



Ayudantía 5

Heurísticas y A*

Por Daniel Toribio y Felipe Espinoza

22 de abril 2024



Algoritmos vistos hasta ahora

**Búsqueda
Genérica**

BFS

DFS

**Best First
Search**

Otros algoritmos:

Dijkstra

**Bellman-
Ford**

**Beam
Search**

¿Qué tienen en común?



Algoritmos vistos hasta ahora

- Encuentran caminos más cortos en un grafo ponderado, es decir, **encuentran soluciones óptimas**
- ¿Qué sucede cuando el grafo de búsqueda es **MUY** grande?

**Alto consumo
de memoria**

**Exploración
innecesaria**

**Gran tiempo de
ejecución**

- Queremos que nuestra búsqueda obtenga el camino al nodo objetivo, **evitando revisar nodos innecesarios**. ¿Cómo lo logramos?



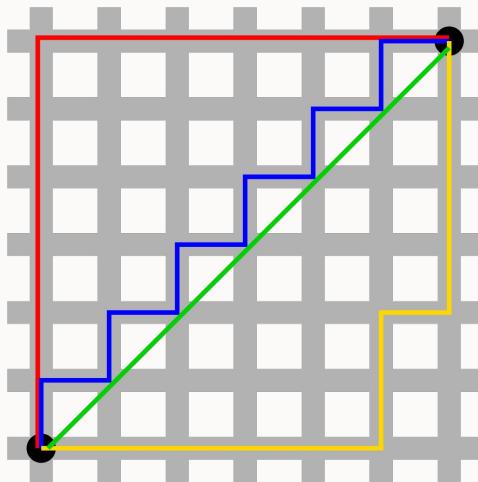
Función Heurística

- Función que estima la cercanía de un nodo **s** a un nodo objetivo **G**.
- Para crearla necesitamos saber donde se ubica el nodo objetivo o como conseguirlo.
- Con ella podemos discriminar qué nodo **conviene** explorar, priorizando nodos que podrían llevar a la solución de forma más rápida.

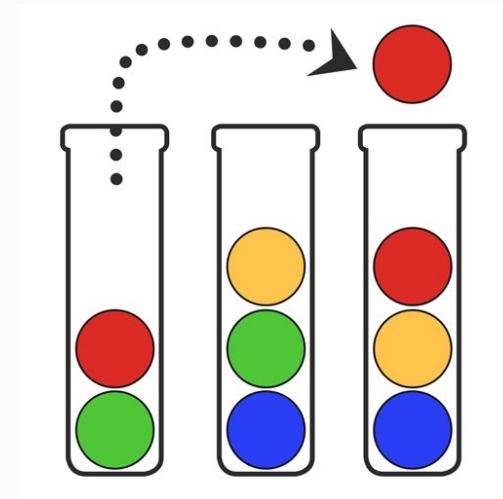
$$h(s)$$



Ejemplos de Heurísticas



- ● ● Distancia Manhattan
- ● Distancia Euclidiana



Cantidad de bolitas en un frasco que no tienen un mismo color



Ejemplos de Heurísticas

1	2	3
4		6
7	8	5

Cantidad de piezas en la posición correcta

O	X	O
X		X
	O	

Cantidad de figuras del jugador que están alineadas



Admisibilidad

Definición

Una heurística h se dice **admissible** si para todo estado s se cumple que:

$$h(s) \leq h^*(s)$$

Donde $h^*(s)$ es el costo de un camino óptimo desde el estado s a un estado objetivo

Nuestra función es admissible si nunca sobreestima respecto a un camino óptimo



Admisibilidad

Definamos el costo de moverse entre casillas igual a 1. ¿Cuál heurística es admisible?

H1

$H=6$				G
S				
$H=8$				

H2

$H=8$				G
S				
$H=6$				



Admisibilidad

Definamos el costo de moverse entre casillas igual a 1. ¿Cuál heurística es admisible?

H1 es admisible

H=6				G
S				
H=8				

H2 no es admisible

$$H2 = 8 > 6 = h^*$$

H=8				G
S				
H=6				



Consistencia

Definición

Una heurística h se dice **consistente** si y sólo si:

- $h(s) = 0$, para todo $s \in G$
- $h(s) \leq c(s, s') + h(s')$ para todo vecino s' de s

O de manera equivalente $h(s) - h(s') \leq c(s, s')$

Nuestra función es consistente si nunca sobreestima en los estados objetivos ni el **costo es mayor que la variación de la heurística** entre dos estados vecinos



Consistencia

Definamos el costo de moverse entre casillas igual a 1. ¿Cuál heurística es admisible?

H1

$H=10$	$H=8$	$H=6$	$H=4$	$H=2$
$H=12$		$H=8$		$H=0$
S		$H=10$		$H=2$
$H=16$	$H=14$	$H=12$		$H=4$
$H=18$		$H=14$		$H=6$

G

H2

$H=5$	$H=4$	$H=3$	$H=2$	$H=1$
$H=6$		$H=4$		$H=0$
S		$H=5$		$H=1$
$H=8$	$H=7$	$H=6$		$H=2$
$H=9$		$H=7$		$H=3$

G



Consistencia

Definamos el costo de moverse entre casillas igual a 1. ¿Cuál heurística es admisible?

H1 no es consistente

$H=10$	$H=8$	$H=6$	$H=4$	$H=2$
$H=12$		$H=8$		$H=0$
S		$H=10$		$H=2$
$H=16$	$H=14$	$H=12$		$H=4$
$H=18$		$H=14$		$H=6$

G

H2 si lo es

$H=5$	$H=4$	$H=3$	$H=2$	$H=1$
$H=6$		$H=4$		$H=0$
S		$H=5$		$H=1$
$H=8$	$H=7$	$H=6$		$H=2$
$H=9$		$H=7$		$H=3$

G



Relación consistencia y admisibilidad

Definición

Si una heurística h es consistente, entonces h es admisible

¡Esta relación no es invertible! Es decir, si una heurística es admisible, no necesariamente va a ser consistente.



A*

A* es un algoritmo utilizado en problemas de búsqueda, su característica distintiva es que utiliza heurística, utilizando la siguiente función de evaluación $f(n) = g(n) + h(n)$

- La rapidez y complejidad computacional está estrechamente relacionada a la heurística seleccionada.
- Siempre entrega soluciones óptimas si se usa con una **heurística admisible**.
- Dijkstra es A* con heurística cero.

Dijkstra

$$f(n) = g(n)$$

A*

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



A*

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

$g(n)$: Costo de un camino desde s_0 hasta un nodo n .

$g(n)$ **NO ES FUNCIÓN!**



Algoritmo de A*

Algoritmo A*

Input: Un problema de búsqueda (S, A, s_0, G)

Output: Un nodo objetivo

- 1 **for each** $s \in S$ **do** $g(s) \leftarrow \infty$
- 2 $Open \leftarrow \{s_0\}$
- 3 $g(s_0) \leftarrow 0; f(s_0) \leftarrow h(s_0)$
- 4 **while** $Open \neq \emptyset$
- 5 Extrae un u desde $Open$ con menor valor- f
- 6 **if** u es objetivo **return** u
- 7 **for each** $v \in Succ(u)$ **do**
- 8 Insertar v

La extracción es del u con menor valor $f(s) = g(s) + h(s)$

Revisa si llego al nodo objetivo cuando se expande

Procedimiento de insertar v

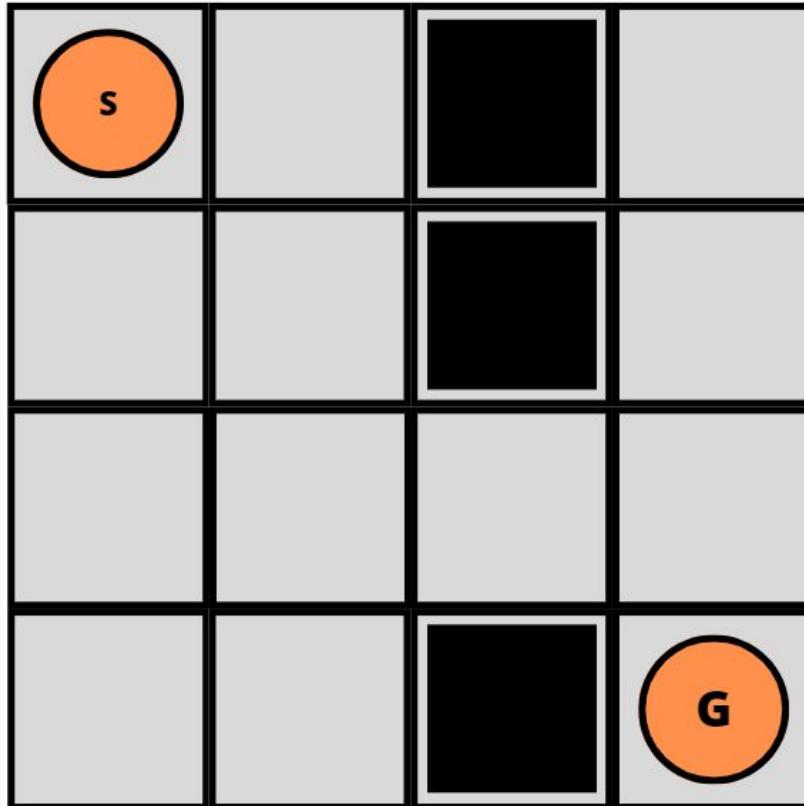


Procedimiento insertar v

Insertar v en Open

- 1 $\text{cost}_v = g(u) + c(u, v)$ // el costo de llegar a v por u
- 2 **if** $\text{cost}_v \geq g(v)$ **return** // seguimos solo si $\text{cost}_v < g(v)$
- 3 $\text{parent}(v) \leftarrow u$
- 4 $g(v) \leftarrow \text{cost}_v$
- 5 $f(v) \leftarrow g(v) + h(v)$
- 6 **if** $v \in \text{Open}$ **then** Reordenar Open // depende de la impl.
- 7 **else** Insertar v en Open

EJEMPLO



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

$g(s)$: largo del camino

$h(s)$: distancia de manhattan



1 for each $s \in S$ do $g(s) \leftarrow \infty$

2 $Open \leftarrow \{s_0\}$

3 $g(s_0) \leftarrow 0; f(s_0) \leftarrow h(s_0)$

4 while $Open \neq \emptyset$

5 Extrae un u desde $Open$ con menor valor- f

6 if u es objetivo return u

7 for each $v \in Succ(u)$ do

1 $cost_v = g(u) + c(u, v)$

2 if $cost_v \geq g(v)$ return

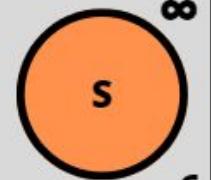
3 $parent(v) \leftarrow u$

4 $g(v) \leftarrow cost_v$

5 $f(v) \leftarrow g(v) + h(v)$

6 if $v \in Open$ then Reordenar $Open$

7 else Insertar v en $Open$

	∞	∞	∞	∞
6		5		3
∞	∞	∞		∞
	5	4		2
∞	∞	∞	∞	∞
	4	3	2	1
∞	∞	∞		
	3	2		
	∞			0

1 for each $s \in S$ do $g(s) \leftarrow \infty$

2 $Open \leftarrow \{s_0\}$

3 $g(s_0) \leftarrow 0; f(s_0) \leftarrow h(s_0)$

4 while $Open \neq \emptyset$

5 Extrae un u desde $Open$ con menor valor- f

6 if u es objetivo return u

7 for each $v \in Succ(u)$ do

1 $cost_v = g(u) + c(u, v)$

2 if $cost_v \geq g(v)$ return

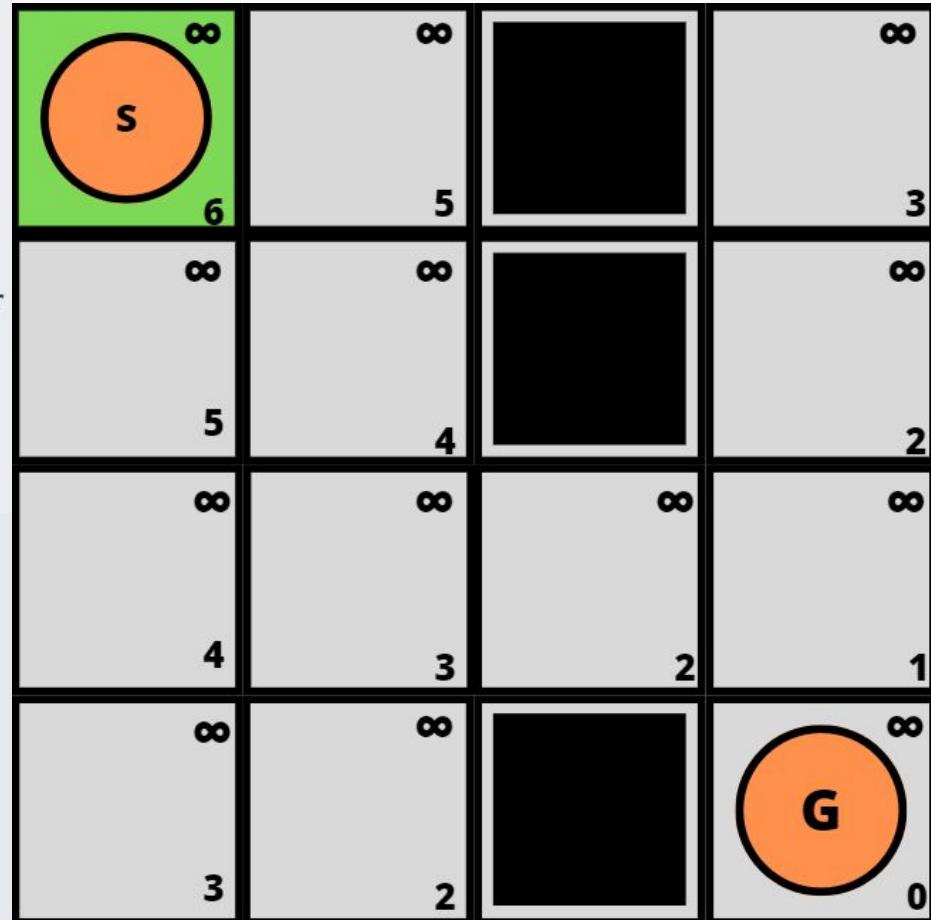
3 $parent(v) \leftarrow u$

4 $g(v) \leftarrow cost_v$

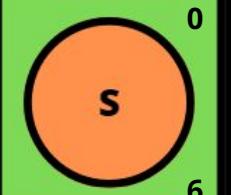
5 $f(v) \leftarrow g(v) + h(v)$

6 if $v \in Open$ then Reordenar $Open$

7 else Insertar v en $Open$



- 1 **for each** $s \in \mathcal{S}$ **do** $g(s) \leftarrow \infty$
- 2 $Open \leftarrow \{s_0\}$
- 3 $g(s_0) \leftarrow 0; f(s_0) \leftarrow h(s_0)$
- 4 **while** $Open \neq \emptyset$
- 5 Extrae un u desde $Open$ con menor valor- f
- 6 **if** u es objetivo **return** u
- 7 **for each** $v \in Succ(u)$ **do**
 - 1 $cost_v = g(u) + c(u, v)$
 - 2 **if** $cost_v \geq g(v)$ **return**
 - 3 $parent(v) \leftarrow u$
 - 4 $g(v) \leftarrow cost_v$
 - 5 $f(v) \leftarrow g(v) + h(v)$
 - 6 **if** $v \in Open$ **then** Reordenar $Open$
 - 7 **else** Insertar v en $Open$

	0 6	∞ 5		∞ 3
∞	∞	∞		∞
5	4		2	
∞	∞	∞	∞	∞
4	3	2	1	
∞	∞		0	0

1 for each $s \in S$ do $g(s) \leftarrow \infty$

2 $Open \leftarrow \{s_0\}$

3 $g(s_0) \leftarrow 0; f(s_0) \leftarrow h(s_0)$

4 while $Open \neq \emptyset$

5 Extrae un u desde $Open$ con menor valor- f

6 if u es objetivo return u

7 for each $v \in Succ(u)$ do

1 $cost_v = g(u) + c(u, v)$

2 if $cost_v \geq g(v)$ return

3 $parent(v) \leftarrow u$

4 $g(v) \leftarrow cost_v$

5 $f(v) \leftarrow g(v) + h(v)$

6 if $v \in Open$ then Reordenar $Open$

7 else Insertar v en $Open$

6	6	∞		
s	0	5		
∞	∞	∞		
5	4	2		
∞	∞	∞	∞	∞
4	3	2	1	
∞	∞	∞		
3	2			
∞	∞	∞	∞	∞
G	0			

```

1 for each  $s \in S$  do  $g(s) \leftarrow \infty$ 
2  $Open \leftarrow \{s_0\}$ 
3  $g(s_0) \leftarrow 0; f(s_0) \leftarrow h(s_0)$ 
4 while  $Open \neq \emptyset$ 
5   Extrae un  $u$  desde  $Open$  con menor valor- $f$ 
6   if  $u$  es objetivo return  $u$ 
7   for each  $v \in Succ(u)$  do
        1  $cost_v = g(u) + c(u, v)$ 
        2 if  $cost_v \geq g(v)$  return
        3  $parent(v) \leftarrow u$ 
        4  $g(v) \leftarrow cost_v$ 
        5  $f(v) \leftarrow g(v) + h(v)$ 
        6 if  $v \in Open$  then Reordenar  $Open$ 
        7 else Insertar  $v$  en  $Open$ 

```

6	6	∞	∞
s	0	5	3
∞	∞	∞	∞
5	4	2	2
∞	∞	∞	∞
4	3	2	1
∞	∞	∞	∞
3	2		0

```

1 for each  $s \in S$  do  $g(s) \leftarrow \infty$ 
2  $Open \leftarrow \{s_0\}$ 
3  $g(s_0) \leftarrow 0; f(s_0) \leftarrow h(s_0)$ 
4 while  $Open \neq \emptyset$ 
5   Extrae un  $u$  desde  $Open$  con menor valor- $f$ 
6   if  $u$  es objetivo return  $u$ 
7   for each  $v \in Succ(u)$  do
        1  $cost_v = g(u) + c(u, v)$ 
        2 if  $cost_v \geq g(v)$  return
        3  $parent(v) \leftarrow u$ 
        4  $g(v) \leftarrow cost_v$ 
        5  $f(v) \leftarrow g(v) + h(v)$ 
        6 if  $v \in Open$  then Reordenar  $Open$ 
        7 else Insertar  $v$  en  $Open$ 

```

6	6	∞		∞
s		(0,0)	5	
0				3
∞		∞		∞
5		4		2
∞		∞	∞	∞
4		3	2	1
∞		∞	∞	∞
3		2		
∞		∞		
G				0

- 1 **for each** $s \in \mathcal{S}$ **do** $g(s) \leftarrow \infty$
- 2 $Open \leftarrow \{s_0\}$
- 3 $g(s_0) \leftarrow 0; f(s_0) \leftarrow h(s_0)$
- 4 **while** $Open \neq \emptyset$
- 5 Extrae un u desde $Open$ con menor valor- f
- 6 **if** u es objetivo **return** u
- 7 **for each** $v \in Succ(u)$ **do**
 - 1 $cost_v = g(u) + c(u, v)$
 - 2 **if** $cost_v \geq g(v)$ **return**
 - 3 $parent(v) \leftarrow u$
 - 4 $g(v) \leftarrow cost_v$ **highlighted**
 - 5 $f(v) \leftarrow g(v) + h(v)$
 - 6 **if** $v \in Open$ **then** Reordenar $Open$
 - 7 **else** Insertar v en $Open$

S	6	6	1		∞
0	(0,0)	5			3
∞		∞			∞
5		4			2
∞		∞			∞
4		3			1
∞		∞			∞
3		2			0
G					∞

```

1 for each  $s \in S$  do  $g(s) \leftarrow \infty$ 
2  $Open \leftarrow \{s_0\}$ 
3  $g(s_0) \leftarrow 0; f(s_0) \leftarrow h(s_0)$ 
4 while  $Open \neq \emptyset$ 
5   Extrae un  $u$  desde  $Open$  con menor valor- $f$ 
6   if  $u$  es objetivo return  $u$ 
7   for each  $v \in Succ(u)$  do
        1  $cost_v = g(u) + c(u, v)$ 
        2 if  $cost_v \geq g(v)$  return
        3  $parent(v) \leftarrow u$ 
        4  $g(v) \leftarrow cost_v$ 
        5  $f(v) \leftarrow g(v) + h(v)$ 
        6 if  $v \in Open$  then Reordenar  $Open$ 
        7 else Insertar  $v$  en  $Open$ 

```

6	6	1	∞
0	(0,0)	5	3
∞	∞	∞	∞
5	4	2	1
∞	∞	∞	∞
4	3	2	1
∞	∞	∞	∞
3	2		0

1 **for each** $s \in S$ **do** $g(s) \leftarrow \infty$

2 $Open \leftarrow \{s_0\}$

3 $g(s_0) \leftarrow 0; f(s_0) \leftarrow h(s_0)$

4 **while** $Open \neq \emptyset$

5 Extrae un u desde $Open$ con menor valor- f

6 **if** u es objetivo **return** u

7 **for each** $v \in Succ(u)$ **do**

1 $cost_v = g(u) + c(u, v)$

2 **if** $cost_v \geq g(v)$ **return**

3 $parent(v) \leftarrow u$

4 $g(v) \leftarrow cost_v$

5 $f(v) \leftarrow g(v) + h(v)$

6 **if** $v \in Open$ **then** Reordenar $Open$

7 **else** Insertar v en $Open$

6	6	1		∞
s	(0,0)	5		3
0				
∞	∞			∞
5	4			2
∞	∞			∞
4	3			1
∞	∞			∞
3	2			
∞	∞			
G				0

- 1 **for each** $s \in \mathcal{S}$ **do** $g(s) \leftarrow \infty$
- 2 $Open \leftarrow \{s_0\}$
- 3 $g(s_0) \leftarrow 0; f(s_0) \leftarrow h(s_0)$
- 4 **while** $Open \neq \emptyset$
- 5 Extrae un u desde $Open$ con menor valor- f
- 6 **if** u es objetivo **return** u
- 7 **for each** $v \in Succ(u)$ **do**
 - 1 $cost_v = g(u) + c(u, v)$
 - 2 **if** $cost_v \geq g(v)$ **return**
 - 3 $parent(v) \leftarrow u$
 - 4 $g(v) \leftarrow cost_v$
 - 5 $f(v) \leftarrow g(v) + h(v)$
 - 6 **if** $v \in Open$ **then** Reordenar $Open$
 - 7 **else** Insertar v en $Open$

S	6	6	1	∞
0	(0,0)	5		3
∞		∞		∞
5		4		2
∞		∞	∞	∞
	4	3	2	1
∞		∞		∞
3		2		
G	∞			0

```

1 for each  $s \in S$  do  $g(s) \leftarrow \infty$ 
2  $Open \leftarrow \{s_0\}$ 
3  $g(s_0) \leftarrow 0; f(s_0) \leftarrow h(s_0)$ 
4 while  $Open \neq \emptyset$ 
5   Extrae un  $u$  desde  $Open$  con menor valor- $f$ 
6   if  $u$  es objetivo return  $u$ 
7   for each  $v \in Succ(u)$  do
        1  $cost_v = g(u) + c(u, v)$ 
        2 if  $cost_v \geq g(v)$  return
        3  $parent(v) \leftarrow u$ 
        4  $g(v) \leftarrow cost_v$ 
        5  $f(v) \leftarrow g(v) + h(v)$ 
        6 if  $v \in Open$  then Reordenar  $Open$ 
        7 else Insertar  $v$  en  $Open$ 

```

6	6	6	1		∞
S					
0	(0,0)	5			3
∞		∞			∞
(0,0)	5	4			2
∞		∞	∞		∞
	4	3	2		1
∞		∞			∞
	3	2			
∞					∞
	G				0

- 1 **for each** $s \in S$ **do** $g(s) \leftarrow \infty$
- 2 $Open \leftarrow \{s_0\}$
- 3 $g(s_0) \leftarrow 0; f(s_0) \leftarrow h(s_0)$
- 4 **while** $Open \neq \emptyset$
- 5 Extrae un u desde $Open$ con menor valor- f
- 6 **if** u es objetivo **return** u
- 7 **for each** $v \in Succ(u)$ **do**
 - 1 $cost_v = g(u) + c(u, v)$
 - 2 **if** $cost_v \geq g(v)$ **return**
 - 3 $parent(v) \leftarrow u$
 - 4 $g(v) \leftarrow cost_v$ **highlighted**
 - 5 $f(v) \leftarrow g(v) + h(v)$
 - 6 **if** $v \in Open$ **then** Reordenar $Open$
 - 7 **else** Insertar v en $Open$

S	6	6	6	1			∞
0	(0,0)	5					3
1		∞					∞
(0,0)	5		4				2
∞		∞		∞			∞
4		3		2			1
∞		∞					∞
3		2					0
G							∞

1 **for each** $s \in S$ **do** $g(s) \leftarrow \infty$

2 $Open \leftarrow \{s_0\}$

3 $g(s_0) \leftarrow 0; f(s_0) \leftarrow h(s_0)$

4 **while** $Open \neq \emptyset$

5 Extrae un u desde $Open$ con menor valor- f

6 **if** u es objetivo **return** u

7 **for each** $v \in Succ(u)$ **do**

1 $cost_v = g(u) + c(u, v)$

2 **if** $cost_v \geq g(v)$ **return**

3 $parent(v) \leftarrow u$

4 $g(v) \leftarrow cost_v$

5 $f(v) \leftarrow g(v) + h(v)$

6 **if** $v \in Open$ **then** Reordenar $Open$

7 **else** Insertar v en $Open$

6	6	6	1		∞
S					
0	(0,0)	5			3
6	1		∞		∞
(0,0)	5	4			2
	∞	∞	∞		∞
	4	3	2		1
	∞	∞			
	3	2			
					∞
				G	0

1 **for each** $s \in \mathcal{S}$ **do** $g(s) \leftarrow \infty$

2 $Open \leftarrow \{s_0\}$

3 $g(s_0) \leftarrow 0; f(s_0) \leftarrow h(s_0)$

4 **while** $Open \neq \emptyset$

5 Extrae un u desde $Open$ con menor valor- f

6 **if** u es objetivo **return** u

7 **for each** $v \in Succ(u)$ **do**

1 $cost_v = g(u) + c(u, v)$

2 **if** $cost_v \geq g(v)$ **return**

3 $parent(v) \leftarrow u$

4 $g(v) \leftarrow cost_v$

5 $f(v) \leftarrow g(v) + h(v)$

6 **if** $v \in Open$ **then** Reordenar $Open$

7 **else** Insertar v en $Open$

6	6	6	1		∞
s					
0	(0,0)	5			3
6	1		∞		∞
(0,0)	5		4		2
	∞	∞	∞	∞	∞
	4	3	2	1	
	∞	∞			
	3	2			
				G	∞
				0	

```

1 for each  $s \in S$  do  $g(s) \leftarrow \infty$ 
2  $Open \leftarrow \{s_0\}$ 
3  $g(s_0) \leftarrow 0; f(s_0) \leftarrow h(s_0)$ 
4 while  $Open \neq \emptyset$ 
5   Extrae un  $u$  desde  $Open$  con menor valor- $f$ 
6   if  $u$  es objetivo return  $u$ 
7   for each  $v \in Succ(u)$  do
        1  $cost_v = g(u) + c(u, v)$ 
        2 if  $cost_v \geq g(v)$  return
        3  $parent(v) \leftarrow u$ 
        4  $g(v) \leftarrow cost_v$ 
        5  $f(v) \leftarrow g(v) + h(v)$ 
        6 if  $v \in Open$  then Reordenar  $Open$ 
        7 else Insertar  $v$  en  $Open$ 

```

	6	6	6	1		∞
	S					
	0	(0,0)	5			3
6	6	1	∞			∞
	(0,0)	5		4		2
		∞	∞	∞		∞
		4	3	2		1
	∞	∞	∞			
	3		2			
					G	0

```

1 for each  $s \in S$  do  $g(s) \leftarrow \infty$ 
2  $Open \leftarrow \{s_0\}$ 
3  $g(s_0) \leftarrow 0; f(s_0) \leftarrow h(s_0)$ 
4 while  $Open \neq \emptyset$ 
5   Extrae un  $u$  desde  $Open$  con menor valor- $f$ 
6   if  $u$  es objetivo return  $u$ 
7   for each  $v \in Succ(u)$  do
        1  $cost_v = g(u) + c(u, v)$ 
        2 if  $cost_v \geq g(v)$  return
        3  $parent(v) \leftarrow u$ 
        4  $g(v) \leftarrow cost_v$ 
        5  $f(v) \leftarrow g(v) + h(v)$ 
        6 if  $v \in Open$  then Reordenar  $Open$ 
        7 else Insertar  $v$  en  $Open$ 

```

6	6	6	1		∞
S					
0	(0,0)	5			3
6	1	∞			∞
(0,0)	5	4			2
∞	∞	∞	∞		∞
	4	3	2		1
∞	∞	∞			∞
3	2				0
	G				∞

```

1 for each  $s \in S$  do  $g(s) \leftarrow \infty$ 
2  $Open \leftarrow \{s_0\}$ 
3  $g(s_0) \leftarrow 0; f(s_0) \leftarrow h(s_0)$ 
4 while  $Open \neq \emptyset$ 
5   Extrae un  $u$  desde  $Open$  con menor valor- $f$ 
6   if  $u$  es objetivo return  $u$ 
7   for each  $v \in Succ(u)$  do
        1  $cost_v = g(u) + c(u, v)$ 
        2 if  $cost_v \geq g(v)$  return
        3  $parent(v) \leftarrow u$ 
        4  $g(v) \leftarrow cost_v$ 
        5  $f(v) \leftarrow g(v) + h(v)$ 
        6 if  $v \in Open$  then Reordenar  $Open$ 
        7 else Insertar  $v$  en  $Open$ 

```

6	6	6	1		∞
S					
0	(0,0)	5			3
6	1	∞			∞
(0,0)	5	(1,0)	4		2
∞	∞	∞	∞		∞
	4	3	2		1
∞	∞	∞			
3	2				
				G	0

```

1 for each  $s \in S$  do  $g(s) \leftarrow \infty$ 
2  $Open \leftarrow \{s_0\}$ 
3  $g(s_0) \leftarrow 0; f(s_0) \leftarrow h(s_0)$ 
4 while  $Open \neq \emptyset$ 
5   Extrae un  $u$  desde  $Open$  con menor valor- $f$ 
6   if  $u$  es objetivo return  $u$ 
7   for each  $v \in Succ(u)$  do
        1  $cost_v = g(u) + c(u, v)$ 
        2 if  $cost_v \geq g(v)$  return
        3  $parent(v) \leftarrow u$ 
        4  $g(v) \leftarrow cost_v$  (highlighted)
        5  $f(v) \leftarrow g(v) + h(v)$ 
        6 if  $v \in Open$  then Reordenar  $Open$ 
        7 else Insertar  $v$  en  $Open$ 

```

6	6	6	1		∞
S					
0	(0,0)	5			3
6	1	2			∞
(0,0)	5	(1,0)	4		2
∞	∞	∞	∞	∞	∞
	4	3	2	1	
∞	∞	∞			
3	2				
				G	∞
				0	

```

1 for each  $s \in S$  do  $g(s) \leftarrow \infty$ 
2  $Open \leftarrow \{s_0\}$ 
3  $g(s_0) \leftarrow 0; f(s_0) \leftarrow h(s_0)$ 
4 while  $Open \neq \emptyset$ 
5   Extrae un  $u$  desde  $Open$  con menor valor- $f$ 
6   if  $u$  es objetivo return  $u$ 
7   for each  $v \in Succ(u)$  do
        1  $cost_v = g(u) + c(u, v)$ 
        2 if  $cost_v \geq g(v)$  return
        3  $parent(v) \leftarrow u$ 
        4  $g(v) \leftarrow cost_v$ 
        5  $f(v) \leftarrow g(v) + h(v)$ 
        6 if  $v \in Open$  then Reordenar  $Open$ 
        7 else Insertar  $v$  en  $Open$ 

```

	6	6	6	1		∞
	S					
0		(0,0)		5		3
6	1	6	2			∞
(0,0)	5	(1,0)	4			2
	∞	∞	∞			∞
	4	3	2			1
	∞	∞				∞
3		2				0
	G					

1 **for each** $s \in \mathcal{S}$ **do** $g(s) \leftarrow \infty$

2 $Open \leftarrow \{s_0\}$

3 $g(s_0) \leftarrow 0; f(s_0) \leftarrow h(s_0)$

4 **while** $Open \neq \emptyset$

5 Extrae un u desde $Open$ con menor valor- f

6 **if** u es objetivo **return** u

7 **for each** $v \in Succ(u)$ **do**

1 $cost_v = g(u) + c(u, v)$

2 **if** $cost_v \geq g(v)$ **return**

3 $parent(v) \leftarrow u$

4 $g(v) \leftarrow cost_v$

5 $f(v) \leftarrow g(v) + h(v)$

6 **if** $v \in Open$ **then** Reordenar $Open$

7 **else** Insertar v en $Open$

6	6	6	1		∞
S					
0	(0,0)	5			3
6	1	6	2		∞
(0,0)	5	(1,0)	4		2
	∞	∞	∞		∞
	4	3	2		1
	∞	∞			∞
	3	2			0
	G				∞

```

1 for each  $s \in S$  do  $g(s) \leftarrow \infty$ 
2  $Open \leftarrow \{s_0\}$ 
3  $g(s_0) \leftarrow 0; f(s_0) \leftarrow h(s_0)$ 
4 while  $Open \neq \emptyset$ 
5   Extrae un  $u$  desde  $Open$  con menor valor- $f$ 
6   if  $u$  es objetivo return  $u$ 
7   for each  $v \in Succ(u)$  do
        1  $cost_v = g(u) + c(u, v)$ 
        2 if  $cost_v \geq g(v)$  return
        3  $parent(v) \leftarrow u$ 
        4  $g(v) \leftarrow cost_v$ 
        5  $f(v) \leftarrow g(v) + h(v)$ 
        6 if  $v \in Open$  then Reordenar  $Open$ 
        7 else Insertar  $v$  en  $Open$ 

```

6 S 0	6 (0,0) 5	1 5		∞ 3
6 (0,0) 5	6 (1,0) 4			∞ 2
				∞ 1
			4 3 2 1	
				∞ 0

```

1 for each  $s \in S$  do  $g(s) \leftarrow \infty$ 
2  $Open \leftarrow \{s_0\}$ 
3  $g(s_0) \leftarrow 0; f(s_0) \leftarrow h(s_0)$ 
4 while  $Open \neq \emptyset$ 
5   Extrae un  $u$  desde  $Open$  con menor valor- $f$ 
6   if  $u$  es objetivo return  $u$ 
7   for each  $v \in Succ(u)$  do
        1  $cost_v = g(u) + c(u, v)$ 
        2 if  $cost_v \geq g(v)$  return
        3  $parent(v) \leftarrow u$ 
        4  $g(v) \leftarrow cost_v$ 
        5  $f(v) \leftarrow g(v) + h(v)$ 
        6 if  $v \in Open$  then Reordenar  $Open$ 
        7 else Insertar  $v$  en  $Open$ 

```

6	6	6	1		∞
S					
0	(0,0)	5			3
6	1	6	2		∞
(0,0)	5	(1,0)	4		2
∞	∞	∞	∞		∞
4		3	2		1
∞	∞	∞			
3		2			
G					0

```

1 for each  $s \in S$  do  $g(s) \leftarrow \infty$ 
2  $Open \leftarrow \{s_0\}$ 
3  $g(s_0) \leftarrow 0; f(s_0) \leftarrow h(s_0)$ 
4 while  $Open \neq \emptyset$ 
5   Extrae un  $u$  desde  $Open$  con menor valor- $f$ 
6   if  $u$  es objetivo return  $u$ 
7   for each  $v \in Succ(u)$  do
        1  $cost_v = g(u) + c(u, v)$ 
        2 if  $cost_v \geq g(v)$  return
        3  $parent(v) \leftarrow u$ 
        4  $g(v) \leftarrow cost_v$ 
        5  $f(v) \leftarrow g(v) + h(v)$ 
        6 if  $v \in Open$  then Reordenar  $Open$ 
        7 else Insertar  $v$  en  $Open$ 

```

	6	6	6	1		
	s					
	0	(0,0)	5			3
	6	1	6	2		∞
	(0,0)	5	(1,0)	4		2
	∞		∞			∞
	4	(1,1)	3		2	1
	∞		∞			
	3		2			
	G					0

```

1 for each  $s \in S$  do  $g(s) \leftarrow \infty$ 
2  $Open \leftarrow \{s_0\}$ 
3  $g(s_0) \leftarrow 0; f(s_0) \leftarrow h(s_0)$ 
4 while  $Open \neq \emptyset$ 
5   Extrae un  $u$  desde  $Open$  con menor valor- $f$ 
6   if  $u$  es objetivo return  $u$ 
7   for each  $v \in Succ(u)$  do
        1  $cost_v = g(u) + c(u, v)$ 
        2 if  $cost_v \geq g(v)$  return
        3  $parent(v) \leftarrow u$ 
        4  $g(v) \leftarrow cost_v$  (highlighted)
        5  $f(v) \leftarrow g(v) + h(v)$ 
        6 if  $v \in Open$  then Reordenar  $Open$ 
        7 else Insertar  $v$  en  $Open$ 

```

	6	6	6	1		∞
	S					
0		(0,0)		5		3
6	1	6	2			∞
(0,0)	5	(1,0)	4			2
	∞		3		∞	∞
	4	(1,1)	3		2	1
	∞		∞			
	G				0	∞

```

1 for each  $s \in S$  do  $g(s) \leftarrow \infty$ 
2  $Open \leftarrow \{s_0\}$ 
3  $g(s_0) \leftarrow 0; f(s_0) \leftarrow h(s_0)$ 
4 while  $Open \neq \emptyset$ 
5   Extrae un  $u$  desde  $Open$  con menor valor- $f$ 
6   if  $u$  es objetivo return  $u$ 
7   for each  $v \in Succ(u)$  do
        1  $cost_v = g(u) + c(u, v)$ 
        2 if  $cost_v \geq g(v)$  return
        3  $parent(v) \leftarrow u$ 
        4  $g(v) \leftarrow cost_v$ 
        5  $f(v) \leftarrow g(v) + h(v)$ 
        6 if  $v \in Open$  then Reordenar  $Open$ 
        7 else Insertar  $v$  en  $Open$ 

```

6	6	6	1		∞
s					
0	(0,0)	5			3
6	1	6	2		∞
(0,0)	5	(1,0)	4		2
∞	6	3		∞	∞
4	(1,1)	3		2	1
∞	∞	∞			
3		2			0
			G		∞

1 **for each** $s \in \mathcal{S}$ **do** $g(s) \leftarrow \infty$

2 $Open \leftarrow \{s_0\}$

3 $g(s_0) \leftarrow 0; f(s_0) \leftarrow h(s_0)$

4 **while** $Open \neq \emptyset$

5 Extrae un u desde $Open$ con menor valor- f

6 **if** u es objetivo **return** u

7 **for each** $v \in Succ(u)$ **do**

1 $cost_v = g(u) + c(u, v)$

2 **if** $cost_v \geq g(v)$ **return**

3 $parent(v) \leftarrow u$

4 $g(v) \leftarrow cost_v$

5 $f(v) \leftarrow g(v) + h(v)$

6 **if** $v \in Open$ **then** Reordenar $Open$

7 **else** Insertar v en $Open$

6	6	6	1		∞
S					
0	(0,0)	5			3
6	1	6	2		∞
(0,0)	5	(1,0)	4		2
∞	6	3		∞	∞
4	(1,1)	3		2	1
∞		∞			
3		2			
G					0

```

1 for each  $s \in S$  do  $g(s) \leftarrow \infty$ 
2  $Open \leftarrow \{s_0\}$ 
3  $g(s_0) \leftarrow 0; f(s_0) \leftarrow h(s_0)$ 
4 while  $Open \neq \emptyset$ 
5   Extrae un  $u$  desde  $Open$  con menor valor- $f$ 
6   if  $u$  es objetivo return  $u$ 
7   for each  $v \in Succ(u)$  do
        1  $cost_v = g(u) + c(u, v)$ 
        2 if  $cost_v \geq g(v)$  return
        3  $parent(v) \leftarrow u$ 
        4  $g(v) \leftarrow cost_v$ 
        5  $f(v) \leftarrow g(v) + h(v)$ 
        6 if  $v \in Open$  then Reordenar  $Open$ 
        7 else Insertar  $v$  en  $Open$ 

```

	6	6	6	1		∞
	S					
	0	(0,0)	5			3
6	1	6	2			∞
(0,0)	5	(1,0)	4			2
	∞	6	3			∞
	4	(1,1)	3			1
	∞	∞				∞
	3		2			0
	∞			G		

```

1 for each  $s \in S$  do  $g(s) \leftarrow \infty$ 
2  $Open \leftarrow \{s_0\}$ 
3  $g(s_0) \leftarrow 0; f(s_0) \leftarrow h(s_0)$ 
4 while  $Open \neq \emptyset$ 
5   Extrae un  $u$  desde  $Open$  con menor valor- $f$ 
6   if  $u$  es objetivo return  $u$ 
7   for each  $v \in Succ(u)$  do
        1  $cost_v = g(u) + c(u, v)$ 
        2 if  $cost_v \geq g(v)$  return
        3  $parent(v) \leftarrow u$ 
        4  $g(v) \leftarrow cost_v$ 
        5  $f(v) \leftarrow g(v) + h(v)$ 
        6 if  $v \in Open$  then Reordenar  $Open$ 
        7 else Insertar  $v$  en  $Open$ 

```

6	6	6	1		∞
s					
0	(0,0)	5			3
6	1	6	2		∞
(0,0)	5	(1,0)	4		2
∞	6	3	∞		∞
4	(1,1)	3		2	1
∞	∞	∞			
3		2			0
			G		∞

- 1 **for each** $s \in \mathcal{S}$ **do** $g(s) \leftarrow \infty$
- 2 $Open \leftarrow \{s_0\}$
- 3 $g(s_0) \leftarrow 0; f(s_0) \leftarrow h(s_0)$
- 4 **while** $Open \neq \emptyset$
- 5 Extrae un u desde $Open$ con menor valor- f
- 6 **if** u es objetivo **return** u
- 7 **for each** $v \in Succ(u)$ **do**
 - 1 $cost_v = g(u) + c(u, v)$
 - 2 **if** $cost_v \geq g(v)$ **return**
 - 3 $parent(v) \leftarrow u$
 - 4 $g(v) \leftarrow cost_v$
 - 5 $f(v) \leftarrow g(v) + h(v)$
 - 6 **if** $v \in Open$ **then** Reordenar $Open$
 - 7 **else** Insertar v en $Open$

6	6	6	1		∞
S					
0	(0,0)	5			3
6	1	6	2		∞
(0,0)	5	(1,0)	4		2
∞	6	3		∞	∞
4	(1,1)	3		2	1
∞		∞			
3		2			
G					0

```

1 for each  $s \in S$  do  $g(s) \leftarrow \infty$ 
2  $Open \leftarrow \{s_0\}$ 
3  $g(s_0) \leftarrow 0; f(s_0) \leftarrow h(s_0)$ 
4 while  $Open \neq \emptyset$ 
5   Extrae un  $u$  desde  $Open$  con menor valor- $f$ 
6   if  $u$  es objetivo return  $u$ 
7   for each  $v \in Succ(u)$  do
        1  $cost_v = g(u) + c(u, v)$ 
        2 if  $cost_v \geq g(v)$  return
        3  $parent(v) \leftarrow u$ 
        4  $g(v) \leftarrow cost_v$ 
        5  $f(v) \leftarrow g(v) + h(v)$ 
        6 if  $v \in Open$  then Reordenar  $Open$ 
        7 else Insertar  $v$  en  $Open$ 

```

6	6	6	1		∞
S					
0	(0,0)	5			3
6	1	6	2		∞
(0,0)	5	(1,0)	4		2
∞	6	3		∞	∞
4	(1,1)	3		2	1
∞	∞				
3		2			
∞	G				0

```

1 for each  $s \in S$  do  $g(s) \leftarrow \infty$ 
2  $Open \leftarrow \{s_0\}$ 
3  $g(s_0) \leftarrow 0; f(s_0) \leftarrow h(s_0)$ 
4 while  $Open \neq \emptyset$ 
5   Extrae un  $u$  desde  $Open$  con menor valor- $f$ 
6   if  $u$  es objetivo return  $u$ 
7   for each  $v \in Succ(u)$  do
        1  $cost_v = g(u) + c(u, v)$ 
        2 if  $cost_v \geq g(v)$  return
        3  $parent(v) \leftarrow u$ 
        4  $g(v) \leftarrow cost_v$ 
        5  $f(v) \leftarrow g(v) + h(v)$ 
        6 if  $v \in Open$  then Reordenar  $Open$ 
        7 else Insertar  $v$  en  $Open$ 

```

6	6	6	1		∞
S					
0	(0,0)	5			3
6	1	6	2		∞
(0,0)	5	(1,0)	4		2
∞	6	3		∞	∞
(0,1)	4	(1,1)	3	2	1
∞	∞	∞			
3		2			
				G	0

```

1 for each  $s \in S$  do  $g(s) \leftarrow \infty$ 
2  $Open \leftarrow \{s_0\}$ 
3  $g(s_0) \leftarrow 0; f(s_0) \leftarrow h(s_0)$ 
4 while  $Open \neq \emptyset$ 
5   Extrae un  $u$  desde  $Open$  con menor valor- $f$ 
6   if  $u$  es objetivo return  $u$ 
7   for each  $v \in Succ(u)$  do
        1  $cost_v = g(u) + c(u, v)$ 
        2 if  $cost_v \geq g(v)$  return
        3  $parent(v) \leftarrow u$ 
        4  $g(v) \leftarrow cost_v$  (highlighted)
        5  $f(v) \leftarrow g(v) + h(v)$ 
        6 if  $v \in Open$  then Reordenar  $Open$ 
        7 else Insertar  $v$  en  $Open$ 

```

	6	6	6	1		
	S					∞
	0	(0,0)	5			3
	6	1	6	2		∞
	(0,0)	5	(1,0)	4		2
	2	6	3		∞	∞
	(0,1)	4	(1,1)	3		2
	∞	∞				1
	3		2			0
				G	∞	

```

1 for each  $s \in S$  do  $g(s) \leftarrow \infty$ 
2  $Open \leftarrow \{s_0\}$ 
3  $g(s_0) \leftarrow 0; f(s_0) \leftarrow h(s_0)$ 
4 while  $Open \neq \emptyset$ 
5   Extrae un  $u$  desde  $Open$  con menor valor- $f$ 
6   if  $u$  es objetivo return  $u$ 
7   for each  $v \in Succ(u)$  do
        1  $cost_v = g(u) + c(u, v)$ 
        2 if  $cost_v \geq g(v)$  return
        3  $parent(v) \leftarrow u$ 
        4  $g(v) \leftarrow cost_v$ 
        5  $f(v) \leftarrow g(v) + h(v)$ 
        6 if  $v \in Open$  then Reordenar  $Open$ 
        7 else Insertar  $v$  en  $Open$ 

```

	6	6	6	1		∞
	S					
	0	(0,0)	5			3
6	1	6	2			∞
(0,0)	5	(1,0)	4			2
6	2	6	3		∞	∞
(0,1)	4	(1,1)	3		2	1
	∞		∞			
	3		2			
				G	∞	0

```

1 for each  $s \in S$  do  $g(s) \leftarrow \infty$ 
2  $Open \leftarrow \{s_0\}$ 
3  $g(s_0) \leftarrow 0; f(s_0) \leftarrow h(s_0)$ 
4 while  $Open \neq \emptyset$ 
5   Extrae un  $u$  desde  $Open$  con menor valor- $f$ 
6   if  $u$  es objetivo return  $u$ 
7   for each  $v \in Succ(u)$  do
        1  $cost_v = g(u) + c(u, v)$ 
        2 if  $cost_v \geq g(v)$  return
        3  $parent(v) \leftarrow u$ 
        4  $g(v) \leftarrow cost_v$ 
        5  $f(v) \leftarrow g(v) + h(v)$ 
        6 if  $v \in Open$  then Reordenar  $Open$ 
        7 else Insertar  $v$  en  $Open$ 

```

	6	6	6	1		∞
	S					
	0	(0,0)	5			3
	6	1	6	2		∞
	(0,0)	5	(1,0)	4		2
	6	2	6	3	∞	∞
	(0,1)	4	(1,1)	3	2	1
	∞	∞	∞			
		3		2		
					G	0

```

1 for each  $s \in S$  do  $g(s) \leftarrow \infty$ 
2  $Open \leftarrow \{s_0\}$ 
3  $g(s_0) \leftarrow 0; f(s_0) \leftarrow h(s_0)$ 
4 while  $Open \neq \emptyset$ 
5   Extrae un  $u$  desde  $Open$  con menor valor- $f$ 
6   if  $u$  es objetivo return  $u$ 
7   for each  $v \in Succ(u)$  do
        1  $cost_v = g(u) + c(u, v)$ 
        2 if  $cost_v \geq g(v)$  return
        3  $parent(v) \leftarrow u$ 
        4  $g(v) \leftarrow cost_v$ 
        5  $f(v) \leftarrow g(v) + h(v)$ 
        6 if  $v \in Open$  then Reordenar  $Open$ 
        7 else Insertar  $v$  en  $Open$ 

```

	6	6	6	1		∞
	S					
	0	(0,0)	5			3
6	1	6	2			∞
(0,0)	5	(1,0)	4			2
6	2	6	3		∞	∞
(0,1)	4	(1,1)	3		2	1
	∞		∞			
	3		2			
					G	0

```

1 for each  $s \in S$  do  $g(s) \leftarrow \infty$ 
2  $Open \leftarrow \{s_0\}$ 
3  $g(s_0) \leftarrow 0; f(s_0) \leftarrow h(s_0)$ 
4 while  $Open \neq \emptyset$ 
5   Extrae un  $u$  desde  $Open$  con menor valor- $f$ 
6   if  $u$  es objetivo return  $u$ 
7   for each  $v \in Succ(u)$  do
        1  $cost_v = g(u) + c(u, v)$ 
        2 if  $cost_v \geq g(v)$  return
        3  $parent(v) \leftarrow u$ 
        4  $g(v) \leftarrow cost_v$ 
        5  $f(v) \leftarrow g(v) + h(v)$ 
        6 if  $v \in Open$  then Reordenar  $Open$ 
        7 else Insertar  $v$  en  $Open$ 

```

	6	6	6	1		∞
	S					
	0	(0,0)	5			3
6	1	6	2			∞
(0,0)	5	(1,0)	4			2
6	2	6	3		∞	∞
(0,1)	4	(1,1)	3		2	1
	∞		∞			
	3		2			
					G	0

```

1 for each  $s \in S$  do  $g(s) \leftarrow \infty$ 
2  $Open \leftarrow \{s_0\}$ 
3  $g(s_0) \leftarrow 0; f(s_0) \leftarrow h(s_0)$ 
4 while  $Open \neq \emptyset$ 
5   Extrae un  $u$  desde  $Open$  con menor valor- $f$ 
6   if  $u$  es objetivo return  $u$ 
7   for each  $v \in Succ(u)$  do
        1  $cost_v = g(u) + c(u, v)$ 
        2 if  $cost_v \geq g(v)$  return
        3  $parent(v) \leftarrow u$ 
        4  $g(v) \leftarrow cost_v$ 
        5  $f(v) \leftarrow g(v) + h(v)$ 
        6 if  $v \in Open$  then Reordenar  $Open$ 
        7 else Insertar  $v$  en  $Open$ 

```

6	6	6	1		∞
0	(0,0)	5	2		3
6	1	6	2		∞
(0,0)	5	(1,0)	4		2
6	2	6	3		∞
(0,1)	4	(1,1)	3	(1,2)	2
∞	∞	∞		0	∞
3	2			0	

```

1 for each  $s \in S$  do  $g(s) \leftarrow \infty$ 
2  $Open \leftarrow \{s_0\}$ 
3  $g(s_0) \leftarrow 0; f(s_0) \leftarrow h(s_0)$ 
4 while  $Open \neq \emptyset$ 
5   Extrae un  $u$  desde  $Open$  con menor valor- $f$ 
6   if  $u$  es objetivo return  $u$ 
7   for each  $v \in Succ(u)$  do
        1  $cost_v = g(u) + c(u, v)$ 
        2 if  $cost_v \geq g(v)$  return
        3  $parent(v) \leftarrow u$ 
        4  $g(v) \leftarrow cost_v$  (highlighted)
        5  $f(v) \leftarrow g(v) + h(v)$ 
        6 if  $v \in Open$  then Reordenar  $Open$ 
        7 else Insertar  $v$  en  $Open$ 

```

6	6	6	1		∞
S					
0	(0,0)	5			3
6	1	6	2		∞
(0,0)	5	(1,0)	4		2
6	2	6	3	4	∞
(0,1)	4	(1,1)	3	(1,2)	2
∞	∞	∞			1
3		2			0
			G		∞

```

1 for each  $s \in S$  do  $g(s) \leftarrow \infty$ 
2  $Open \leftarrow \{s_0\}$ 
3  $g(s_0) \leftarrow 0; f(s_0) \leftarrow h(s_0)$ 
4 while  $Open \neq \emptyset$ 
5   Extrae un  $u$  desde  $Open$  con menor valor- $f$ 
6   if  $u$  es objetivo return  $u$ 
7   for each  $v \in Succ(u)$  do
        1  $cost_v = g(u) + c(u, v)$ 
        2 if  $cost_v \geq g(v)$  return
        3  $parent(v) \leftarrow u$ 
        4  $g(v) \leftarrow cost_v$ 
        5  $f(v) \leftarrow g(v) + h(v)$ 
        6 if  $v \in Open$  then Reordenar  $Open$ 
        7 else Insertar  $v$  en  $Open$ 

```

6	6	6	1			∞
s						
0	(0,0)	5				3
6	1	6	2			∞
(0,0)	5	(1,0)	4			2
6	2	6	3	6	4	∞
(0,1)	4	(1,1)	3	(1,2)	2	1
∞		∞				
3		2				
					G	0

```

1 for each  $s \in S$  do  $g(s) \leftarrow \infty$ 
2  $Open \leftarrow \{s_0\}$ 
3  $g(s_0) \leftarrow 0; f(s_0) \leftarrow h(s_0)$ 
4 while  $Open \neq \emptyset$ 
5   Extrae un  $u$  desde  $Open$  con menor valor- $f$ 
6   if  $u$  es objetivo return  $u$ 
7   for each  $v \in Succ(u)$  do
        1  $cost_v = g(u) + c(u, v)$ 
        2 if  $cost_v \geq g(v)$  return
        3  $parent(v) \leftarrow u$ 
        4  $g(v) \leftarrow cost_v$ 
        5  $f(v) \leftarrow g(v) + h(v)$ 
        6 if  $v \in Open$  then Reordenar  $Open$ 
        7 else Insertar  $v$  en  $Open$ 

```

6	6	6	1		∞
6	1	6	2		∞
(0,0)	5	(1,0)	4		2
6	2	6	3	6	4
(0,1)	4	(1,1)	3	(1,2)	2
∞	∞		2		0

```

1 for each  $s \in S$  do  $g(s) \leftarrow \infty$ 
2  $Open \leftarrow \{s_0\}$ 
3  $g(s_0) \leftarrow 0; f(s_0) \leftarrow h(s_0)$ 
4 while  $Open \neq \emptyset$ 
5   Extrae un  $u$  desde  $Open$  con menor valor- $f$ 
6   if  $u$  es objetivo return  $u$ 
7   for each  $v \in Succ(u)$  do
        1  $cost_v = g(u) + c(u, v)$ 
        2 if  $cost_v \geq g(v)$  return
        3  $parent(v) \leftarrow u$ 
        4  $g(v) \leftarrow cost_v$ 
        5  $f(v) \leftarrow g(v) + h(v)$ 
        6 if  $v \in Open$  then Reordenar  $Open$ 
        7 else Insertar  $v$  en  $Open$ 

```

6	6	6	1			∞
s						
0	(0,0)	5				3
6	1	6	2			∞
(0,0)	5	(1,0)	4			2
6	2	6	3	6	4	∞
(0,1)	4	(1,1)	3	(1,2)	2	1
	∞		∞			
	3		2			
					G	0

```

1 for each  $s \in S$  do  $g(s) \leftarrow \infty$ 
2  $Open \leftarrow \{s_0\}$ 
3  $g(s_0) \leftarrow 0; f(s_0) \leftarrow h(s_0)$ 
4 while  $Open \neq \emptyset$ 
5   Extrae un  $u$  desde  $Open$  con menor valor- $f$ 
6   if  $u$  es objetivo return  $u$ 
7   for each  $v \in Succ(u)$  do
        1  $cost_v = g(u) + c(u, v)$ 
        2 if  $cost_v \geq g(v)$  return
        3  $parent(v) \leftarrow u$ 
        4  $g(v) \leftarrow cost_v$ 
        5  $f(v) \leftarrow g(v) + h(v)$ 
        6 if  $v \in Open$  then Reordenar  $Open$ 
        7 else Insertar  $v$  en  $Open$ 

```

6	6	6	1		∞
s					
0	(0,0)	5			3
6	1	6	2		∞
(0,0)	5	(1,0)	4		2
6	2	6	3	6	4
(0,1)	4	(1,1)	3	(1,2)	2
	∞		∞		1
	3	(1,2)	2		
				G	0

- 1 **for each** $s \in \mathcal{S}$ **do** $g(s) \leftarrow \infty$
- 2 $Open \leftarrow \{s_0\}$
- 3 $g(s_0) \leftarrow 0; f(s_0) \leftarrow h(s_0)$
- 4 **while** $Open \neq \emptyset$
- 5 Extrae un u desde $Open$ con menor valor- f
- 6 **if** u es objetivo **return** u
- 7 **for each** $v \in Succ(u)$ **do**
 - 1 $cost_v = g(u) + c(u, v)$
 - 2 **if** $cost_v \geq g(v)$ **return**
 - 3 $parent(v) \leftarrow u$
 - 4 $g(v) \leftarrow cost_v$
 - 5 $f(v) \leftarrow g(v) + h(v)$
 - 6 **if** $v \in Open$ **then** Reordenar $Open$
 - 7 **else** Insertar v en $Open$

6	6	6	1			∞
S						
0	(0,0)	5				3
6	1	6	2			∞
(0,0)	5	(1,0)	4			2
6	2	6	3	6	4	∞
(0,1)	4	(1,1)	3	(1,2)	2	1
∞			4			
3	(1,2)	2				
						∞
					G	0

- 1 **for each** $s \in \mathcal{S}$ **do** $g(s) \leftarrow \infty$
- 2 $Open \leftarrow \{s_0\}$
- 3 $g(s_0) \leftarrow 0; f(s_0) \leftarrow h(s_0)$
- 4 **while** $Open \neq \emptyset$
- 5 Extrae un u desde $Open$ con menor valor- f
- 6 **if** u es objetivo **return** u
- 7 **for each** $v \in Succ(u)$ **do**
 - 1 $cost_v = g(u) + c(u, v)$
 - 2 **if** $cost_v \geq g(v)$ **return**
 - 3 $parent(v) \leftarrow u$
 - 4 $g(v) \leftarrow cost_v$
 - 5 $f(v) \leftarrow g(v) + h(v)$
 - 6 **if** $v \in Open$ **then** Reordenar $Open$
 - 7 **else** Insertar v en $Open$

6	6	6	1			∞
S						
0	(0,0)	5				3
6	1	6	2			∞
(0,0)	5	(1,0)	4			2
6	2	6	3	6	4	∞
(0,1)	4	(1,1)	3	(1,2)	2	1
∞	6	4				
3	(1,2)	2				
						∞
						0
					G	

```

1 for each  $s \in S$  do  $g(s) \leftarrow \infty$ 
2  $Open \leftarrow \{s_0\}$ 
3  $g(s_0) \leftarrow 0; f(s_0) \leftarrow h(s_0)$ 
4 while  $Open \neq \emptyset$ 
5   Extrae un  $u$  desde  $Open$  con menor valor- $f$ 
6   if  $u$  es objetivo return  $u$ 
7   for each  $v \in Succ(u)$  do
        1  $cost_v = g(u) + c(u, v)$ 
        2 if  $cost_v \geq g(v)$  return
        3  $parent(v) \leftarrow u$ 
        4  $g(v) \leftarrow cost_v$ 
        5  $f(v) \leftarrow g(v) + h(v)$ 
        6 if  $v \in Open$  then Reordenar  $Open$ 
        7 else Insertar  $v$  en  $Open$ 

```

	6	6	6	1			∞
	s						
	0	(0,0)	5				3
6	1	6	2				∞
(0,0)	5	(1,0)	4				2
6	2	6	3	6	4		∞
(0,1)	4	(1,1)	3	(1,2)	2		1
∞	6	4					
3	(1,2)	2					
						G	0

```

1 for each  $s \in S$  do  $g(s) \leftarrow \infty$ 
2  $Open \leftarrow \{s_0\}$ 
3  $g(s_0) \leftarrow 0; f(s_0) \leftarrow h(s_0)$ 
4 while  $Open \neq \emptyset$ 
5   Extrae un  $u$  desde  $Open$  con menor valor- $f$ 
6   if  $u$  es objetivo return  $u$ 
7   for each  $v \in Succ(u)$  do
        1  $cost_v = g(u) + c(u, v)$ 
        2 if  $cost_v \geq g(v)$  return
        3  $parent(v) \leftarrow u$ 
        4  $g(v) \leftarrow cost_v$ 
        5  $f(v) \leftarrow g(v) + h(v)$ 
        6 if  $v \in Open$  then Reordenar  $Open$ 
        7 else Insertar  $v$  en  $Open$ 

```

	6	6	6	1		∞
	S					
	0	(0,0)	5			3
6	1	6	2			∞
(0,0)	5	(1,0)	4			2
6	2	6	3	6	4	∞
(0,1)	4	(1,1)	3	(1,2)	2	1
	∞	6	4			
	3	(1,2)	2			
					G	0

```

1 for each  $s \in S$  do  $g(s) \leftarrow \infty$ 
2  $Open \leftarrow \{s_0\}$ 
3  $g(s_0) \leftarrow 0; f(s_0) \leftarrow h(s_0)$ 
4 while  $Open \neq \emptyset$ 
5   Extrae un  $u$  desde  $Open$  con menor valor- $f$ 
6   if  $u$  es objetivo return  $u$ 
7   for each  $v \in Succ(u)$  do
        1  $cost_v = g(u) + c(u, v)$ 
        2 if  $cost_v \geq g(v)$  return
        3  $parent(v) \leftarrow u$ 
        4  $g(v) \leftarrow cost_v$ 
        5  $f(v) \leftarrow g(v) + h(v)$ 
        6 if  $v \in Open$  then Reordenar  $Open$ 
        7 else Insertar  $v$  en  $Open$ 

```

6	6	6	1			∞
S						
0	(0,0)	5				3
6	1	6	2			∞
(0,0)	5	(1,0)	4			2
6	2	6	3	6	4	∞
(0,1)	4	(1,1)	3	(1,2)	2	1
∞	6	4				
3	(1,2)	2				
				G	∞	0

```

1 for each  $s \in S$  do  $g(s) \leftarrow \infty$ 
2  $Open \leftarrow \{s_0\}$ 
3  $g(s_0) \leftarrow 0; f(s_0) \leftarrow h(s_0)$ 
4 while  $Open \neq \emptyset$ 
5   Extrae un  $u$  desde  $Open$  con menor valor- $f$ 
6   if  $u$  es objetivo return  $u$ 
7   for each  $v \in Succ(u)$  do
        1  $cost_v = g(u) + c(u, v)$ 
        2 if  $cost_v \geq g(v)$  return
        3  $parent(v) \leftarrow u$ 
        4  $g(v) \leftarrow cost_v$ 
        5  $f(v) \leftarrow g(v) + h(v)$ 
        6 if  $v \in Open$  then Reordenar  $Open$ 
        7 else Insertar  $v$  en  $Open$ 

```

6	6	6	1			∞
S						
0	(0,0)	5				3
6	1	6	2			∞
(0,0)	5	(1,0)	4			2
6	2	6	3	6	4	∞
(0,1)	4	(1,1)	3	(1,2)	2	1
∞	6	4				
3	(1,2)	2				
					G	0

```

1 for each  $s \in S$  do  $g(s) \leftarrow \infty$ 
2  $Open \leftarrow \{s_0\}$ 
3  $g(s_0) \leftarrow 0; f(s_0) \leftarrow h(s_0)$ 
4 while  $Open \neq \emptyset$ 
5   Extrae un  $u$  desde  $Open$  con menor valor- $f$ 
6   if  $u$  es objetivo return  $u$ 
7   for each  $v \in Succ(u)$  do
        1  $cost_v = g(u) + c(u, v)$ 
        2 if  $cost_v \geq g(v)$  return
        3  $parent(v) \leftarrow u$ 
        4  $g(v) \leftarrow cost_v$ 
        5  $f(v) \leftarrow g(v) + h(v)$ 
        6 if  $v \in Open$  then Reordenar  $Open$ 
        7 else Insertar  $v$  en  $Open$ 

```

The diagram illustrates the A* search algorithm's execution on a grid. The grid has columns labeled 0, (0,0), 5, (1,0), 4, (0,1), 4, (1,1), 3, (1,2), 2, and (1,2). The last two columns are blacked out.

6	6	6	1							∞
S										
0	(0,0)	5								3
6	1	6	2							∞
(0,0)	5	(1,0)	4							2
6	2	6	3	6	4					∞
(0,1)	4	(1,1)	3	(1,2)	2					1
∞	6	4								
3	(1,2)	2								
G										0

The first row contains values 6, 6, 6, 1, followed by four empty cells and ∞ . The second row contains a large orange circle labeled 'S' at its center, followed by five empty cells. The third row contains 0, (0,0), 5, followed by three empty cells. The fourth row contains 6, 1, 6, 2, followed by three empty cells. The fifth row contains (0,0), 5, (1,0), 4, followed by three empty cells. The sixth row contains 6, 2, 6, 3, 6, 4, followed by three empty cells. The seventh row contains (0,1), 4, (1,1), 3, (1,2), 2, followed by three empty cells. The eighth row contains ∞ , 6, 4, followed by three empty cells. The ninth row contains 3, (1,2), 2, followed by three empty cells. The tenth row contains a large orange circle labeled 'G' at its center, followed by five empty cells.

```

1 for each  $s \in S$  do  $g(s) \leftarrow \infty$ 
2  $Open \leftarrow \{s_0\}$ 
3  $g(s_0) \leftarrow 0; f(s_0) \leftarrow h(s_0)$ 
4 while  $Open \neq \emptyset$ 
5   Extrae un  $u$  desde  $Open$  con menor valor- $f$ 
6   if  $u$  es objetivo return  $u$ 
7   for each  $v \in Succ(u)$  do
        1  $cost_v = g(u) + c(u, v)$ 
        2 if  $cost_v \geq g(v)$  return
        3  $parent(v) \leftarrow u$ 
        4  $g(v) \leftarrow cost_v$ 
        5  $f(v) \leftarrow g(v) + h(v)$ 
        6 if  $v \in Open$  then Reordenar  $Open$ 
        7 else Insertar  $v$  en  $Open$ 

```

6	6	6	1			∞
s						
0	(0,0)	5				3
6	1	6	2			∞
(0,0)	5	(1,0)	4			2
6	2	6	3	6	4	∞
(0,1)	4	(1,1)	3	(1,2)	2	(2,2) 1
∞	6	4				
3	(1,2)	2				
					G	0

```

1 for each  $s \in S$  do  $g(s) \leftarrow \infty$ 
2  $Open \leftarrow \{s_0\}$ 
3  $g(s_0) \leftarrow 0; f(s_0) \leftarrow h(s_0)$ 
4 while  $Open \neq \emptyset$ 
5   Extrae un  $u$  desde  $Open$  con menor valor- $f$ 
6   if  $u$  es objetivo return  $u$ 
7   for each  $v \in Succ(u)$  do
        1  $cost_v = g(u) + c(u, v)$ 
        2 if  $cost_v \geq g(v)$  return
        3  $parent(v) \leftarrow u$ 
        4  $g(v) \leftarrow cost_v$  (highlighted)
        5  $f(v) \leftarrow g(v) + h(v)$ 
        6 if  $v \in Open$  then Reordenar  $Open$ 
        7 else Insertar  $v$  en  $Open$ 

```

6	6	6	1			∞
s						
0	(0,0)	5				3
6	1	6	2			∞
(0,0)	5	(1,0)	4			2
6	2	6	3	6	4	5
(0,1)	4	(1,1)	3	(1,2)	2	(2,2) 1
∞	6	4				
3	(1,2)	2				
					G	0

```

1 for each  $s \in S$  do  $g(s) \leftarrow \infty$ 
2  $Open \leftarrow \{s_0\}$ 
3  $g(s_0) \leftarrow 0; f(s_0) \leftarrow h(s_0)$ 
4 while  $Open \neq \emptyset$ 
5   Extrae un  $u$  desde  $Open$  con menor valor- $f$ 
6   if  $u$  es objetivo return  $u$ 
7   for each  $v \in Succ(u)$  do
        1  $cost_v = g(u) + c(u, v)$ 
        2 if  $cost_v \geq g(v)$  return
        3  $parent(v) \leftarrow u$ 
        4  $g(v) \leftarrow cost_v$ 
        5  $f(v) \leftarrow g(v) + h(v)$ 
        6 if  $v \in Open$  then Reordenar  $Open$ 
        7 else Insertar  $v$  en  $Open$ 

```

6	6	6	1			∞
S						
0	(0,0)	5				3
6	1	6	2			∞
(0,0)	5	(1,0)	4			2
6	2	6	3	6	4	6
(0,1)	4	(1,1)	3	(1,2)	2	(2,2)
∞	6	4				∞
3	(1,2)	2				0
				G		

```

1 for each  $s \in S$  do  $g(s) \leftarrow \infty$ 
2  $Open \leftarrow \{s_0\}$ 
3  $g(s_0) \leftarrow 0; f(s_0) \leftarrow h(s_0)$ 
4 while  $Open \neq \emptyset$ 
5   Extrae un  $u$  desde  $Open$  con menor valor- $f$ 
6   if  $u$  es objetivo return  $u$ 
7   for each  $v \in Succ(u)$  do
        1  $cost_v = g(u) + c(u, v)$ 
        2 if  $cost_v \geq g(v)$  return
        3  $parent(v) \leftarrow u$ 
        4  $g(v) \leftarrow cost_v$ 
        5  $f(v) \leftarrow g(v) + h(v)$ 
        6 if  $v \in Open$  then Reordenar  $Open$ 
        7 else Insertar  $v$  en  $Open$ 

```

6	6	6	1			∞
0	(0,0)	5				3
6	1	6	2			∞
(0,0)	5	(1,0)	4			2
6	2	6	3	6	4	5
(0,1)	4	(1,1)	3	(1,2)	2	(2,2) 1
∞	6	4				∞
3	(1,2)	2				0

```

1 for each  $s \in \mathcal{S}$  do  $g(s) \leftarrow \infty$ 
2  $Open \leftarrow \{s_0\}$ 
3  $g(s_0) \leftarrow 0; f(s_0) \leftarrow h(s_0)$ 
4 while  $Open \neq \emptyset$ 
5   Extrae un  $u$  desde  $Open$  con menor valor- $f$ 
6   if  $u$  es objetivo return  $u$ 
7   for each  $v \in Succ(u)$  do
        1  $cost_v = g(u) + c(u, v)$ 
        2 if  $cost_v \geq g(v)$  return
        3  $parent(v) \leftarrow u$ 
        4  $g(v) \leftarrow cost_v$ 
        5  $f(v) \leftarrow g(v) + h(v)$ 
        6 if  $v \in Open$  then Reordenar  $Open$ 
        7 else Insertar  $v$  en  $Open$ 

```

6	6	6	1			∞
S						
0	(0,0)	5				3
6	1	6	2			∞
(0,0)	5	(1,0)	4			2
6	2	6	3	6	4	6 5
(0,1)	4	(1,1)	3	(1,2)	2	(2,2) 1
∞	6	4				∞
3	(1,2)	2				∞
						G 0

```

1 for each  $s \in S$  do  $g(s) \leftarrow \infty$ 
2  $Open \leftarrow \{s_0\}$ 
3  $g(s_0) \leftarrow 0; f(s_0) \leftarrow h(s_0)$ 
4 while  $Open \neq \emptyset$ 
5   Extrae un  $u$  desde  $Open$  con menor valor- $f$ 
6   if  $u$  es objetivo return  $u$ 
7   for each  $v \in Succ(u)$  do
        1  $cost_v = g(u) + c(u, v)$ 
        2 if  $cost_v \geq g(v)$  return
        3  $parent(v) \leftarrow u$ 
        4  $g(v) \leftarrow cost_v$ 
        5  $f(v) \leftarrow g(v) + h(v)$ 
        6 if  $v \in Open$  then Reordenar  $Open$ 
        7 else Insertar  $v$  en  $Open$ 

```

	6	6	6	1			∞
	S						
	0	(0,0)	5				3
6	1	6	2				∞
(0,0)	5	(1,0)	4				2
6	2	6	3	6	4	6	5
(0,1)	4	(1,1)	3	(1,2)	2	(2,2)	1
∞	6	4					∞
3	(1,2)	2					0
	G						

```

1 for each  $s \in S$  do  $g(s) \leftarrow \infty$ 
2  $Open \leftarrow \{s_0\}$ 
3  $g(s_0) \leftarrow 0; f(s_0) \leftarrow h(s_0)$ 
4 while  $Open \neq \emptyset$ 
5   Extrae un  $u$  desde  $Open$  con menor valor- $f$ 
6   if  $u$  es objetivo return  $u$ 
7   for each  $v \in Succ(u)$  do
        1  $cost_v = g(u) + c(u, v)$ 
        2 if  $cost_v \geq g(v)$  return
        3  $parent(v) \leftarrow u$ 
        4  $g(v) \leftarrow cost_v$ 
        5  $f(v) \leftarrow g(v) + h(v)$ 
        6 if  $v \in Open$  then Reordenar  $Open$ 
        7 else Insertar  $v$  en  $Open$ 

```

2024-1

```

1 for each  $s \in S$  do  $g(s) \leftarrow \infty$ 
2  $Open \leftarrow \{s_0\}$ 
3  $g(s_0) \leftarrow 0; f(s_0) \leftarrow h(s_0)$ 
4 while  $Open \neq \emptyset$ 
5   Extrae un  $u$  desde  $Open$  con menor valor- $f$ 
6   if  $u$  es objetivo return  $u$ 
7   for each  $v \in Succ(u)$  do
        1  $cost_v = g(u) + c(u, v)$ 
        2 if  $cost_v \geq g(v)$  return
        3  $parent(v) \leftarrow u$ 
        4  $g(v) \leftarrow cost_v$  (highlighted)
        5  $f(v) \leftarrow g(v) + h(v)$ 
        6 if  $v \in Open$  then Reordenar  $Open$ 
        7 else Insertar  $v$  en  $Open$ 

```

6	6	6	1				∞
S							3
0	(0,0)	5					
6	1	6	2				6
(0,0)	5	(1,0)	4				
6	2	6	3	6	4	6	5
(0,1)	4	(1,1)	3	(1,2)	2	(2,2)	1
∞	6	4					
3	(1,2)	2					
G							0

```

1 for each  $s \in S$  do  $g(s) \leftarrow \infty$ 
2  $Open \leftarrow \{s_0\}$ 
3  $g(s_0) \leftarrow 0; f(s_0) \leftarrow h(s_0)$ 
4 while  $Open \neq \emptyset$ 
5   Extrae un  $u$  desde  $Open$  con menor valor- $f$ 
6   if  $u$  es objetivo return  $u$ 
7   for each  $v \in Succ(u)$  do
        1  $cost_v = g(u) + c(u, v)$ 
        2 if  $cost_v \geq g(v)$  return
        3  $parent(v) \leftarrow u$ 
        4  $g(v) \leftarrow cost_v$ 
        5  $f(v) \leftarrow g(v) + h(v)$ 
        6 if  $v \in Open$  then Reordenar  $Open$ 
        7 else Insertar  $v$  en  $Open$ 

```

	6	6	6	1			∞
	S						
	0	(0,0)	5				3
6	1	6	2				6
(0,0)	5	(1,0)	4				2
6	2	6	3	6	4	6	5
(0,1)	4	(1,1)	3	(1,2)	2	(2,2)	1
∞	6	4					
3	(1,2)	2					
	G						0

```

1 for each  $s \in \mathcal{S}$  do  $g(s) \leftarrow \infty$ 
2  $Open \leftarrow \{s_0\}$ 
3  $g(s_0) \leftarrow 0; f(s_0) \leftarrow h(s_0)$ 
4 while  $Open \neq \emptyset$ 
5   Extrae un  $u$  desde  $Open$  con menor valor- $f$ 
6   if  $u$  es objetivo return  $u$ 
7   for each  $v \in Succ(u)$  do
        1  $cost_v = g(u) + c(u, v)$ 
        2 if  $cost_v \geq g(v)$  return
        3  $parent(v) \leftarrow u$ 
        4  $g(v) \leftarrow cost_v$ 
        5  $f(v) \leftarrow g(v) + h(v)$ 
        6 if  $v \in Open$  then Reordenar  $Open$ 
        7 else Insertar  $v$  en  $Open$ 
    
```

6	6	6	1			∞
S						3
0	(0,0)	5				
6	1	6	2			6
(0,0)	5	(1,0)	4			2
6	2	6	3	6	4	5
(0,1)	4	(1,1)	3	(1,2)	2	(2,2) 1
∞	6	4				
3	(1,2)	2				
					G	0

```

1 for each  $s \in S$  do  $g(s) \leftarrow \infty$ 
2  $Open \leftarrow \{s_0\}$ 
3  $g(s_0) \leftarrow 0; f(s_0) \leftarrow h(s_0)$ 
4 while  $Open \neq \emptyset$ 
5   Extrae un  $u$  desde  $Open$  con menor valor- $f$ 
6   if  $u$  es objetivo return  $u$ 
7   for each  $v \in Succ(u)$  do
        1  $cost_v = g(u) + c(u, v)$ 
        2 if  $cost_v \geq g(v)$  return
        3  $parent(v) \leftarrow u$ 
        4  $g(v) \leftarrow cost_v$ 
        5  $f(v) \leftarrow g(v) + h(v)$ 
        6 if  $v \in Open$  then Reordenar  $Open$ 
        7 else Insertar  $v$  en  $Open$ 

```

6	6	6	1		
S					∞
0	(0,0)	5			3
6	1	6	2		8 6
(0,0)	5	(1,0)	4		(3,2) 2
6	2	6	3	6	6 5
(0,1)	4	(1,1)	3	(1,2)	2 (2,2) 1
∞	6	4			G 0
3	(1,2)	2			

```

1 for each  $s \in S$  do  $g(s) \leftarrow \infty$ 
2  $Open \leftarrow \{s_0\}$ 
3  $g(s_0) \leftarrow 0; f(s_0) \leftarrow h(s_0)$ 
4 while  $Open \neq \emptyset$ 
5   Extrae un  $u$  desde  $Open$  con menor valor- $f$ 
6   if  $u$  es objetivo return  $u$ 
7   for each  $v \in Succ(u)$  do
        1  $cost_v = g(u) + c(u, v)$ 
        2 if  $cost_v \geq g(v)$  return
        3  $parent(v) \leftarrow u$ 
        4  $g(v) \leftarrow cost_v$ 
        5  $f(v) \leftarrow g(v) + h(v)$ 
        6 if  $v \in Open$  then Reordenar  $Open$ 
        7 else Insertar  $v$  en  $Open$ 

```

6	6	6	1			∞
S						
0	(0,0)	5				3
6	1	6	2			6
(0,0)	5	(1,0)	4			2
6	2	6	3	6	4	5
(0,1)	4	(1,1)	3	(1,2)	2	(2,2) 1
∞	6	4				
3	(1,2)	2				
G						0

```

1 for each  $s \in S$  do  $g(s) \leftarrow \infty$ 
2  $Open \leftarrow \{s_0\}$ 
3  $g(s_0) \leftarrow 0; f(s_0) \leftarrow h(s_0)$ 
4 while  $Open \neq \emptyset$ 
5   Extrae un  $u$  desde  $Open$  con menor valor- $f$ 
6   if  $u$  es objetivo return  $u$ 
7   for each  $v \in Succ(u)$  do
        1  $cost_v = g(u) + c(u, v)$ 
        2 if  $cost_v \geq g(v)$  return
        3  $parent(v) \leftarrow u$ 
        4  $g(v) \leftarrow cost_v$  (4)
        5  $f(v) \leftarrow g(v) + h(v)$ 
        6 if  $v \in Open$  then Reordenar  $Open$ 
        7 else Insertar  $v$  en  $Open$ 

```

6	6	6	1				∞
S							
0	(0,0)	5					3
6	1	6	2				
(0,0)	5	(1,0)	4				6
6	2	6	3	6	4	6	5
(0,1)	4	(1,1)	3	(1,2)	2	(2,2)	1
∞	6	4					
3	(1,2)	2					
G							0

```

1 for each  $s \in S$  do  $g(s) \leftarrow \infty$ 
2  $Open \leftarrow \{s_0\}$ 
3  $g(s_0) \leftarrow 0; f(s_0) \leftarrow h(s_0)$ 
4 while  $Open \neq \emptyset$ 
5   Extrae un  $u$  desde  $Open$  con menor valor- $f$ 
6   if  $u$  es objetivo return  $u$ 
7   for each  $v \in Succ(u)$  do
        1  $cost_v = g(u) + c(u, v)$ 
        2 if  $cost_v \geq g(v)$  return
        3  $parent(v) \leftarrow u$ 
        4  $g(v) \leftarrow cost_v$ 
        5  $f(v) \leftarrow g(v) + h(v)$ 
        6 if  $v \in Open$  then Reordenar  $Open$ 
        7 else Insertar  $v$  en  $Open$ 

```

6	6	6	1			∞
S						
0	(0,0)	5				3
6	1	6	2			8 6
(0,0)	5	(1,0)	4			(3,2) 2
6	2	6	3	6	4	6 5
(0,1)	4	(1,1)	3	(1,2)	2	(2,2) 1
∞	6	4				
3	(1,2)	2				
6	6	G				
						0

1 **for each** $s \in \mathcal{S}$ **do** $g(s) \leftarrow \infty$

2 $Open \leftarrow \{s_0\}$

3 $g(s_0) \leftarrow 0; f(s_0) \leftarrow h(s_0)$

4 **while** $Open \neq \emptyset$

5 Extrae un u desde $Open$ con menor valor- f

6 **if** u es objetivo **return** u

7 **for each** $v \in Succ(u)$ **do**

1 $cost_v = g(u) + c(u, v)$

2 **if** $cost_v \geq g(v)$ **return**

3 $parent(v) \leftarrow u$

4 $g(v) \leftarrow cost_v$

5 $f(v) \leftarrow g(v) + h(v)$

6 **if** $v \in Open$ **then** Reordenar $Open$

7 **else** Insertar v en $Open$

6	6	6	1				∞
S							
0	(0,0)	5					3
6	1	6	2				
(0,0)	5	(1,0)	4				6
6	2	6	3	6	4	6	5
(0,1)	4	(1,1)	3	(1,2)	2	(2,2)	1
∞	6	4					
3	(1,2)	2					
6	6	6	0				
G							
(3,2)							

```

1 for each  $s \in S$  do  $g(s) \leftarrow \infty$ 
2  $Open \leftarrow \{s_0\}$ 
3  $g(s_0) \leftarrow 0; f(s_0) \leftarrow h(s_0)$ 
4 while  $Open \neq \emptyset$ 
5   Extrae un  $u$  desde  $Open$  con menor valor- $f$ 
6   if  $u$  es objetivo return  $u$ 
7   for each  $v \in Succ(u)$  do
        1  $cost_v = g(u) + c(u, v)$ 
        2 if  $cost_v \geq g(v)$  return
        3  $parent(v) \leftarrow u$ 
        4  $g(v) \leftarrow cost_v$ 
        5  $f(v) \leftarrow g(v) + h(v)$ 
        6 if  $v \in Open$  then Reordenar  $Open$ 
        7 else Insertar  $v$  en  $Open$ 

```

	6	6	6	1			∞
	S						
	0	(0,0)	5				3
6	1	6	2				6
(0,0)	5	(1,0)	4				2
6	2	6	3	6	4	6	5
(0,1)	4	(1,1)	3	(1,2)	2	(2,2)	1
	∞	6	4				
	3	(1,2)	2				
	G						0

```

1 for each  $s \in S$  do  $g(s) \leftarrow \infty$ 
2  $Open \leftarrow \{s_0\}$ 
3  $g(s_0) \leftarrow 0; f(s_0) \leftarrow h(s_0)$ 
4 while  $Open \neq \emptyset$ 
5   Extrae un  $u$  desde  $Open$  con menor valor- $f$ 
6   if  $u$  es objetivo return  $u$ 
7   for each  $v \in Succ(u)$  do
        1  $cost_v = g(u) + c(u, v)$ 
        2 if  $cost_v \geq g(v)$  return
        3  $parent(v) \leftarrow u$ 
        4  $g(v) \leftarrow cost_v$ 
        5  $f(v) \leftarrow g(v) + h(v)$ 
        6 if  $v \in Open$  then Reordenar  $Open$ 
        7 else Insertar  $v$  en  $Open$ 

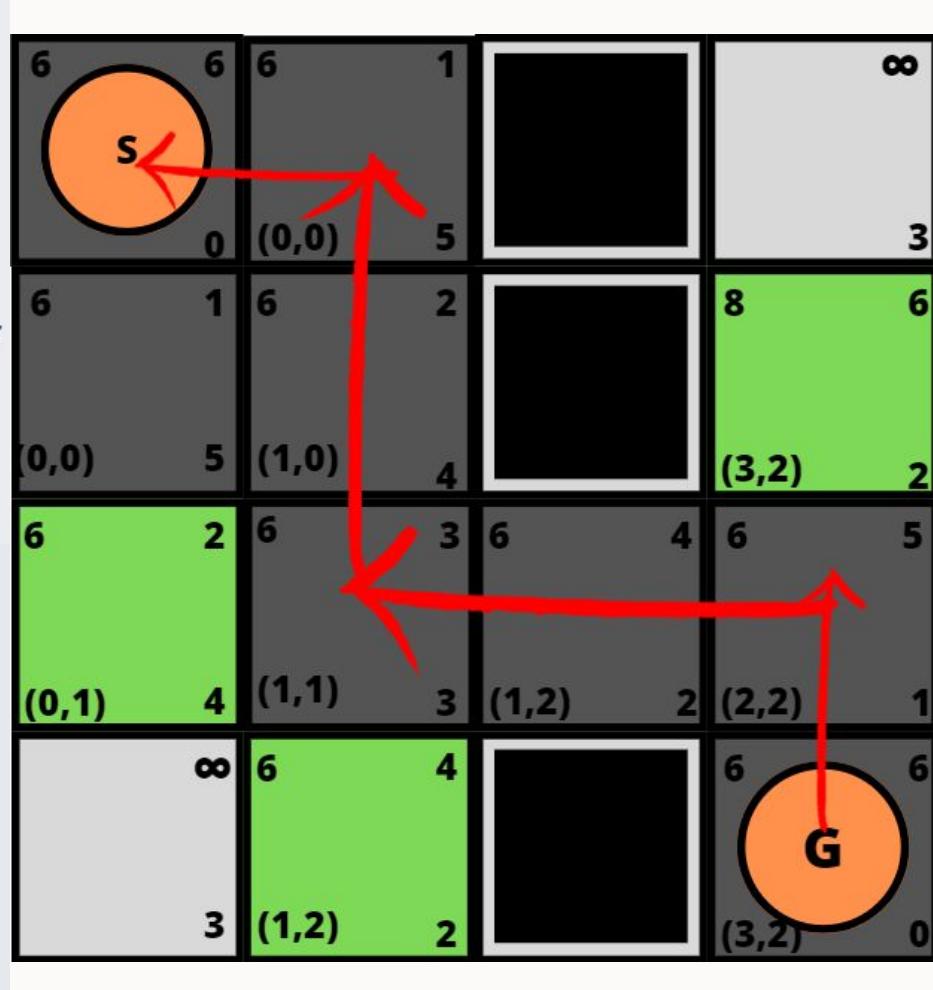
```

6	6	6	1			∞
S						
0	(0,0)	5				3
6	1	6	2			8 6
(0,0)	5	(1,0)	4			(3,2) 2
6	2	6	3	6	4	6 5
(0,1)	4	(1,1)	3	(1,2)	2	(2,2) 1
∞	6	4				
3	(1,2)	2				
6	6	G				0
(3,2)						

```

1 for each  $s \in S$  do  $g(s) \leftarrow \infty$ 
2  $Open \leftarrow \{s_0\}$ 
3  $g(s_0) \leftarrow 0; f(s_0) \leftarrow h(s_0)$ 
4 while  $Open \neq \emptyset$ 
5   Extrae un  $u$  desde  $Open$  con menor valor- $f$ 
6   if  $u$  es objetivo return  $u$ 
7   for each  $v \in Succ(u)$  do
        1  $cost_v = g(u) + c(u, v)$ 
        2 if  $cost_v \geq g(v)$  return
        3  $parent(v) \leftarrow u$ 
        4  $g(v) \leftarrow cost_v$ 
        5  $f(v) \leftarrow g(v) + h(v)$ 
        6 if  $v \in Open$  then Reordenar  $Open$ 
        7 else Insertar  $v$  en  $Open$ 

```





Otro ejemplo de A* - Computerphile - Youtube

A* (A Star) Search Algorithm



¿Por qué A* es óptimo? - Youtube

¿Por qué A* es óptimo? - YouTube



Ayudantía 5

Heurísticas y A*

Por Daniel Toribio y Felipe Espinoza

22 de abril 2024