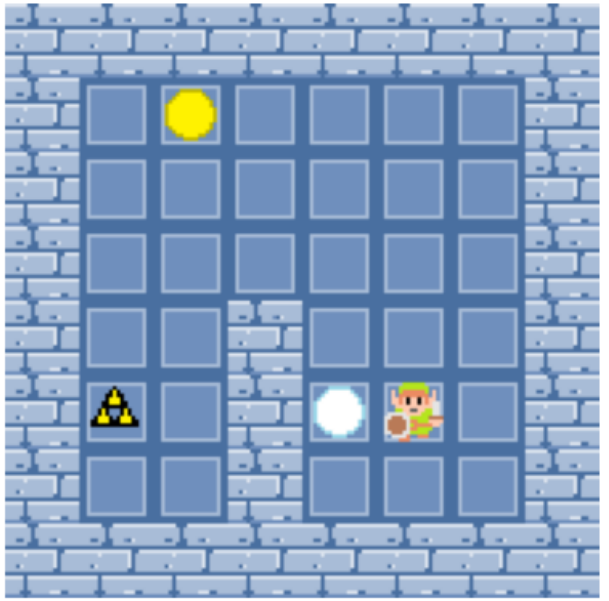
$$h_i = \left|x_{caja\ i} - x_{marca\ i}\right| + \left|y_{caja\ i} - y_{marca\ i}\right|$$



$h_1 = 0$

$h_2 = 3$

Costo =3



$h_1 = 0$

$h_2 = 3$

Costo =13

Costo: cantidad de movimientos tomados

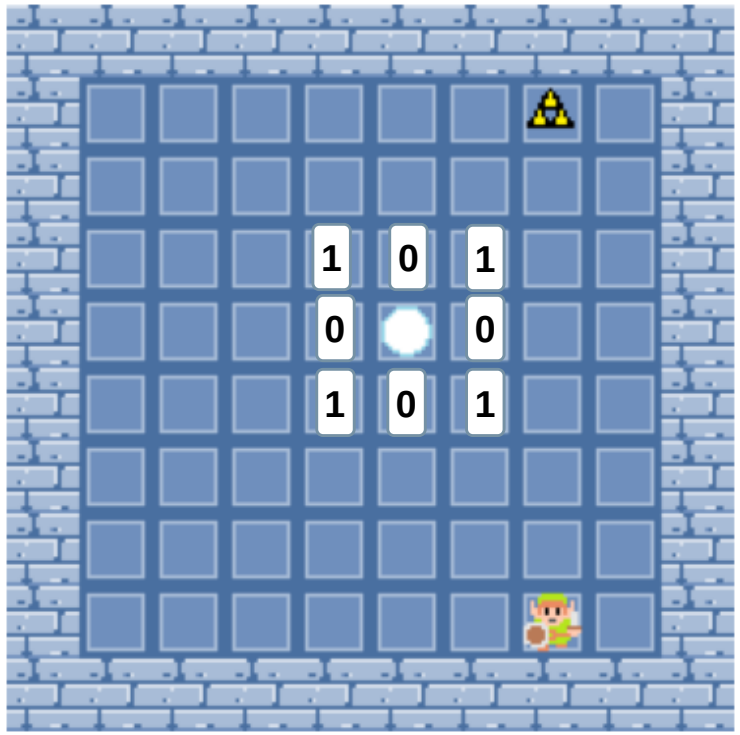
Heurísticas elegidas

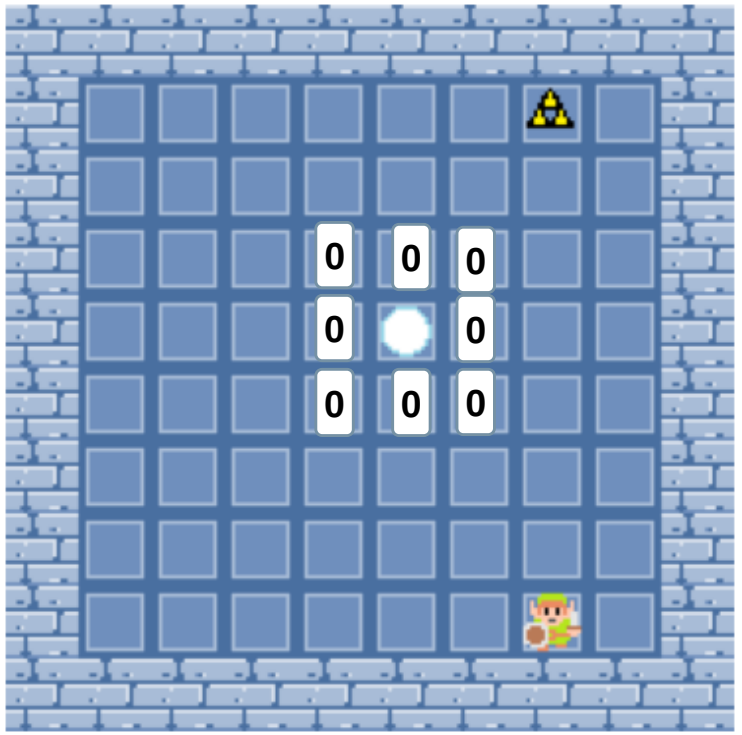
(DM-PX-do) Distancia manhattan entre el personaje y la celda circundante a la caja.

$$h = \begin{cases} 0 \\ |x_{caja\ i} - x_{personaje}| + |y_{caja\ i} - y_{personaje}| - 1 \end{cases}$$

$si\ |x_{caja\ i} - x_{personaje}| = |y_{caja\ i} - y_{personaje}| = 1$

en cualquier otro caso

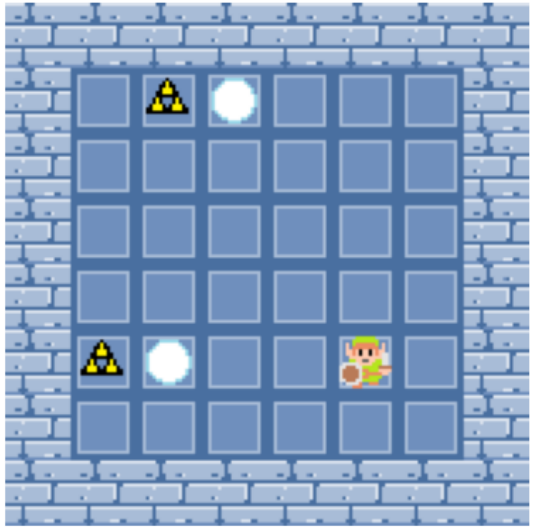




Heurísticas elegidas

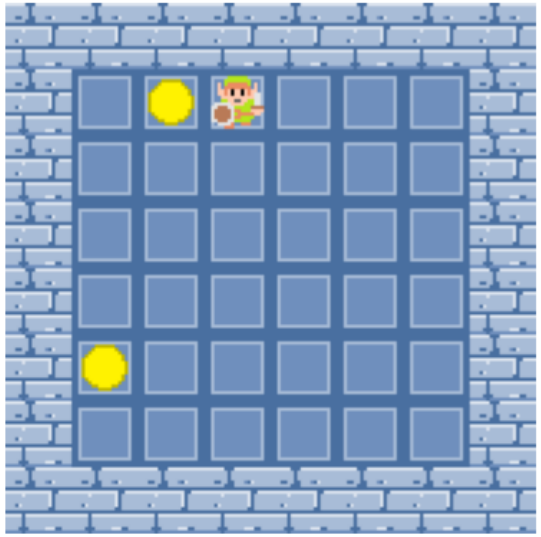
(conteoXM) Cantidad de cajas en la posición incorrecta

$$h = cantidad\ de\ cajas - cantidad\ de\ cajas\ activadas$$



h=2 Costo >2

h=1 Costo >1



h= 0 Condición ganadora

Costo: cantidad de movimientos tomados

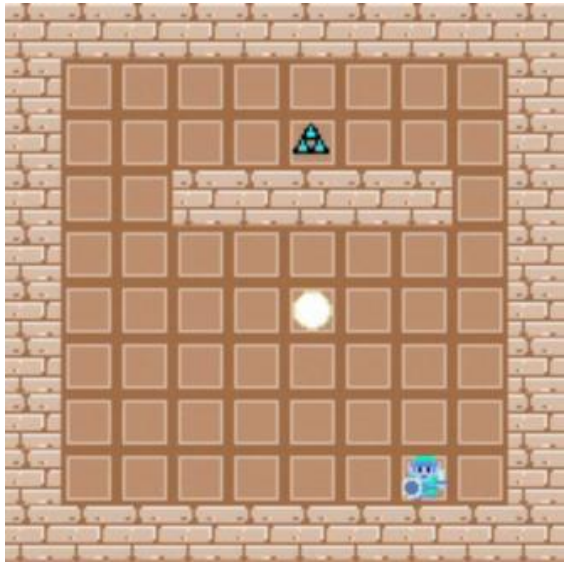
El tiempo de ejecución no es estable

Desvío del tiempo de ejecución de dos algoritmos

							Desvío [s]
Tiempo1 [s]	1.81	0.86	0.75	0.62	0.87	0.60	0.45
Tiempo 2 [s]	2.61	1.57	1.49	2	1.57	1.58	0.43

Código realizado en :

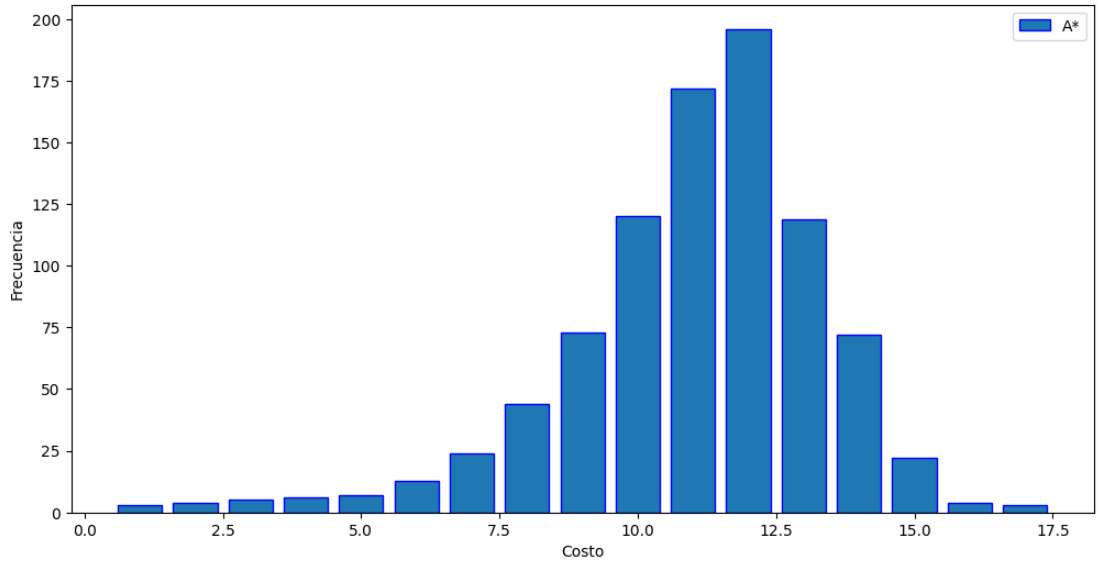
Heurística manhattan CM



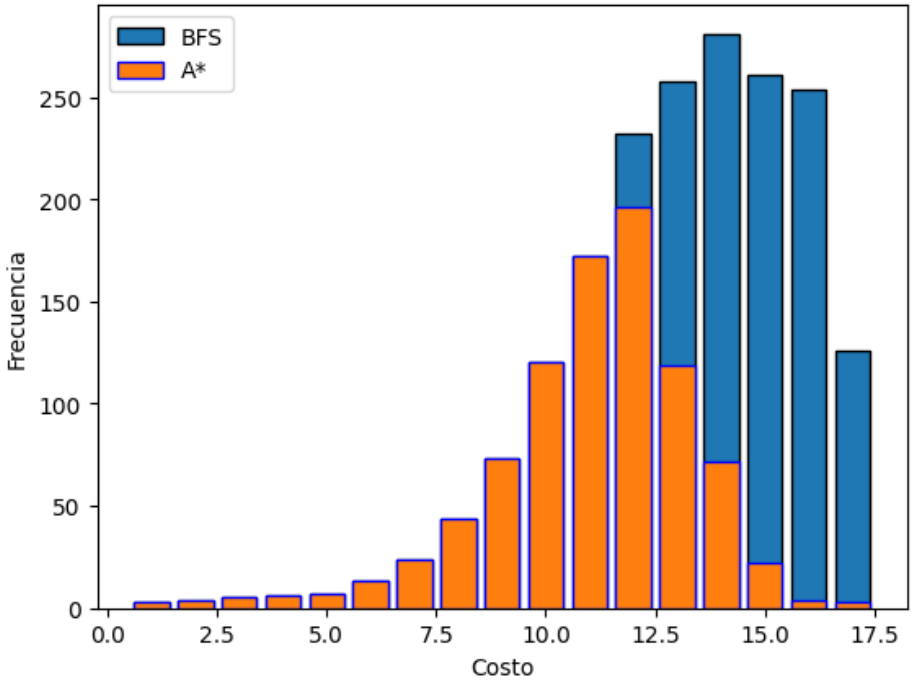
Comparación de los modelos

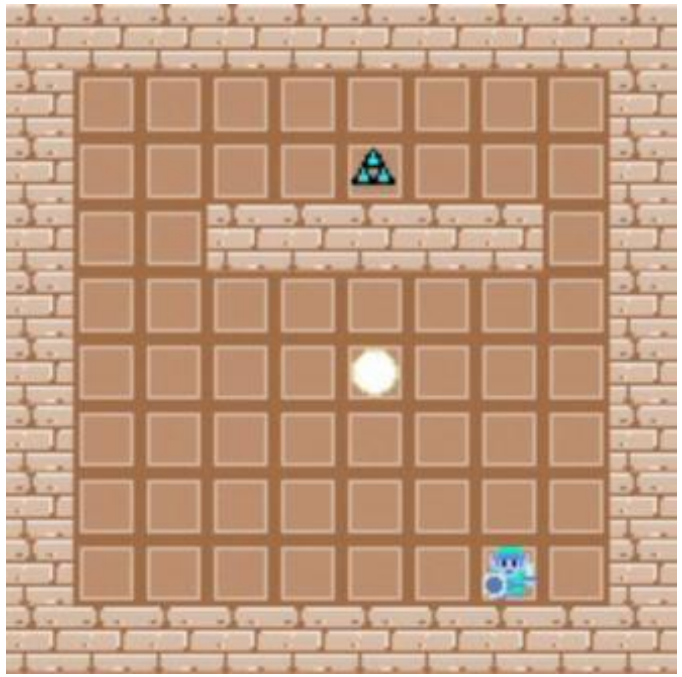
	Costo	Nodos expandidos	Nodos frontera	tiempo
BFS	17	1642	242	1.24 s
DFS	131	2508	122	4.58 s
Greedy	19	1009	54	1.34 s
Greedy con duplicado	Se usa toda la RAM disponible			
A*	17	679	209	0.58 s
SA con do	60	50 intentos		0.8 s
	Media 42, min 31	5000 intentos(0.5% efectividad)		3 min

Frecuencia de estados visitados en función del costo



Frecuencia de estados visitados en función del costo

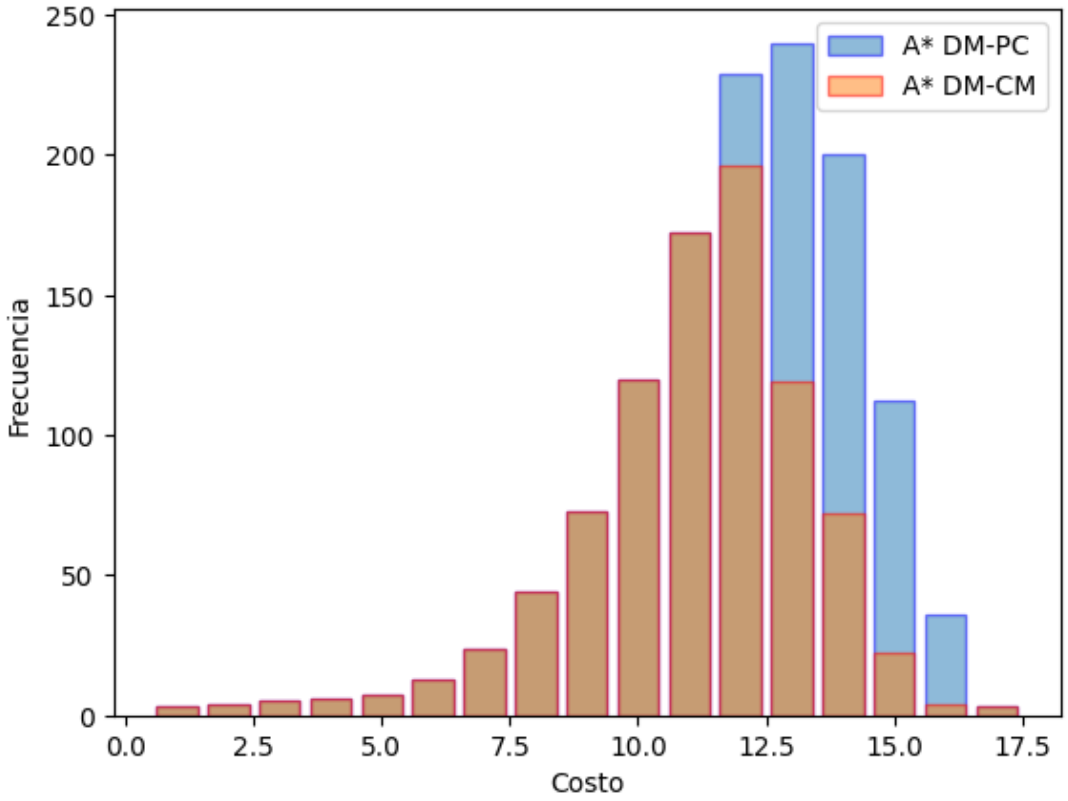




Comparación de heurísticas

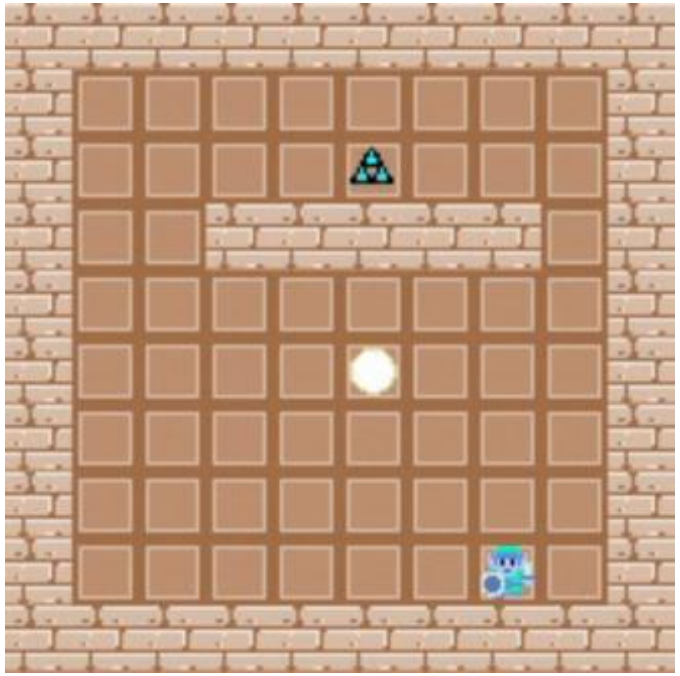
	Costo	Nodos expandidos	Nodos frontera	tiempo
A* DM-CM	17	679	209	0.58 s
A* DM-PC	17	963	329	1.04 s
Nodos expandidos en común		589		

Frecuencia de estados visitados en función del costo



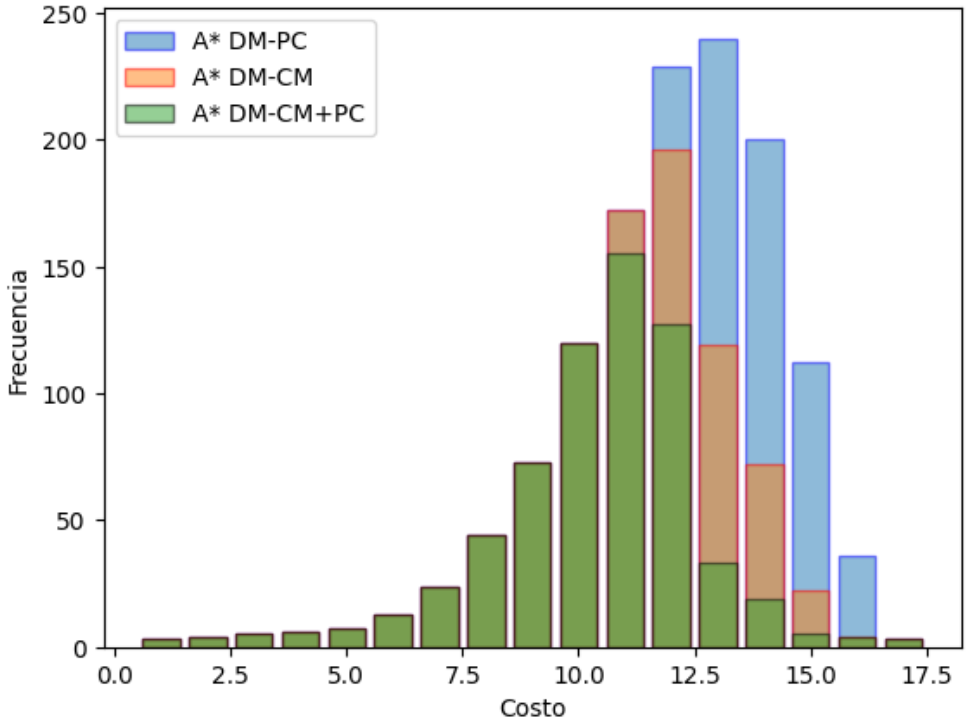
Comparación de heurísticas

		Costo	Nodos expandidos	Nodos frontera	Tiempo
Caso 1	A* DM-CM	17	679	209	0.58 s
Caso 2	A* DM-PC	17	963	329	1.04 s
Caso 3	A* DM-CM+PC	17	406	237	1.2

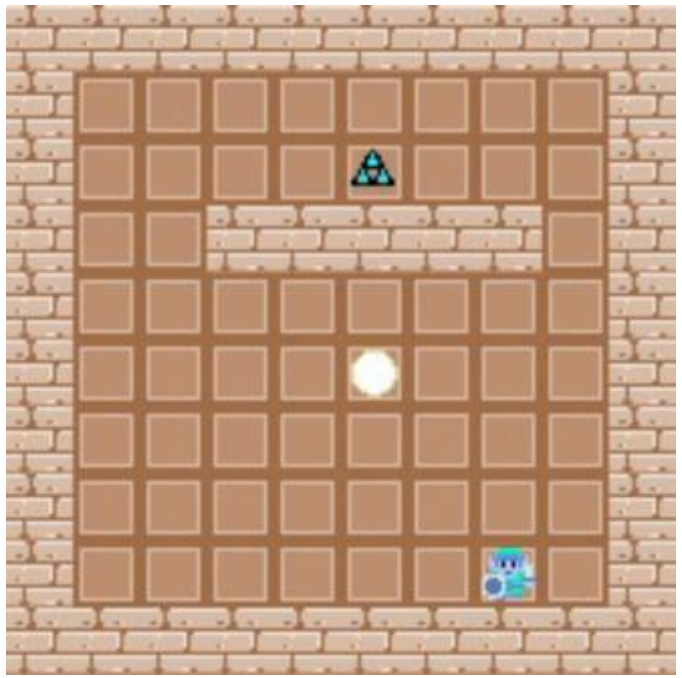


Nodos expandidos en común h1-h3: 406 -> h3 domina a h1
Nodos expandidos en común h2-h3: 406 -> h3 domina a h2

Frecuencia de estados visitados en función del costo

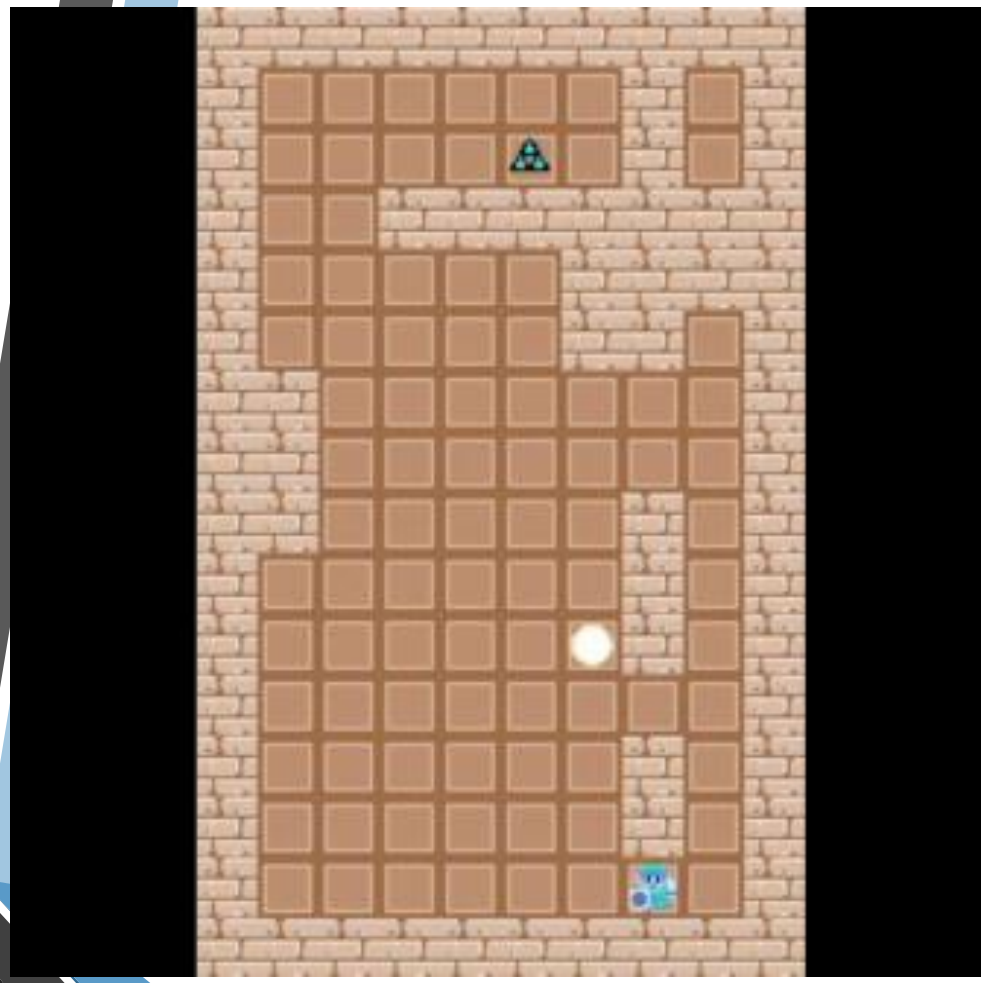


Comparación de heurísticas



		Costo	Nodos expandidos	Nodos frontera	Tiempo
Caso 3	A* DM-CM+PC	17	406	240	1.2 s
Caso 4	Caso 3 con diagonal cero	17	406	240	0.39 s

Nodos expandidos en común h4-h3: 406



Comparación de heurísticas

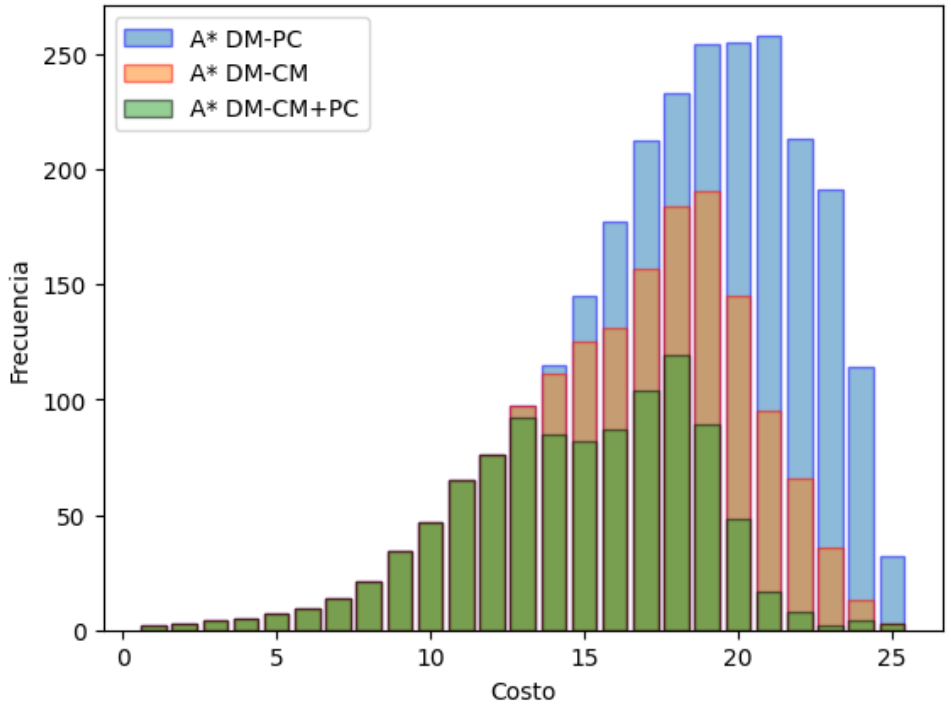
		Costo	Nodos expandidos	Nodos frontera	Nodos totales	tiempo
Caso 3	A* DM-CM+PC	25	741	287	1028	1.2 s
Caso 4	Caso 3 con diagonal cero	25	740	288	1028	2 s

```
resultado = [a - b for a, b in zip(frecuencias_values_A4, frecuencias_values_A3)]
print(resultado)
```

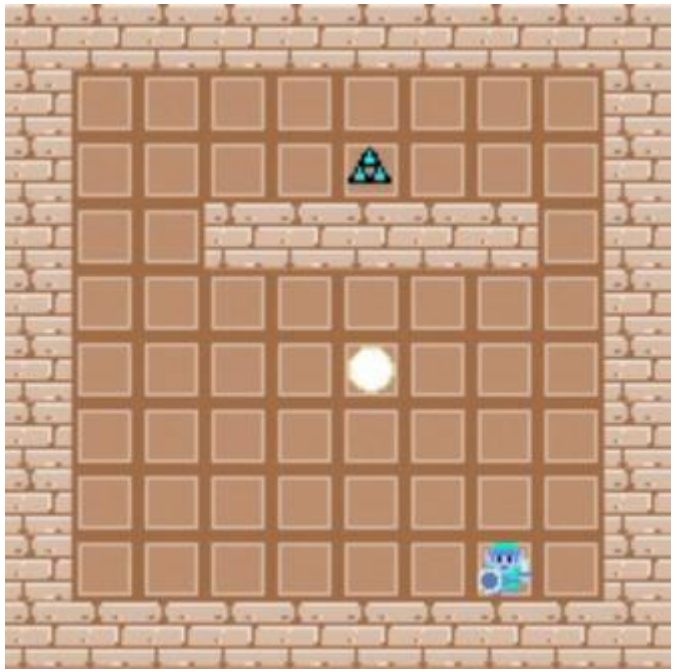
```
[0, 0, 0, -1, -1, 0, 0, 0, -1, -1, 2, 2, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
```

Nodos en común h3 y h4: 740 h4 domina a h3

Frecuencia de estados visitados en función del costo



Greedy



Problemas de memoria si se cuenta los duplicados

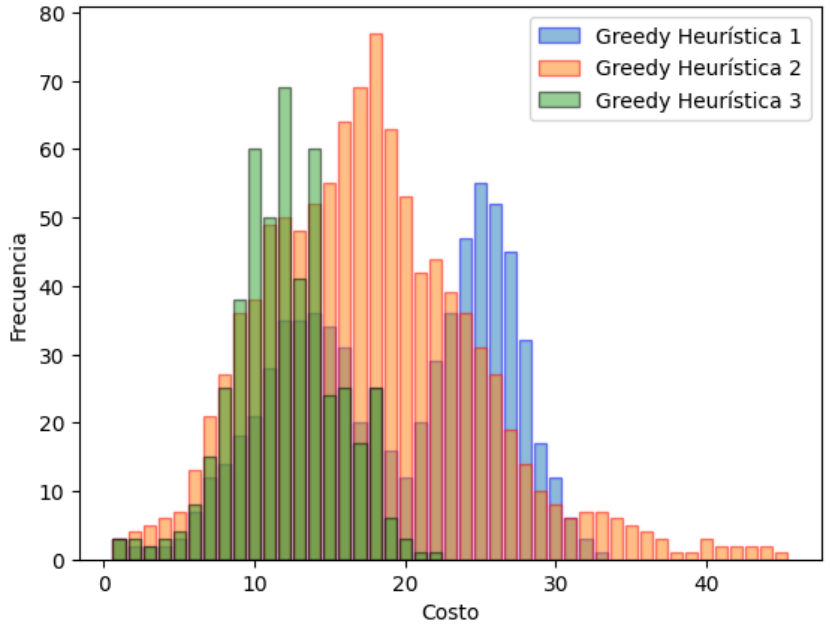
Comparación de heurísticas

	Modelo	Costo	Nodos expandidos	Nodos frontera	tiempo
Heurística 1 Caja-Marca	A*	17	679	209	0.58 s
Heurística 2 Caja-Personaje	A*	17	963	329	1.04 s
Heurística 3 : Heurística 1+2	A*	17	406	237	1.2

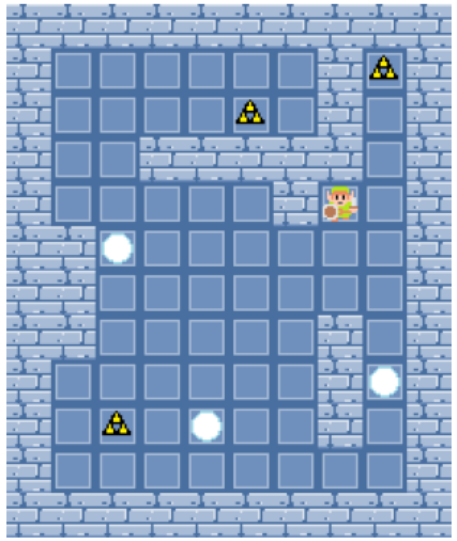
Comparación de heurísticas

	Modelo	Costo	Nodos expandidos	Nodos frontera	tiempo
Heurística 1 Caja-Marca	Greedy	19	1009	54	0.6 s
Heurística 2 Caja-Personaje	Greedy	19	393	319	0.33 s
Heurística 3 : Heurística 1+2	Greedy	19	291	193	0.4

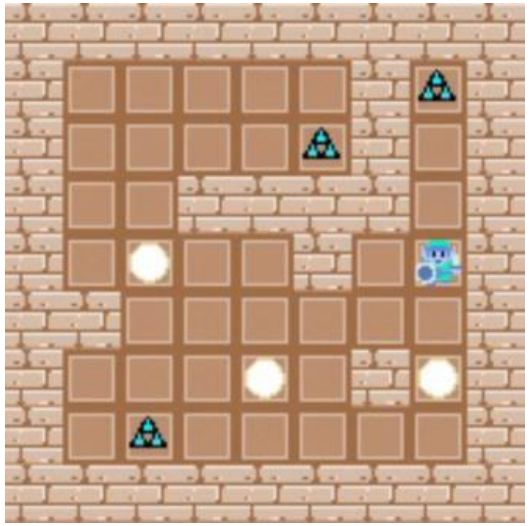
Frecuencia de estados visitados en función del costo



Problemas de +1 caja



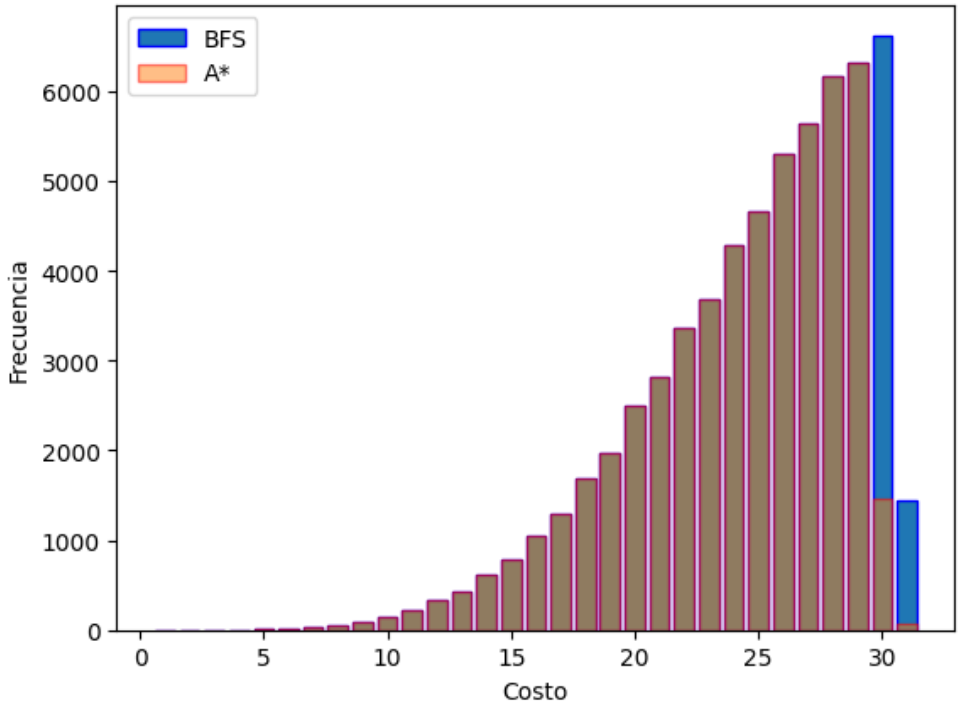
Problemas de memoria



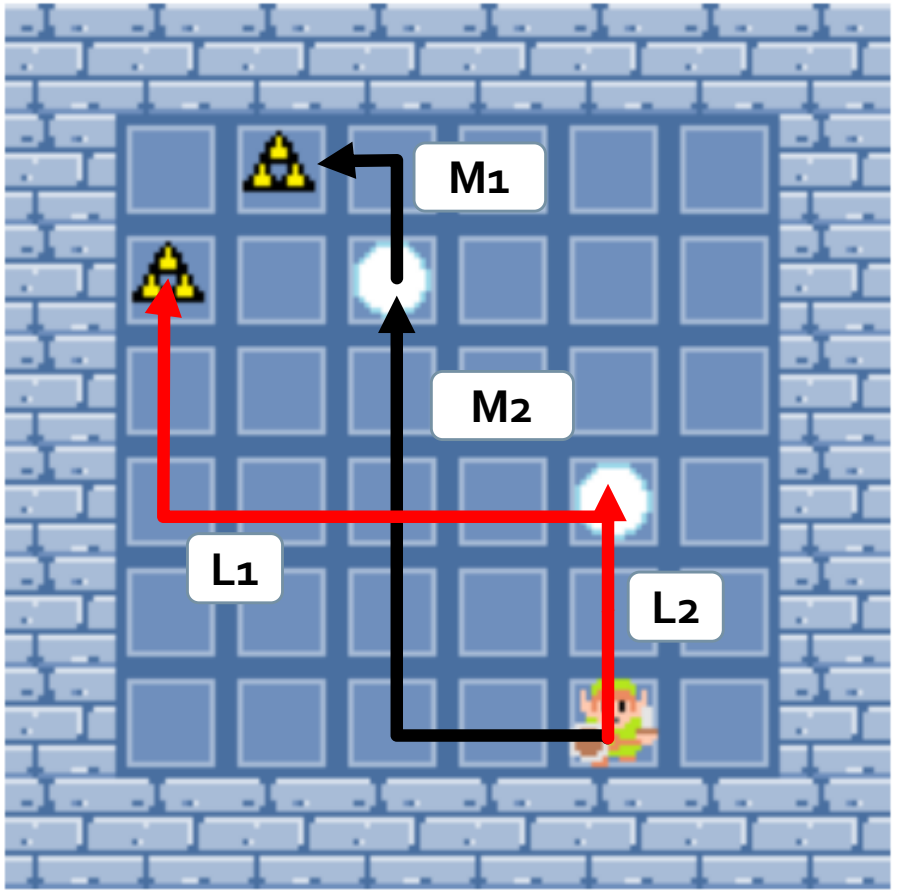
Comparación de los modelos

	Costo	Nodos expandidos	Nodos frontera	tiempo
BFS	31	54914	6618	41 s
A* conteo cajas incorrectas	31	48510	6487	39 s

Frecuencia de estados visitados en función del costo



Propuesta para Simulated Annealing multicaja



$FO: L1 \,.\, L2 + M1.M2$

Cuando una caja se activa su termino vale cero. Se puede Priorizar una caja añadiendo un factor

$FO: k(L1 \,.\, L2) + M1.M2$

Heurísticas no admisibles

Planteo, no se implementaron.

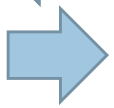
Penalizar si el personaje se mueve para una dirección, no se mueve una caja y su siguiente movimiento es la dirección contraria.

Puede solucionar el problema del Greedy

si  Se mueve el personaje y la caja

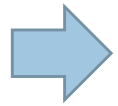
si 

si 

si 

Siguiente turno no se penaliza

Siguiente turno se penaliza: 

Siguiente turno se penaliza: 

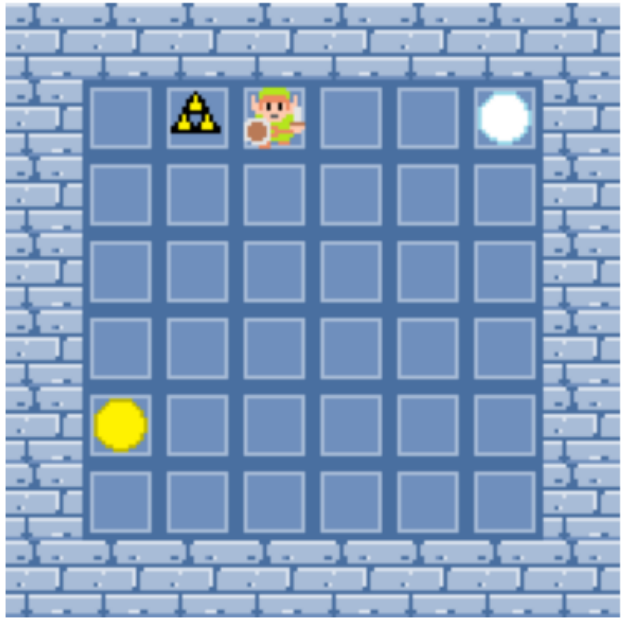
Siguiente turno se penaliza: 

Heurísticas no admisibles

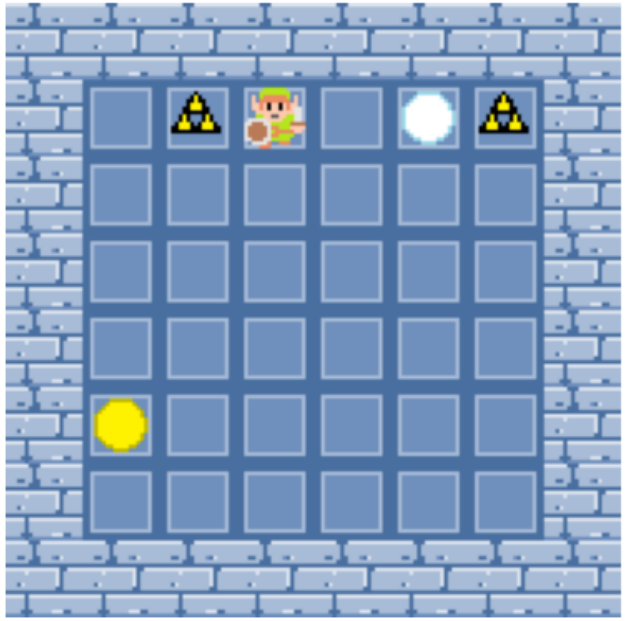
Planteo, no se implementaron.

Penalizar si la caja está rodeada por n paredes.

Problema: ¿el agente es mas inteligente?



Pero si:



Ejercicio 1

tablero inicial:

5	7	3
8	2	
1	6	4

tablero solución:

1	2	3
8		4
7	6	5

Estructura de datos:

-Matriz, el espacio vacío se le asigna un valor

```
[[5, 7, 3], [8, 2, 10], [1, 6, 4]]
```

-Adyacencia:

$Ad_1 = [5,7]$

$Ad_2 = [6,7,8,10]$

$Ad_3 = [7,10]$

$Ad_4 = [6,10]$

$Ad_5 = [7,8]$

$Ad_6 = [1,2,4]$

$Ad_7 = [2,3,5]$

$Ad_8 = [1,2,5]$

Ejercicio 1

tablero inicial:

5	7	3
8	2	
1	6	4

tablero solución:

1	2	3
8		4
7	6	5

Heurísticas:

-Distancia manhattan numero tablero actual(ta) y solución(ts)

$$h_i = |x_{i\,ta} - x_{i\,ts}| + |y_{i\,ta} - y_{i\,ts}|$$
$$con\ i\ \in\ \{1,2,3,4,5,6,7,8\}$$

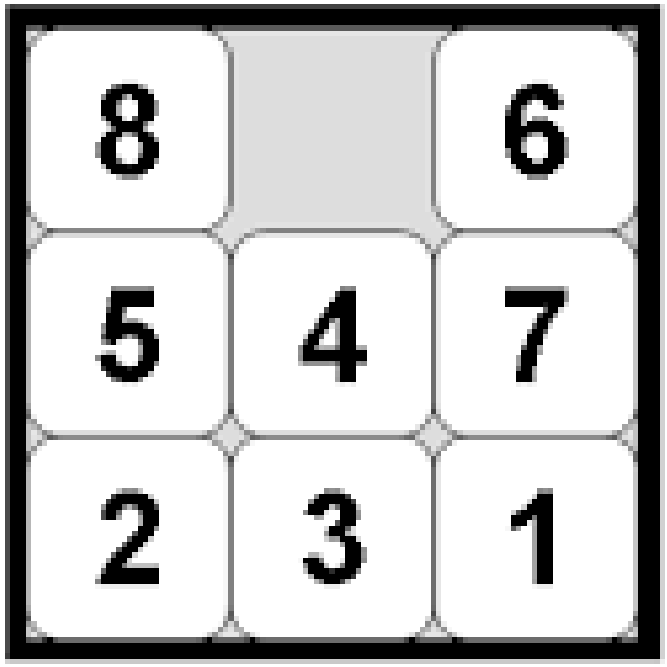
-Contar las fichas en las posiciones incorrectas.

-Distancia Euclidiana

$$h_i = \sqrt{(x_{i\,ta} - x_{i\,ts})^2 + (y_{i\,ta} - y_{i\,ts})^2}$$
$$con\ i\ \in\ \{1,2,3,4,5,6,7,8\}$$

Ejercicio 1

Métodos de búsqueda



¿Es simple?

Gracias por su atención