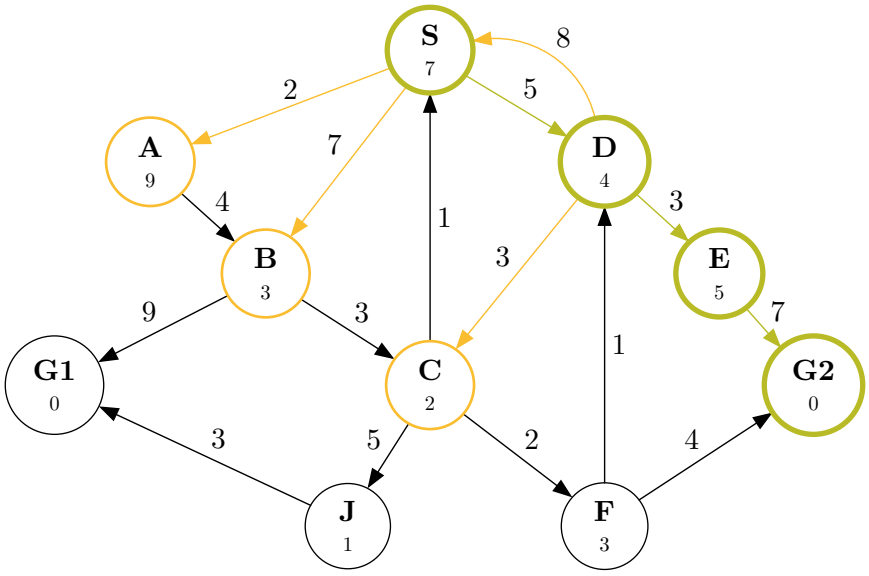
1 E.A.3.1

Implementations in `Rust` and exercises

- https://github.com/CuriousCI/artificial-intelligence

1.1 Ricerca in profondità

DFS (depth first search)



#		azioni	esplorati	frontiera
0	∅	{}	{}	[(S, g: 0, h: 7, f: 7, d: 0),]

1	S	{A, B, D}	{S}	[
				  (A, g: 2, h: 9, f: 11, d: 1),
				  (B, g: 7, h: 3, f: 10, d: 1),
				  (D, g: 5, h: 4, f: 9, d: 1),
				]

2	D	{C, E, S}	{D, S}	[
				  (A, g: 2, h: 9, f: 11, d: 1),
				  (B, g: 7, h: 3, f: 10, d: 1),
				  (C, g: 8, h: 2, f: 10, d: 2),
				  (E, g: 8, h: 5, f: 13, d: 2),
				]

3	E	{G2}	{D, E, S}	[
				  (A, g: 2, h: 9, f: 11, d: 1),
				  (B, g: 7, h: 3, f: 10, d: 1),
				  (C, g: 8, h: 2, f: 10, d: 2),

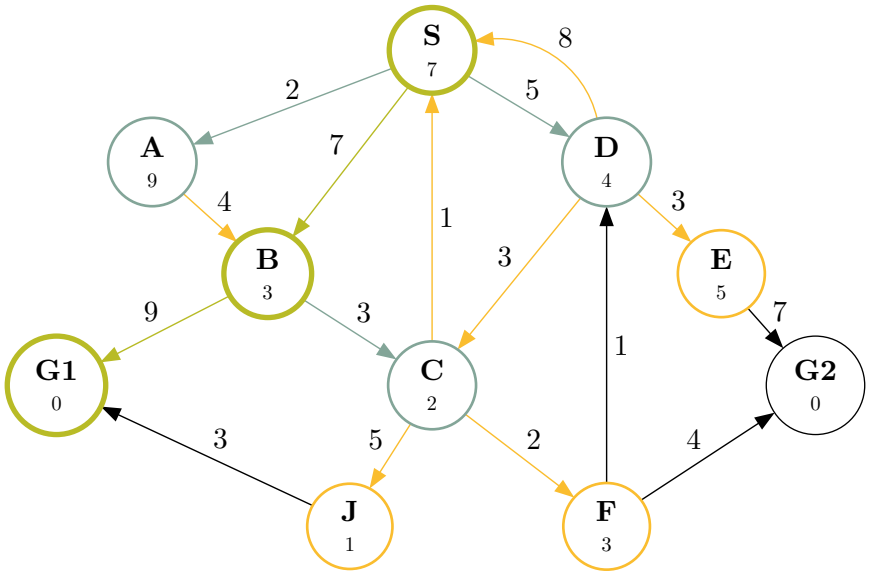
#		azioni	esplorati	frontiera
				(G2, g: 15, h: 0, f: 15, d: 3),]
4	G2	{}	{D, E, G2, S}	is goal

Percorso S → D → E → G2
Costo $5 + 3 + 7 = 15$
Iterazioni 1
Ottimalità il cammino non è ottimale (il costo ottimale è 14)

Generalmente la DFS non garantisce l'ottimalità

1.2 Ricerca in ampiezza

BFS (breadth first search)



#		azioni	esplorati	frontiera
0	∅	{}	{}	[(S, g: 0, h: 7, f: 7, d: 0),]
1	S	{A, B, D}	{S}	[(A, g: 2, h: 9, f: 11, d: 1), (B, g: 7, h: 3, f: 10, d: 1), (D, g: 5, h: 4, f: 9, d: 1),]
2	A	{B}	{A, S}	[(A, g: 2, h: 9, f: 11, d: 1), (B, g: 7, h: 3, f: 10, d: 1), (D, g: 5, h: 4, f: 9, d: 1),]
3	B	{C, G1}	{A, B, S}	[(D, g: 5, h: 4, f: 9, d: 1), (C, g: 10, h: 2, f: 12, d: 2), (G1, g: 16, h: 0, f: 16, d: 2),]
4	D	{C, E, S}	{A, B, D, S}	[(C, g: 10, h: 2, f: 12, d: 2), (G1, g: 16, h: 0, f: 16, d: 2), (E, g: 8, h: 5, f: 13, d: 2),]

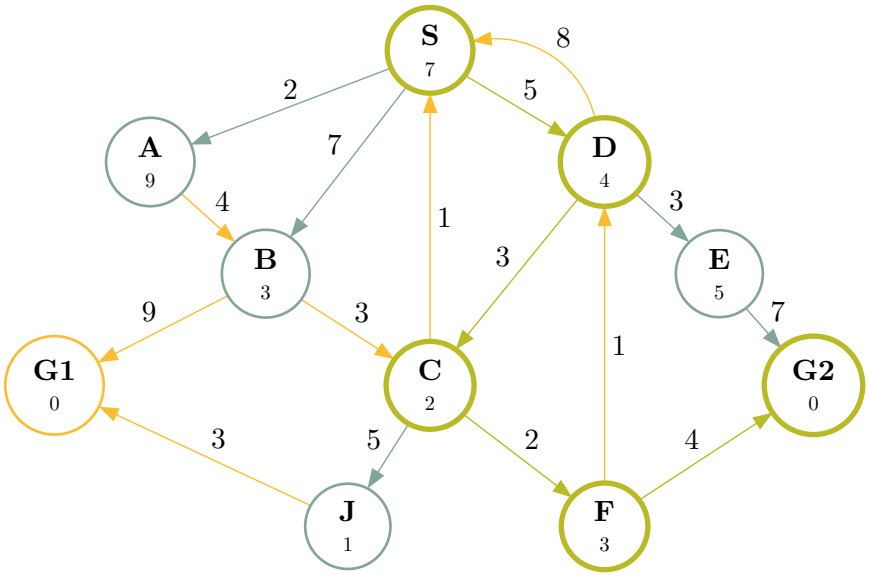
#		azioni	esplorati	frontiera
5	C	{F, J, S}	{A, B, C, D, S}	[(G1, g: 16, h: 0, f: 16, d: 2), (E, g: 8, h: 5, f: 13, d: 2), (F, g: 12, h: 3, f: 15, d: 3), (J, g: 15, h: 1, f: 16, d: 3),]
6	G1	{}	{A, B, D, G1, S}	is goal

Percorso S → B → G1
Costo 7 + 9 = 16
Iterazioni 1
Ottimalità il cammino non è ottimale

Generalmente la BFS garantisce l'ottimalità solo se il costo per tutte le azioni è lo stesso (perché la BFS trova il minor numero di azioni per raggiungere l'obiettivo), e se è possibile trovare un piano (quindi le azioni per un dato stato non sono infinite).

1.3 Ricerca a costi uniformi

Min cost search



#		azioni	esplorati	frontiera
0	∅	{}	{}	{ (S, g: 0, h: 7, f: 7, d: 0), }
1	S	{A, B, D}	{S}	{ (A, g: 2, h: 9, f: 11, d: 1), (D, g: 5, h: 4, f: 9, d: 1), (B, g: 7, h: 3, f: 10, d: 1), }
2	A	{B}	{A, S}	{ (D, g: 5, h: 4, f: 9, d: 1), (B, g: 6, h: 3, f: 9, d: 2), }
3	D	{C, E, S}	{A, D, S}	{ (B, g: 6, h: 3, f: 9, d: 2), (C, g: 8, h: 2, f: 10, d: 2), (E, g: 8, h: 5, f: 13, d: 2), }
4	B	{C, G1}	{A, B, D, S}	{ (C, g: 8, h: 2, f: 10, d: 2), (E, g: 8, h: 5, f: 13, d: 2), (G1, g: 15, h: 0, f: 15, d: 3), }

#		azioni	esplorati	frontiera

5	C	{F, J, S}	{A, B, C, D, S}	{
				(E, g: 8, h: 5, f: 13, d: 2),
				(F, g: 10, h: 3, f: 13, d: 3),
				(J, g: 13, h: 1, f: 14, d: 3),
				(G1, g: 15, h: 0, f: 15, d: 3),
				}

6	E	{G2}	{A, B, C, D, E, S}	{
				(F, g: 10, h: 3, f: 13, d: 3),
				(J, g: 13, h: 1, f: 14, d: 3),
				(G1, g: 15, h: 0, f: 15, d: 3),
				(G2, g: 15, h: 0, f: 15, d: 3),
				}

7	F	{D, G2}	{A, B, C, D, E, F, S}	{
				(J, g: 13, h: 1, f: 14, d: 3),
				(G2, g: 14, h: 0, f: 14, d: 4),
				(G1, g: 15, h: 0, f: 15, d: 3),
				}

8	J	{G1}	{A, B, C, D, E, F, J, S}	{
				(G2, g: 14, h: 0, f: 14, d: 4),
				(G1, g: 15, h: 0, f: 15, d: 3),
				}

9	G2	{}	{A, B, C, D, E, F, G2, J, S}	is goal

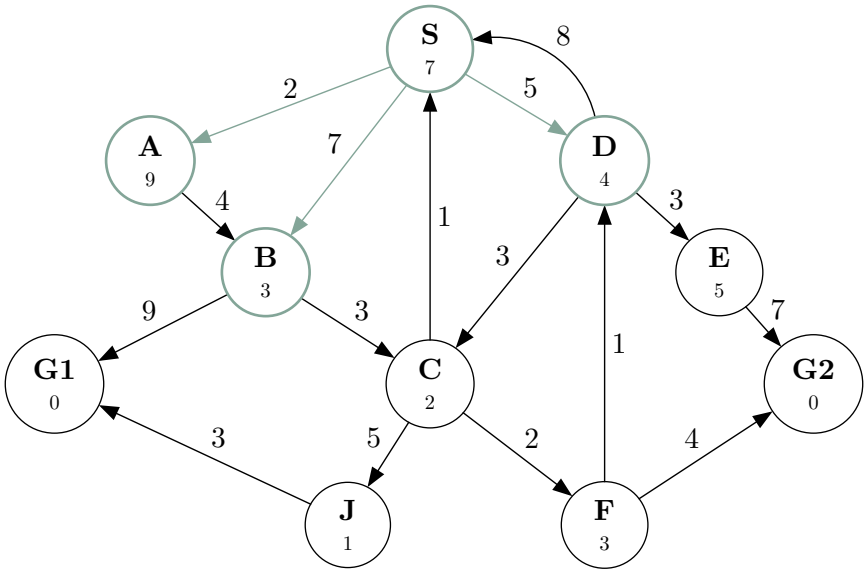
Percorso $S \to D \to C \to F \to G2$
Costo $5 + 3 + 2 + 4 = 14$
Iterazioni 1
Ottimalità il costo del cammino è ottimale

L'algoritmo min cost trova sempre il cammino ottimale (se l'albero di ricerca è finito).

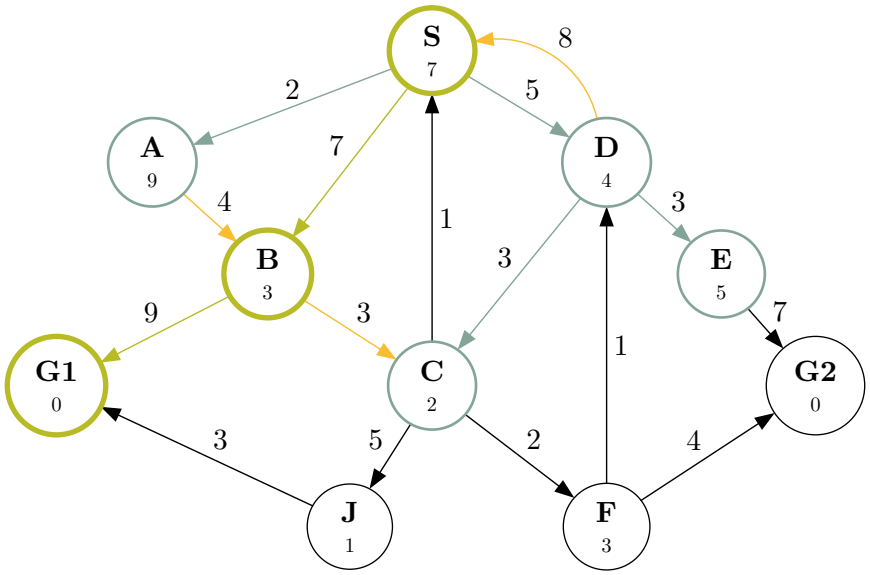
1.4 Ricerca ad approfondimento iterativo

Iterative deepening search

1.4.1 Iterazione 1



1.4.2 Iterazione 2



#		azioni	esplorati	frontiera
0	∅	{}	{}	[(S, g: 0, h: 7, f: 7, d: 0),]

1	S	{A, B, D}	{S}	[
				  (A, g: 2, h: 9, f: 11, d: 1),
				  (B, g: 7, h: 3, f: 10, d: 1),
				  (D, g: 5, h: 4, f: 9, d: 1),
				]

2	D	{}	{D, S}	[
				  (A, g: 2, h: 9, f: 11, d: 1),
				  (B, g: 7, h: 3, f: 10, d: 1),
				]

3	B	{}	{B, D, S}	[(A, g: 2, h: 9, f: 11, d: 1),]
4	A	{}	{A, B, D, S}	[]
5	∅	{}	{}	[(S, g: 0, h: 7, f: 7, d: 0),]

6	S	{A, B, D}	{S}	[
				  (A, g: 2, h: 9, f: 11, d: 1),
				  (B, g: 7, h: 3, f: 10, d: 1),
				  (D, g: 5, h: 4, f: 9, d: 1),
				]

7	D	{C, E, S}	{D, S}	[
				  (A, g: 2, h: 9, f: 11, d: 1),
				  (B, g: 7, h: 3, f: 10, d: 1),
				  (C, g: 8, h: 2, f: 10, d: 2),
				  (E, g: 8, h: 5, f: 13, d: 2),
				]

8	E	{}	{D, E, S}	[
				  (A, g: 2, h: 9, f: 11, d: 1),
				  (B, g: 7, h: 3, f: 10, d: 1),
				  (C, g: 8, h: 2, f: 10, d: 2),
				]

9	C	{}	{C, D, E, S}	[
				  (A, g: 2, h: 9, f: 11, d: 1),
				  (B, g: 7, h: 3, f: 10, d: 1),
				]

10	B	{C, G1}	{B, C, D, E, S}	[
				  (A, g: 2, h: 9, f: 11, d: 1),
				  (G1, g: 16, h: 0, f: 16, d: 2),
				]

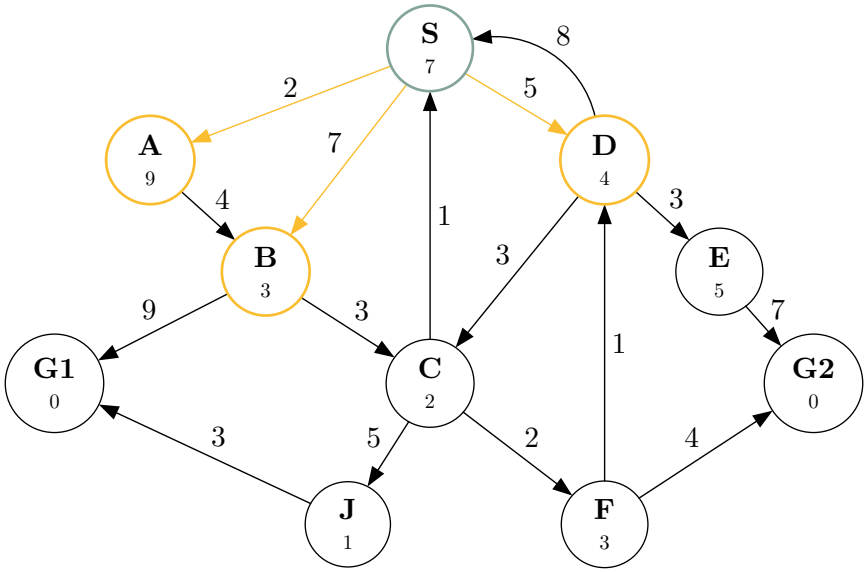
11	G1	{}	{B, C, D, E, G1, S}	is goal

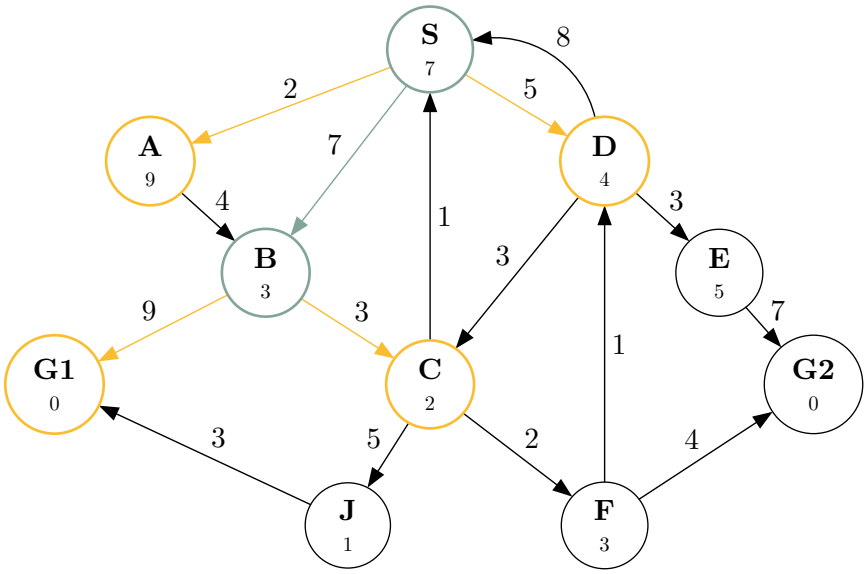
Percorso S $\rightarrow$ B $\rightarrow$ G1
Costo $7 + 9 = 16$
Iterazioni 2
Ottimalità il cammino non è ottimale

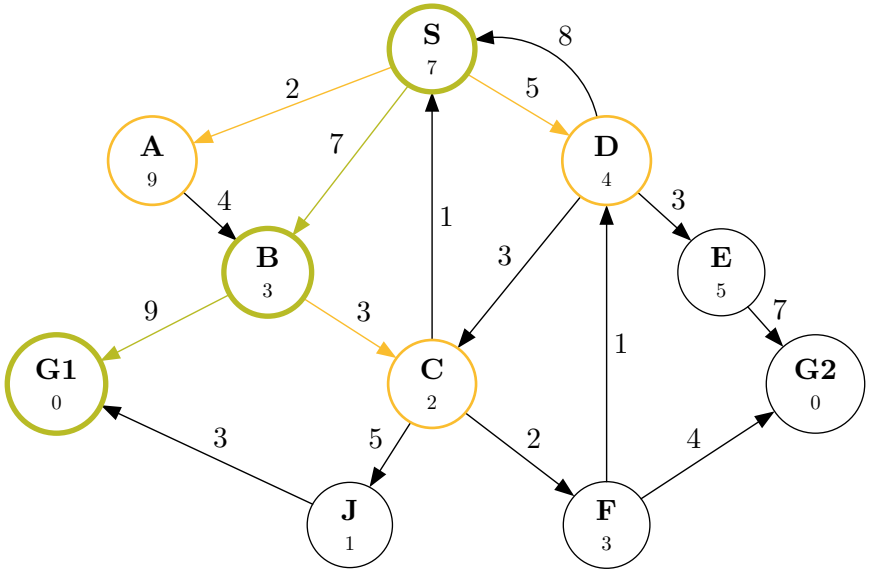
Generalmente non è detto che la ricerca ad approfondimento iterativo trovi un cammino con costo ottimale.

1.5 Ricerca best-first greedy

Best-first greedy search







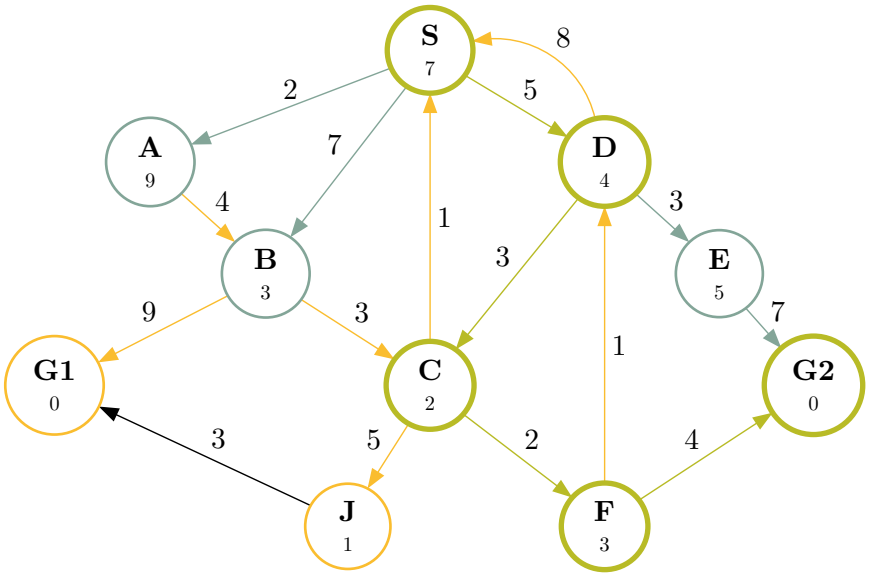
#		azioni	esplorati	frontiera
0	∅	{}	{}	{ (S, g: 0, h: 7, f: 7, d: 0), }
1	S	{A, B, D}	{S}	{ (B, g: 7, h: 3, f: 10, d: 1), (D, g: 5, h: 4, f: 9, d: 1), (A, g: 2, h: 9, f: 11, d: 1), }
2	B	{C, G1}	{B, S}	{ (G1, g: 16, h: 0, f: 16, d: 2), (C, g: 10, h: 2, f: 12, d: 2), (D, g: 5, h: 4, f: 9, d: 1), (A, g: 2, h: 9, f: 11, d: 1), }
3	G1	{}	{B, G1, S}	is goal

Percorso S → B → G1
Costo $7 + 9 = 16$
Iterazioni 1
Ottimalità la soluzione non è ottimale

L'algoritmo best-first non è ottimale, a meno che l'euristica non corrisponde esattamente alla distanza minima per raggiungere un goal. Questo perché il best-first non considera i costi effettivi (che sono maggiori o uguali all'euristica).

1.6 A*

A*



#		azioni	esplorati	frontiera

0	∅	{}	{}	{
				(S, g: 0, h: 7, f: 7, d: 0),
				}

1	S	{A, B, D}	{S}	{
				(D, g: 5, h: 4, f: 9, d: 1),
				(B, g: 7, h: 3, f: 10, d: 1),
				(A, g: 2, h: 9, f: 11, d: 1),
				}

2	D	{C, E, S}	{D, S}	{
				(B, g: 7, h: 3, f: 10, d: 1),
				(C, g: 8, h: 2, f: 10, d: 2),
				(A, g: 2, h: 9, f: 11, d: 1),
				(E: g: 8, h: 5, f: 13, d: 2),
				}

3	B	{C, G1}	{B, D, S}	{
				(C, g: 8, h: 2, f: 10, d: 2),
				(A, g: 2, h: 9, f: 11, d: 1),
				(E: g: 8, h: 5, f: 13, d: 2),
				(G1: g: 16, h: 0, f: 16, d: 2),
				}

4	C	{F, J, S}	{B, C, D, S}	{ (A, g: 2, h: 9, f: 11, d: 1),

#		azioni	esplorati	frontiera
				(E: g: 8, h: 5, f: 13, d: 2), (F: g: 10, h: 3, f: 13, d: 3), (J: g: 13, h: 1, f: 14, d: 3), (G1: g: 16, h: 0, f: 16, d: 2), }

5	A	{B}	{A, B, C, D, S}	{
				(E: g: 8, h: 5, f: 13, d: 2),
				(F: g: 10, h: 3, f: 13, d: 3),
				(J: g: 13, h: 1, f: 14, d: 3),
				(G1: g: 16, h: 0, f: 16, d: 2),
				}

6	E	{G2}	{A, B, C, D, E, S}	{
				(F: g: 10, h: 3, f: 13, d: 3),
				(J: g: 13, h: 1, f: 14, d: 3),
				(G2: g: 15, h: 0, f: 15, d: 3),
				(G1: g: 16, h: 0, f: 16, d: 2),
				}

7	F	{D, G2}	{A, B, C, D, E, F, S}	{
				(G2, g: 14, h: 0, f: 14, d: 4),
				(J: g: 13, h: 1, f: 14, d: 3),
				(G1, g: 16, h: 0, f: 16, d: 2),
				}

8	G2	{}	{A, B, C, D, E, F, G2, S}	is goal

Percorso S → D → C → F → G2
Costo $5 + 3 + 2 + 4 = 14$
Iterazioni 1
Ottimalità la soluzione è ottimale

L'algoritmo A* è ottimale quando la funzione $h(s)$ è consistente (quindi anche ammissibile). In questo esempio l'euristica è ammissibile ma non consistente, il che non basta per avere l'ottimalità di A*.

1.7 Euristica

Come si nota dalla tabella sotto l'euristica **è ammissibile**, perché per ogni stato s vale $h(s)$ è minore o uguale al costo minimo per raggiungere un obiettivo.

s	$h(s)$	$\text{dist}(s)$
A	4	13
B	3	9
C	2	6
D	4	9
E	5	7
F	3	4
G1	0	0
G2	0	0
J	1	3
S	7	14

L'euristica **non è consistente**: $9 = h(\text{A}) > 4 + h(\text{B}) = 4 + 3 = 7$.