Università
degli Studi
della Campania
Luigi Vanvitelli

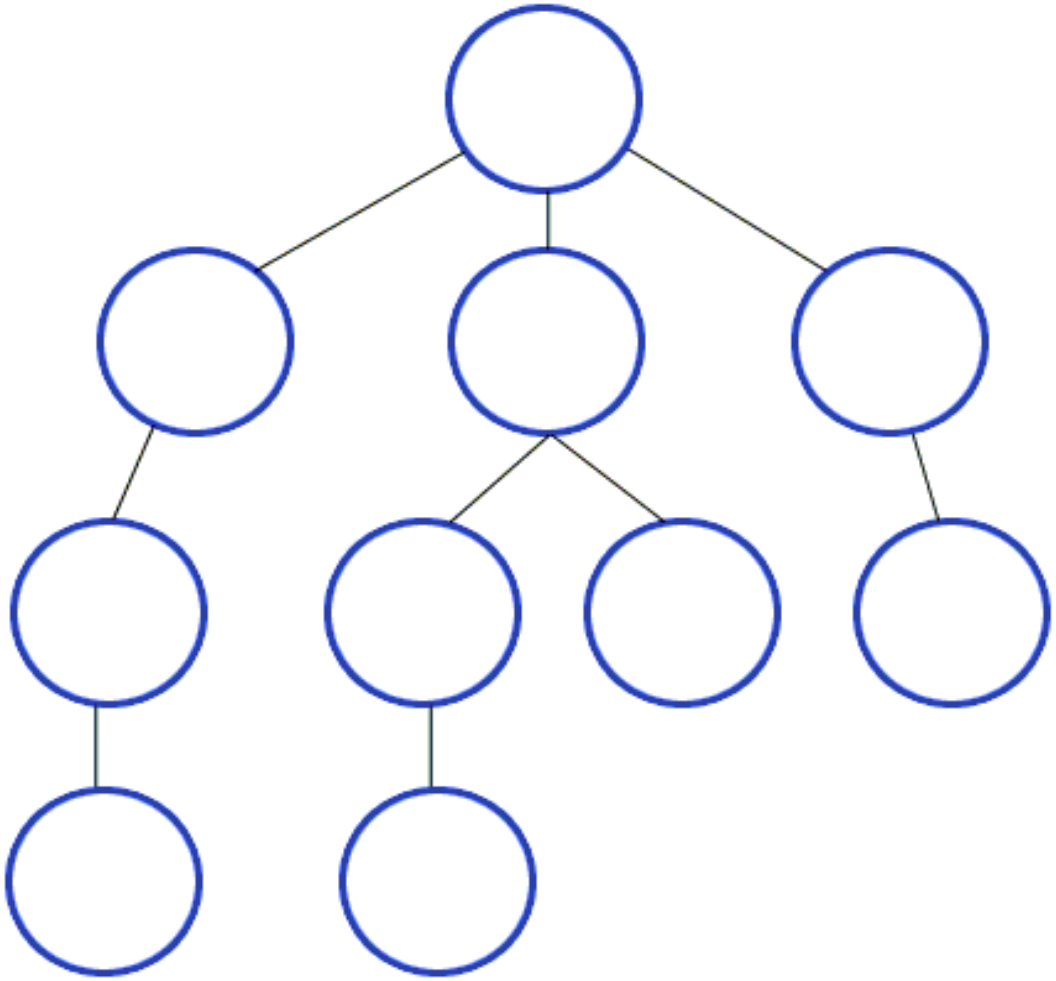
Esercitazione: Algoritmi di Ricerca nello Spazio degli Stati

Corso di Knowledge Engineering and Big Data Intelligence

A.A 2023/2024

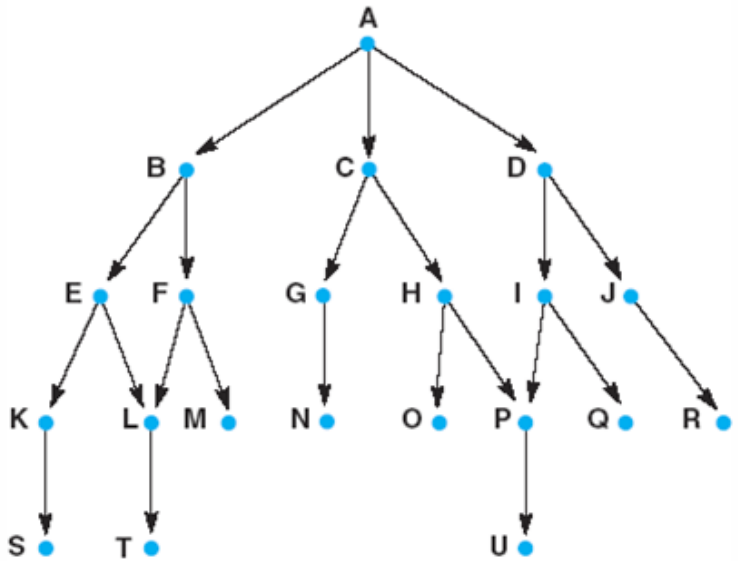
ING. Luigi Colucci Cante

Algoritmo Depth First Search (DFS)



Fonte GIF:
https://it.m.wikipedia.org/wiki/File:Depth-First-Search.gif

Algoritmo Depth First Search (DFS)

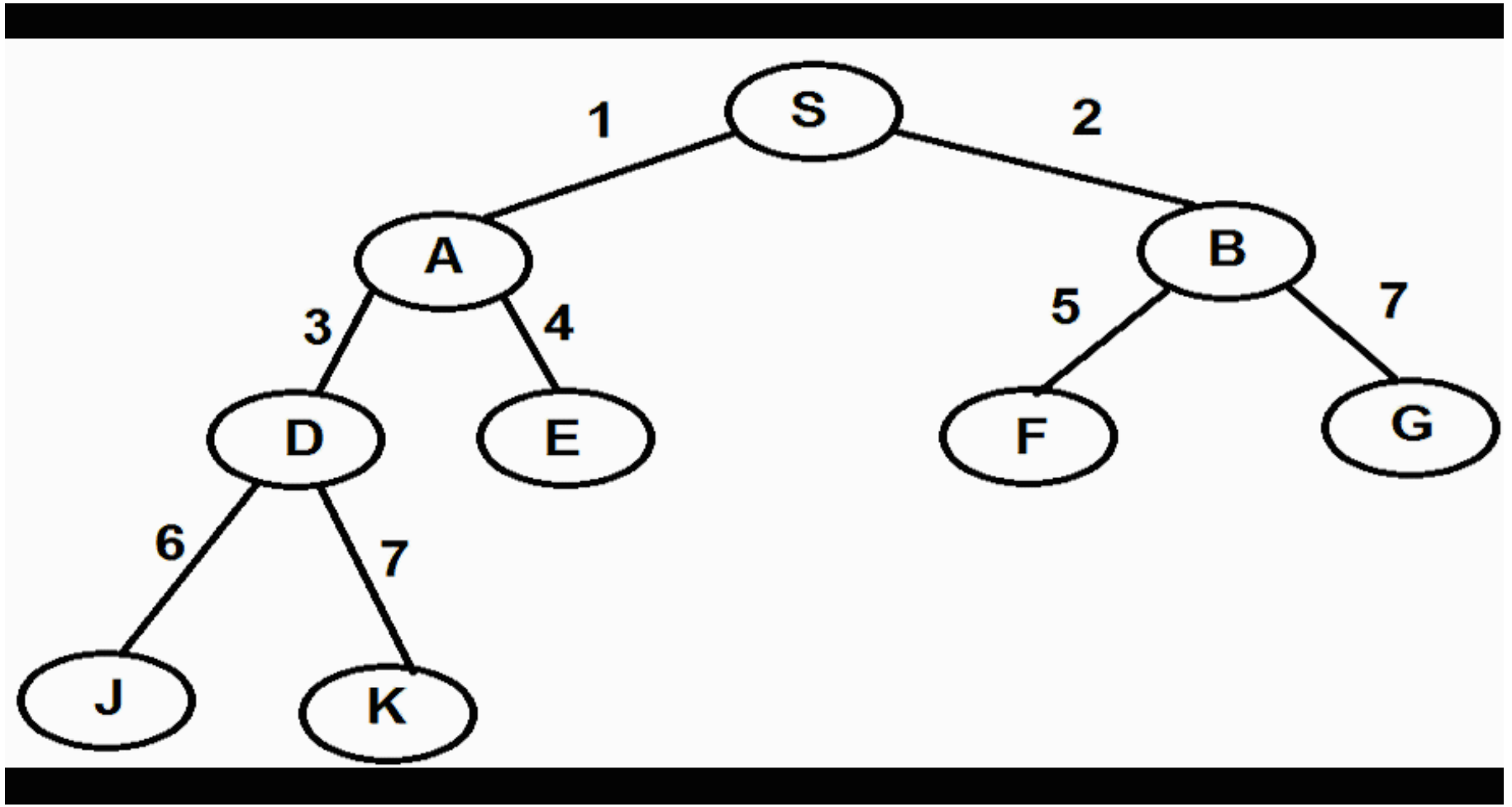


1. **open = [A]; closed = [ ]**
2. **open = [B,C,D]; closed = [A]**
3. **open = [E,F,C,D]; closed = [B,A]**
4. **open = [K,L,F,C,D]; closed = [E,B,A]**
5. **open = [S,L,F,C,D]; closed = [K,E,B,A]**
6. **open = [L,F,C,D]; closed = [S,K,E,B,A]**
7. **open = [T,F,C,D]; closed = [L,S,K,E,B,A]**
8. **open = [F,C,D]; closed = [T,L,S,K,E,B,A]**
9. **open = [M,C,D],** as L is already on **closed**; **closed = [F,T,L,S,K,E,B,A]**
10. **open = [C,D]; closed = [M,F,T,L,S,K,E,B,A]**
11. **open = [G,H,D]; closed = [C,M,F,T,L,S,K,E,B,A]**

Algoritmo Depth First Search (DFS)

https://www.youtube.com/watch?v=iaBEKo5sM7w&t=159s

Algoritmo Breadth First Search (BFS)

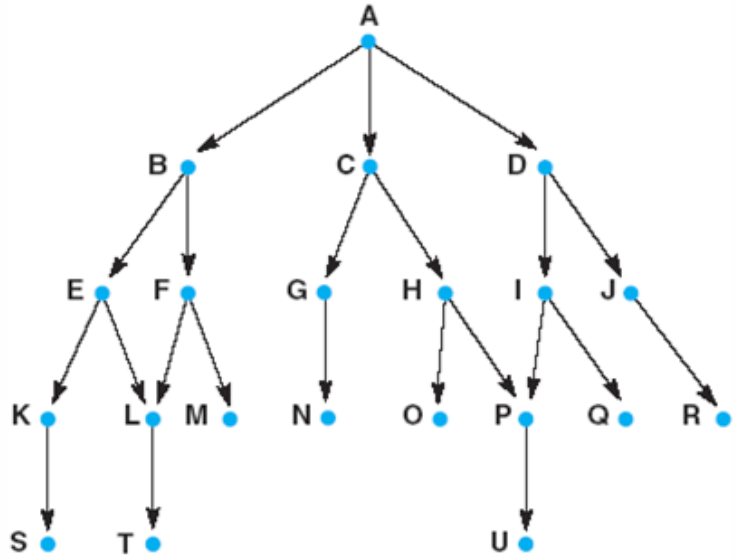


Fonte GIF:
https://github.com/jcink/breadth-first-search-php/blob/master/breadth-first-search.gif

Algoritmo Breadth First Search (BFS)

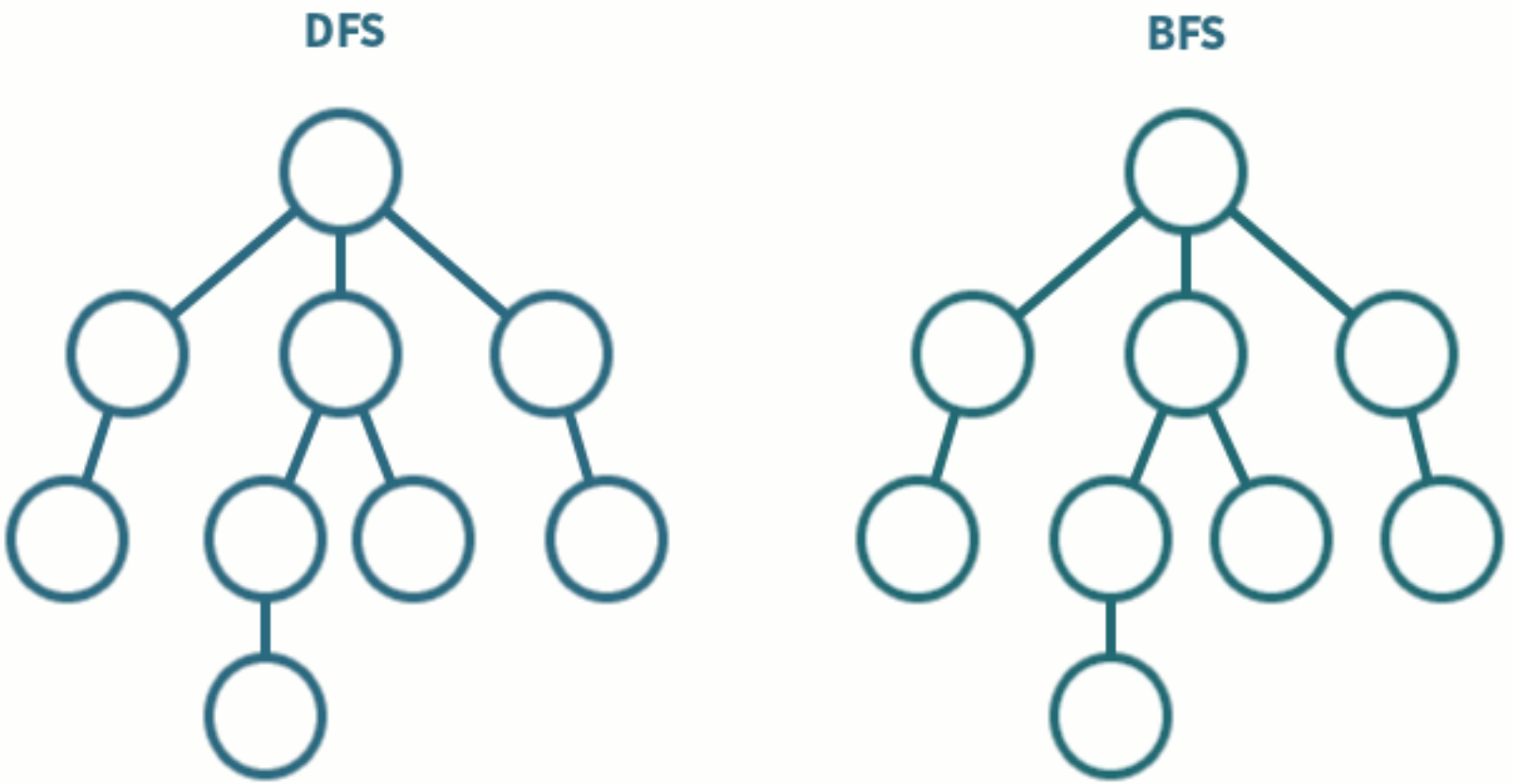
https://www.youtube.com/watch?v=QRq6p9s8NVg

Algoritmo Breadth First Search (BFS)



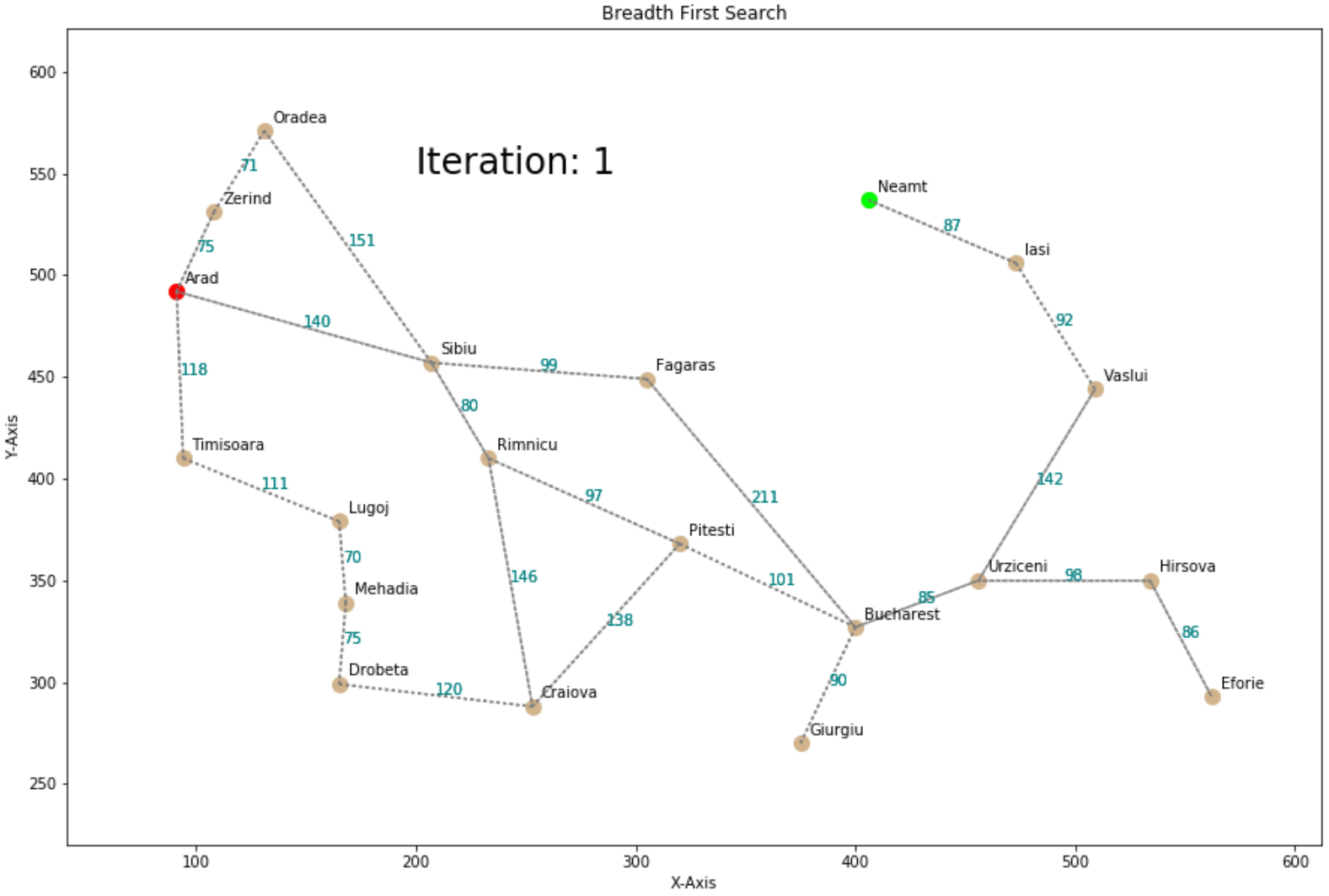
1. **open = [A]; closed = [ ]**
2. **open = [B,C,D]; closed = [A]**
3. **open = [C,D,E,F]; closed = [B,A]**
4. **open = [D,E,F,G,H]; closed = [C,B,A]**
5. **open = [E,F,G,H,I,J]; closed = [D,C,B,A]**
6. **open = [F,G,H,I,J,K,L]; closed = [E,D,C,B,A]**
7. **open = [G,H,I,J,K,L,M]** (as L is already on open); **closed = [F,E,D,C,B,A]**
8. **open = [H,I,J,K,L,M,N]; closed = [G,F,E,D,C,B,A]**
9. and so on until either U is found or **open** = [ ]

Confronto DFS e BFS



Fonte GIF: https://medium.com/@kenny.hom27/breadth-first-vs-depth-first-tree-traversal-in-javascript-48df2ebfc6d1

Confronto con Map Navigation 1/3



Punto di Partenza: Città di **Arad** (S);

Punto di Arrivo: Città di **Neamt** (G);

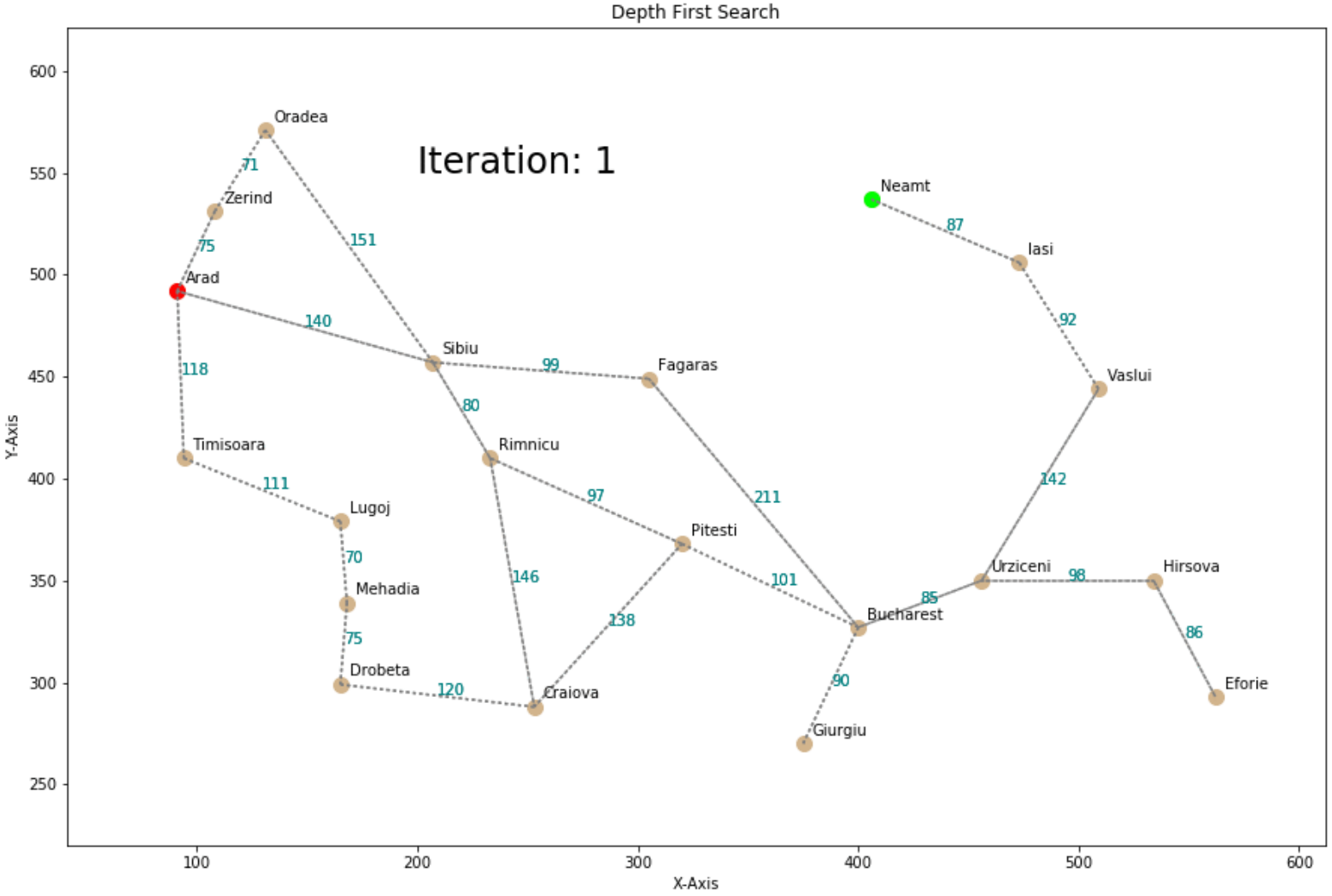
Algoritmo utilizzato: **BFS**.

Iterazioni: **20**.

Obiettivo Raggiunto: **SI**.

Fonte GIF: http://teleported.in/posts/ai-search-algorithms/

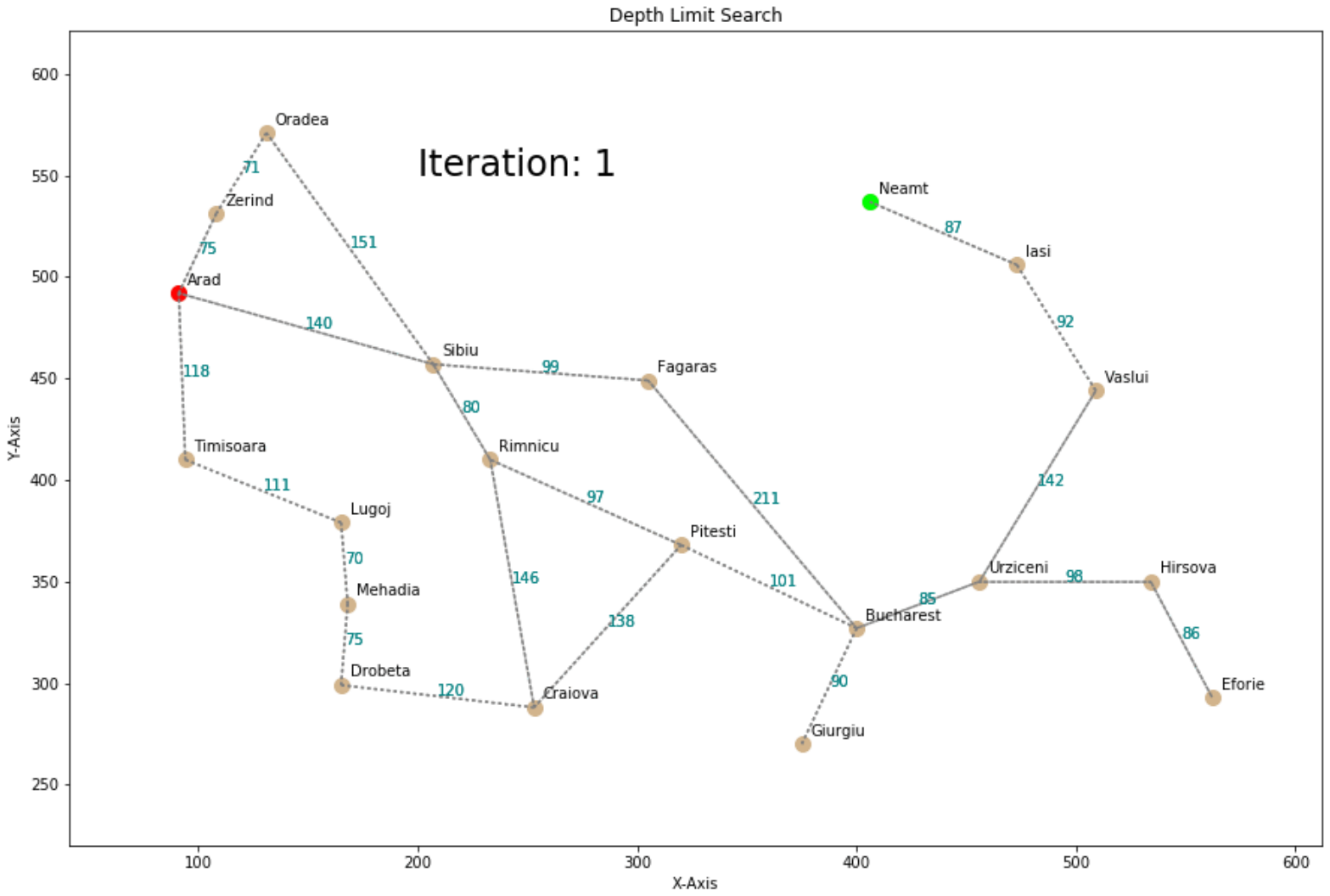
Confronto con Map Navigation 2/3



- Punto di Partenza: Città di **Arad** (S);
- Punto di Arrivo: Città di **Neamt** (G);
- Algoritmo utilizzato: **DFS**.
- Iterazioni: **11**.
- Obiettivo Raggiunto: **SI**.

Fonte GIF: http://teleported.in/posts/ai-search-algorithms/

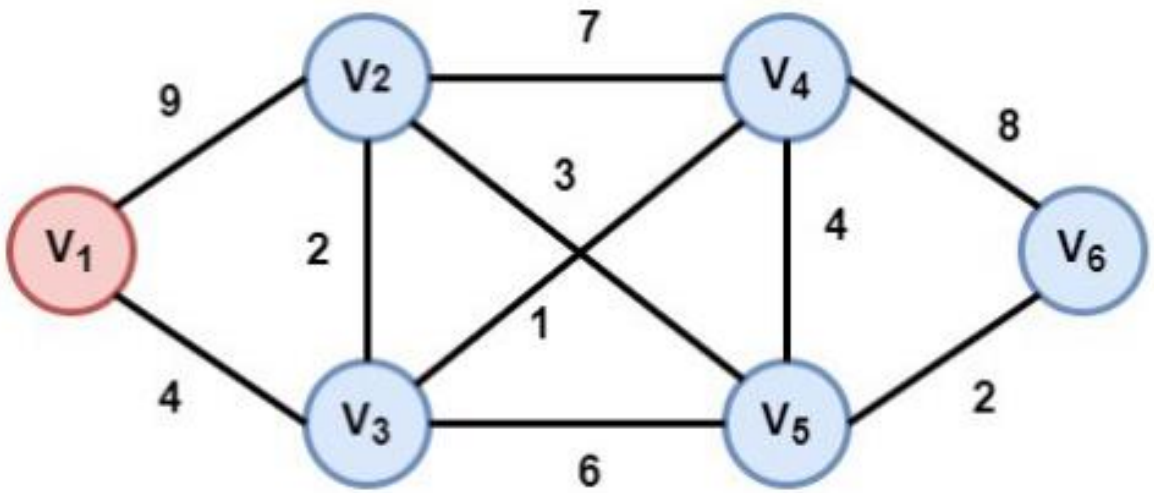
Confronto con Map Navigation 3/3



- Punto di Partenza: Città di **Arad** (S);
- Punto di Arrivo: Città di **Neamt** (G);
- Algoritmo utilizzato: **DLS**.
- Limite l: **4**.
- Iterazioni: **10**.
- Obiettivo Raggiunto: **NO**.

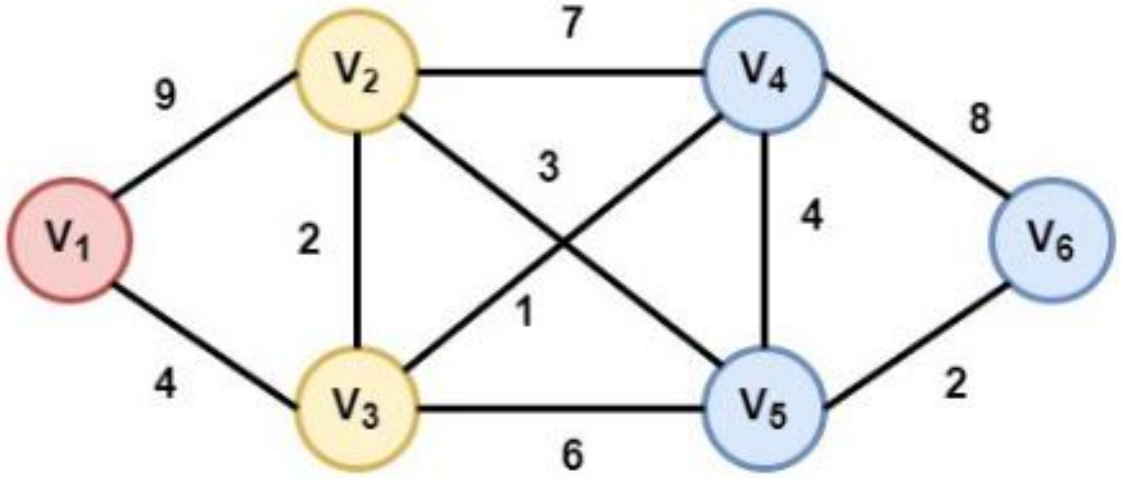
Fonte GIF: http://teleported.in/posts/ai-search-algorithms/

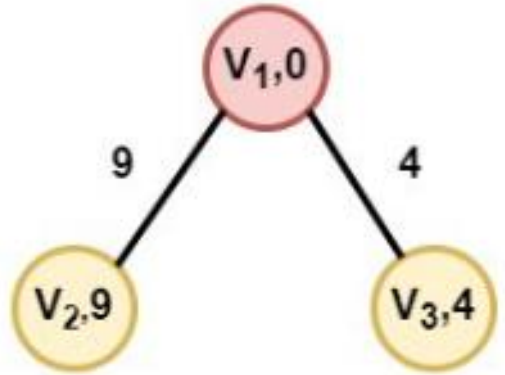
Esempio UCS: Passaggio 1 - Inizializzazione



Pace	Opened	Closed
0	$\{(V_1,0)\}$	{-}

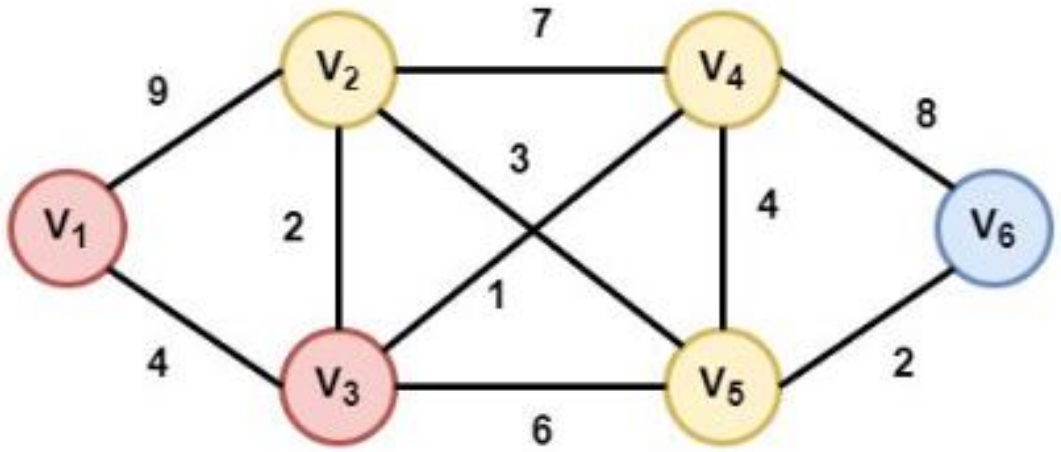
Esempio UCS - Viene selezionato il nodo V1



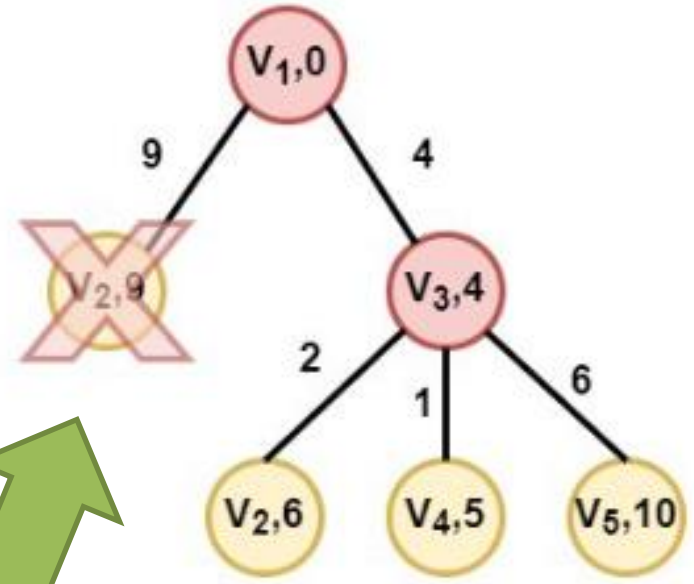


Pace	Opened	Closed
0	$\{(V_1,0)\}$	{-}
1	$\{(V_2,9),(V_3,4)\}$	$\{(V_1,0)\}$

Esempio UCS - Viene selezionato il nodo V3

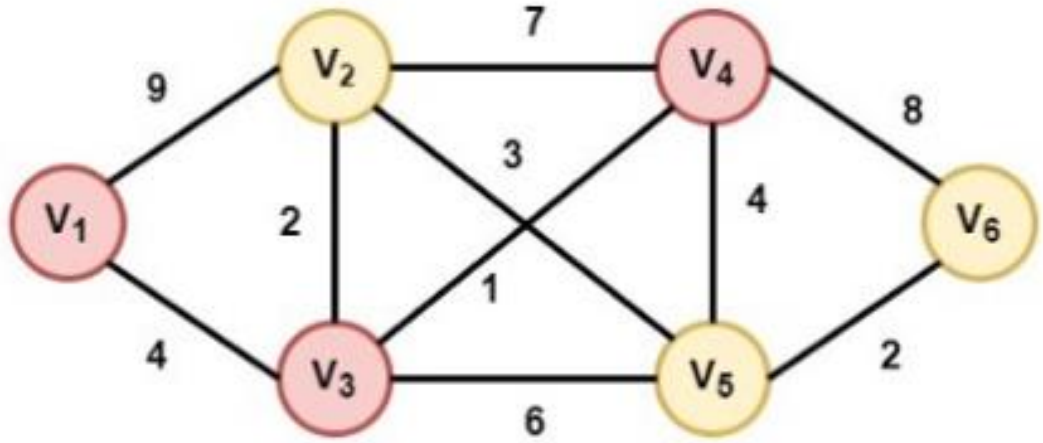


Pace	Opened	Closed
0	{(V_1,0)}	{-}
1	{(V_2,9),(V_3,4)}	{(V_1,0)}
2	{(V_2,6),(V_4,5),(V_5,10)}	{(V_1,0),(V_3,4)}

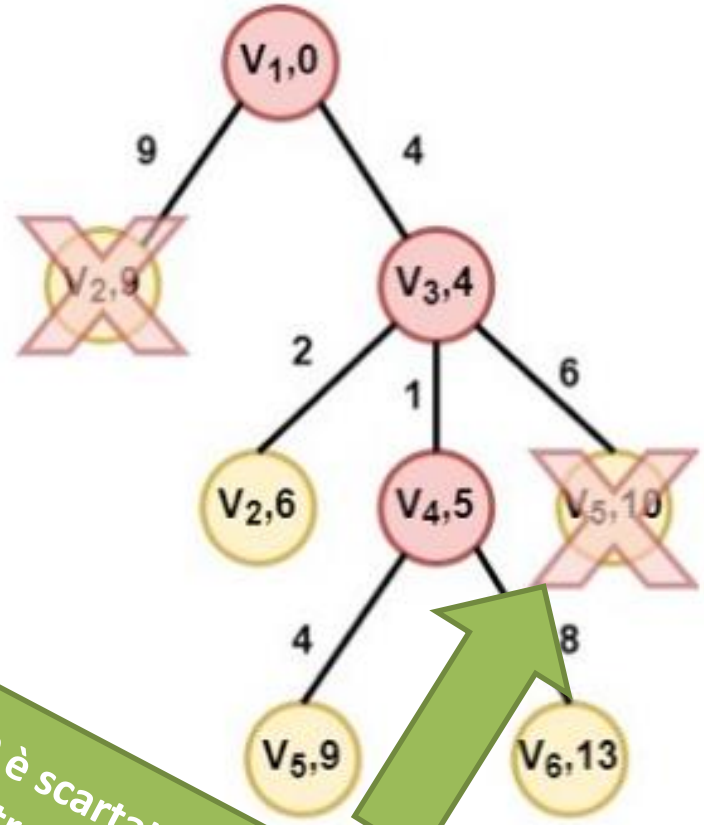


V2,9 è scartato perché è stato trovato un nuovo percorso per raggiungere V2 con costo minore (6)

Esempio UCS - Viene selezionato il nodo V4

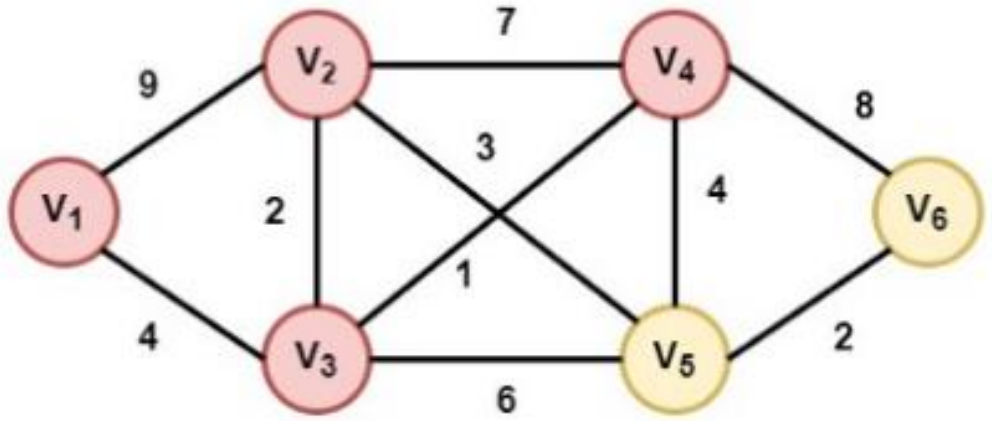


Pace	Opened	Closed
0	{(V_1,0)}	{-}
1	{(V_2,9),(V_3,4)}	{(V_1,0)}
2	{(V_2,6),(V_4,5),(V_5,10)}	{(V_1,0),(V_3,4)}
3	{(V_2,6),(V_6,13),(V_5,9)}	{(V_1,0),(V_3,4),(V_4,5)}

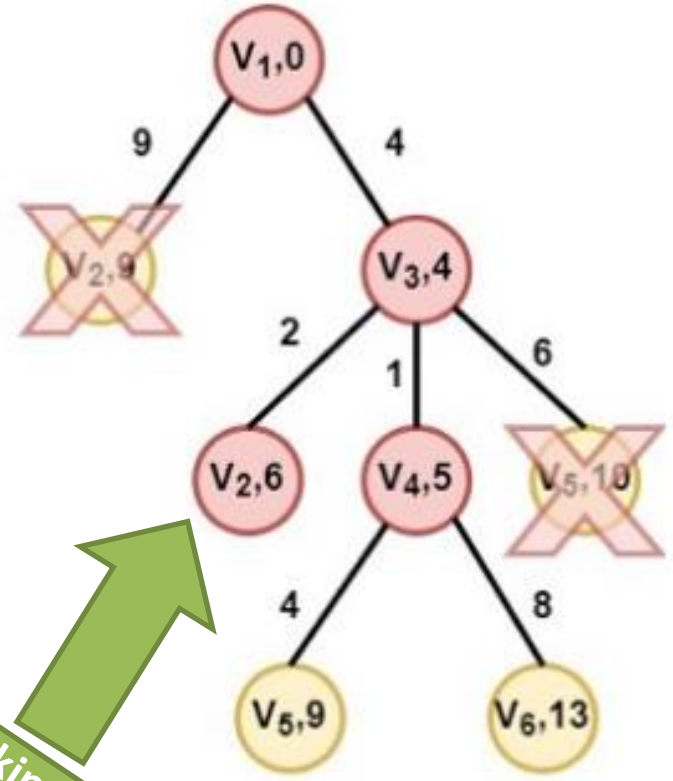


V5,10 è scartato perché è stato trovato un nuovo percorso per raggiungere V5 con costo minore (9)

Esempio UCS - Viene selezionato il nodo V2

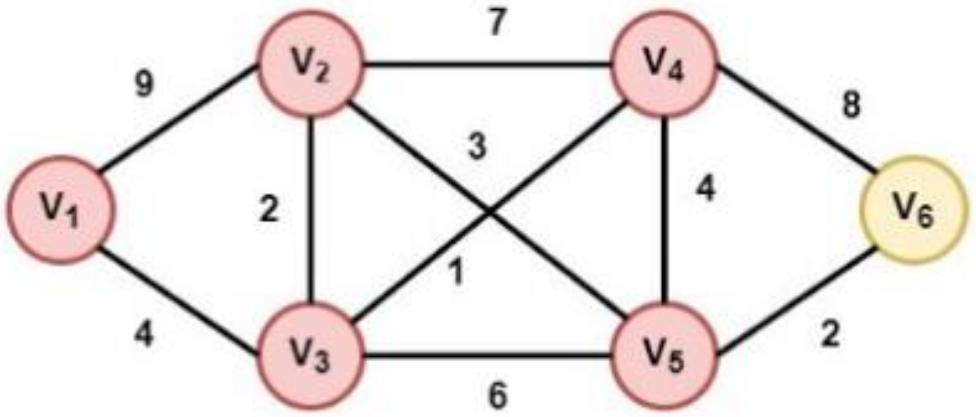


Pace	Opened	Closed
0	{(V_1,0)}	{-}
1	{(V_2,9),(V_3,4)}	{(V_1,0)}
2	{(V_2,6),(V_4,5),(V_5,10)}	{(V_1,0),(V_3,4)}
3	{(V_2,6),(V_6,13),(V_5,9)}	{(V_1,0),(V_3,4),(V_4,5)}
4	{(V_6,13),(V_5,9)}	{(V_1,0),(V_3,4),(V_4,5),(V_2,6)}

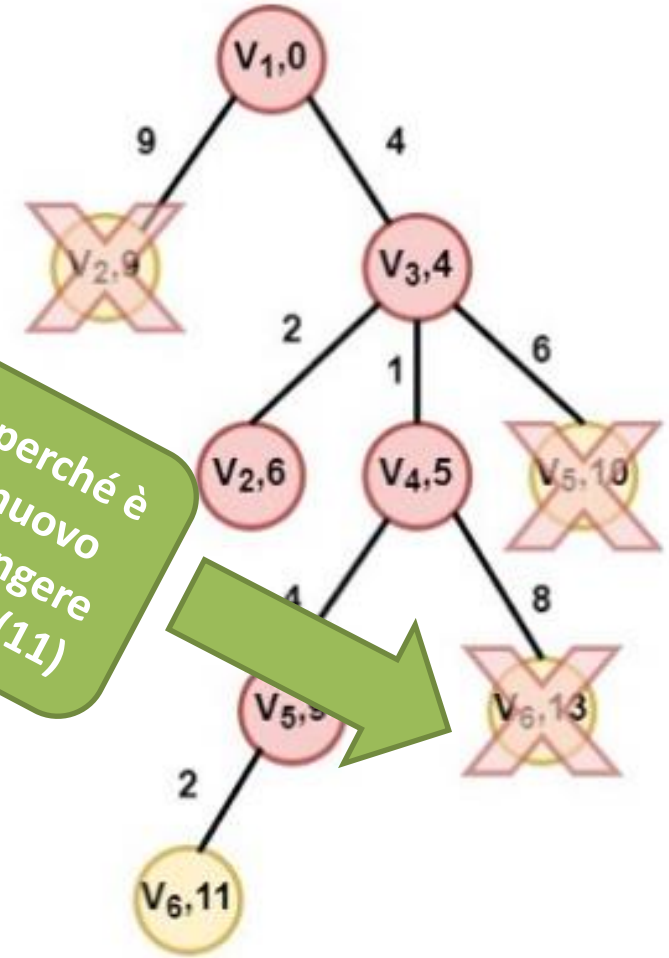


Backtracking. V3 e V4 sono già inseriti nell'elenco chiuso, e V5 ha sempre costo 9

Esempio UCS - Viene selezionato il nodo V5

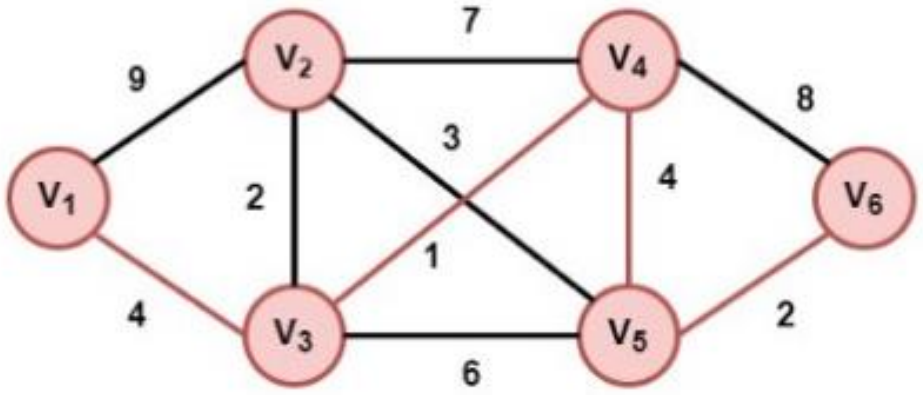


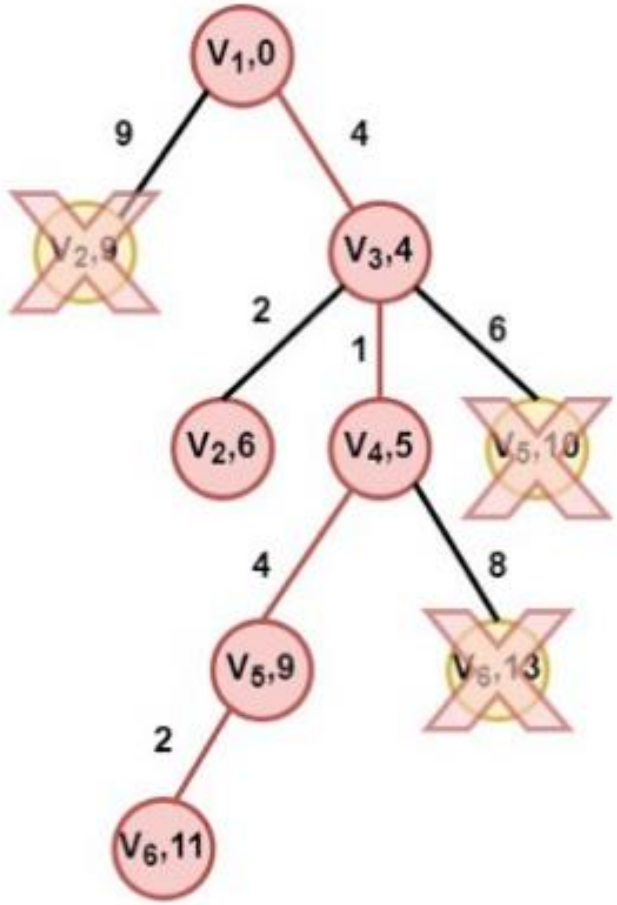
Pace	Opened	Closed
0	{(V_1,0)}	{-}
1	{(V_2,9),(V_3,4)}	{(V_1,0)}
2	{(V_2,6),(V_4,5),(V_5,10)}	{(V_1,0),(V_3,4)}
3	{(V_2,6),(V_6,13),(V_5,9)}	{(V_1,0),(V_3,4),(V_4,5)}
4	{(V_6,13),(V_5,9)}	{(V_1,0),(V_3,4),(V_4,5),(V_2,6)}
5	{(V_6,11)}	{(V_1,0),(V_3,4),(V_4,5),(V_2,6),(V_5,9)}



V6,13 è scartato perché è stato trovato un nuovo percorso per raggiungere V6 con costo minore (11)

Esempio UCS - Viene selezionato il nodo V6

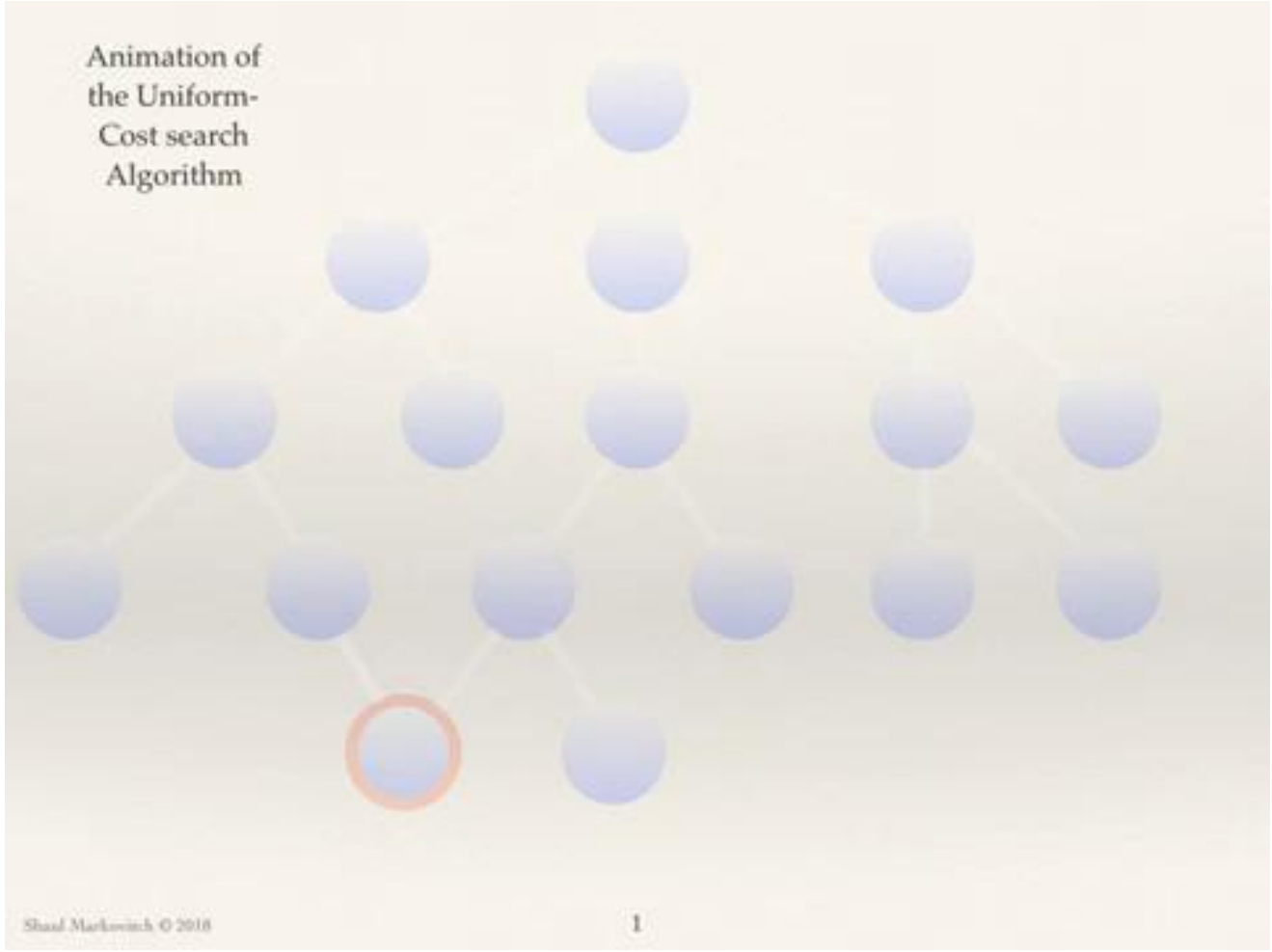




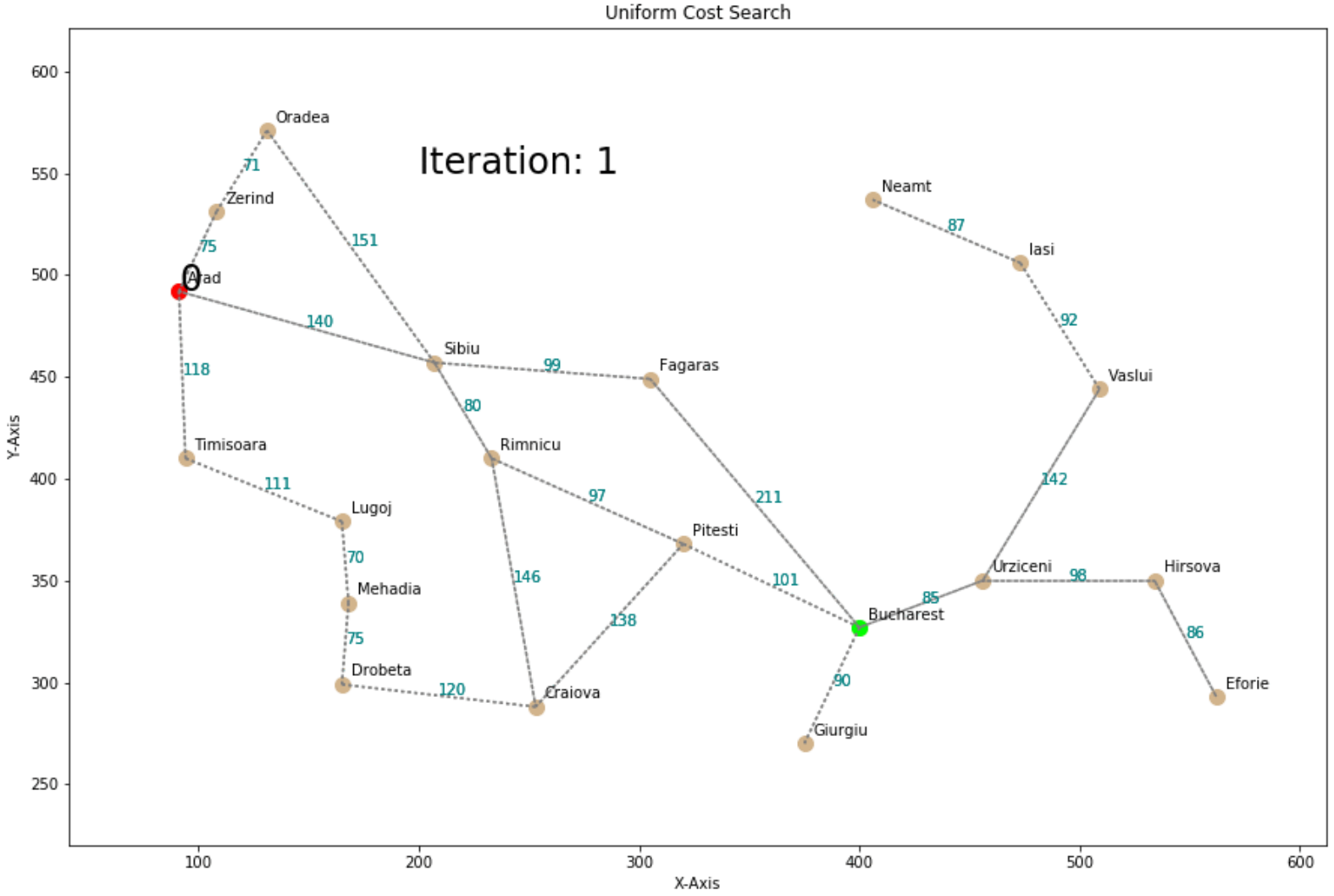
Pace	Opened	Closed
0	{(V_1,0)}	{-}
1	{(V_2,9),(V_3,4)}	{(V_1,0)}
2	{(V_2,6),(V_4,5),(V_5,10)}	{(V_1,0),(V_3,4)}
3	{(V_2,6),(V_6,13),(V_5,9)}	{(V_1,0),(V_3,4),(V_4,5)}
4	{(V_6,13),(V_5,9)}	{(V_1,0),(V_3,4),(V_4,5),(V_2,6)}
5	{(V_6,11)}	{(V_1,0),(V_3,4),(V_4,5),(V_2,6),(V_5,9)}
6	{-}	{(V_1,0),(V_3,4),(V_4,5),(V_2,6),(V_5,9)}

Animazione UCS

https://www.youtube.com/watch?v=XyoucHYKYSE&t=1s



Map Navigation con UCS

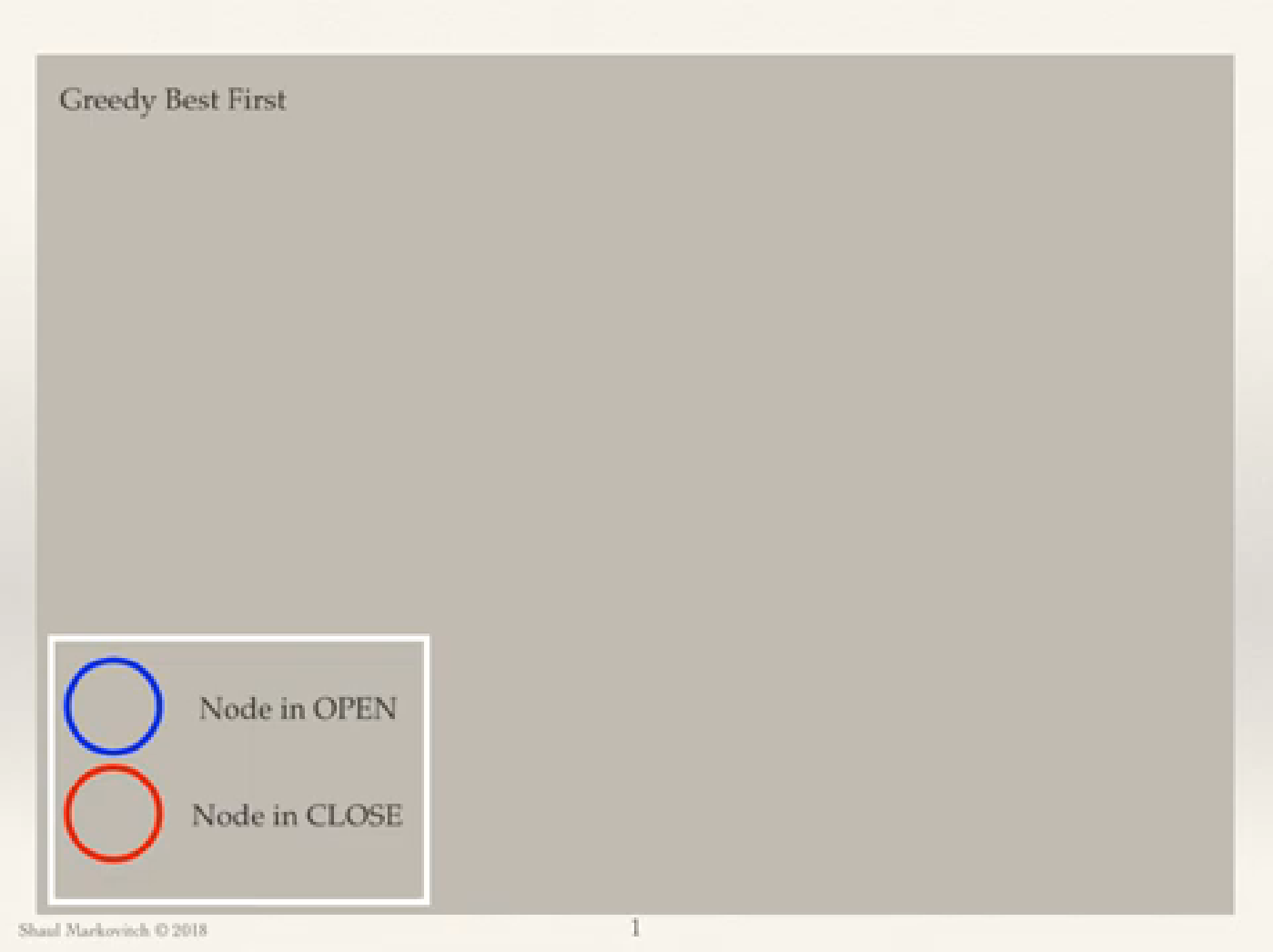


- Punto di Partenza: Città di **Arad** (S);
- Punto di Arrivo: Città di **Bucarest** (G);
- Algoritmo utilizzato: **UCS**.
- Iterazioni: **13**.
- Obiettivo Raggiunto: **SI**.

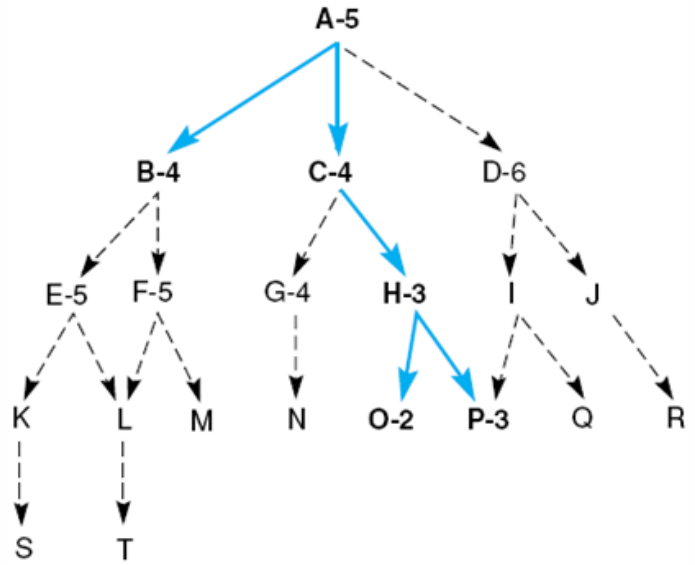
Fonte GIF: http://teleported.in/posts/ai-search-algorithms/

Best First Search: Esempio

https://www.youtube.com/watch?v=A8pmud1Uh0Q

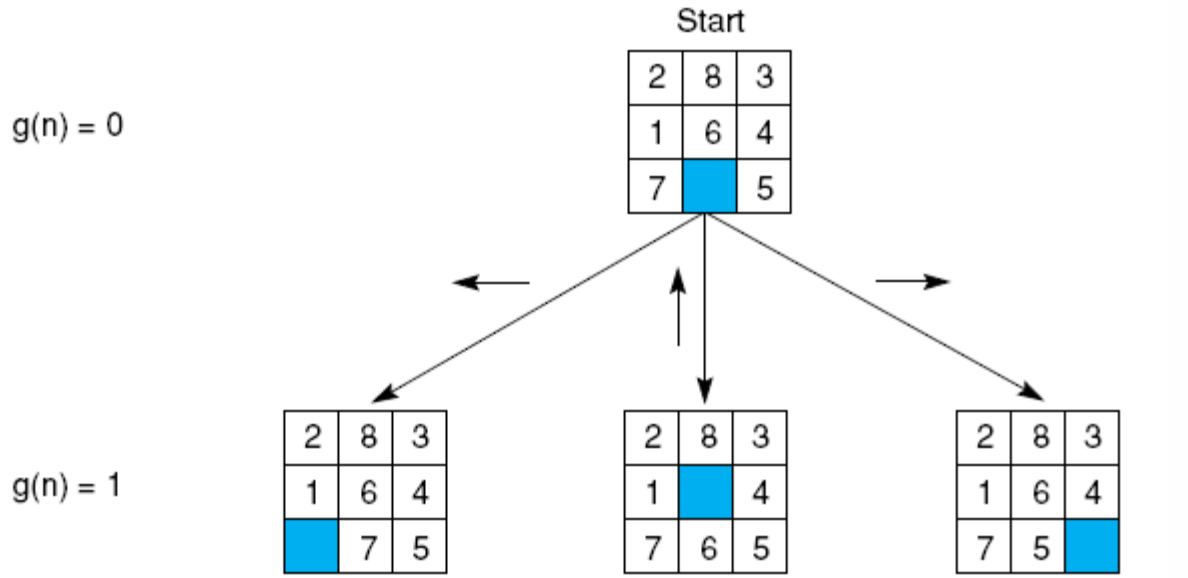


Best First Search: Esempio



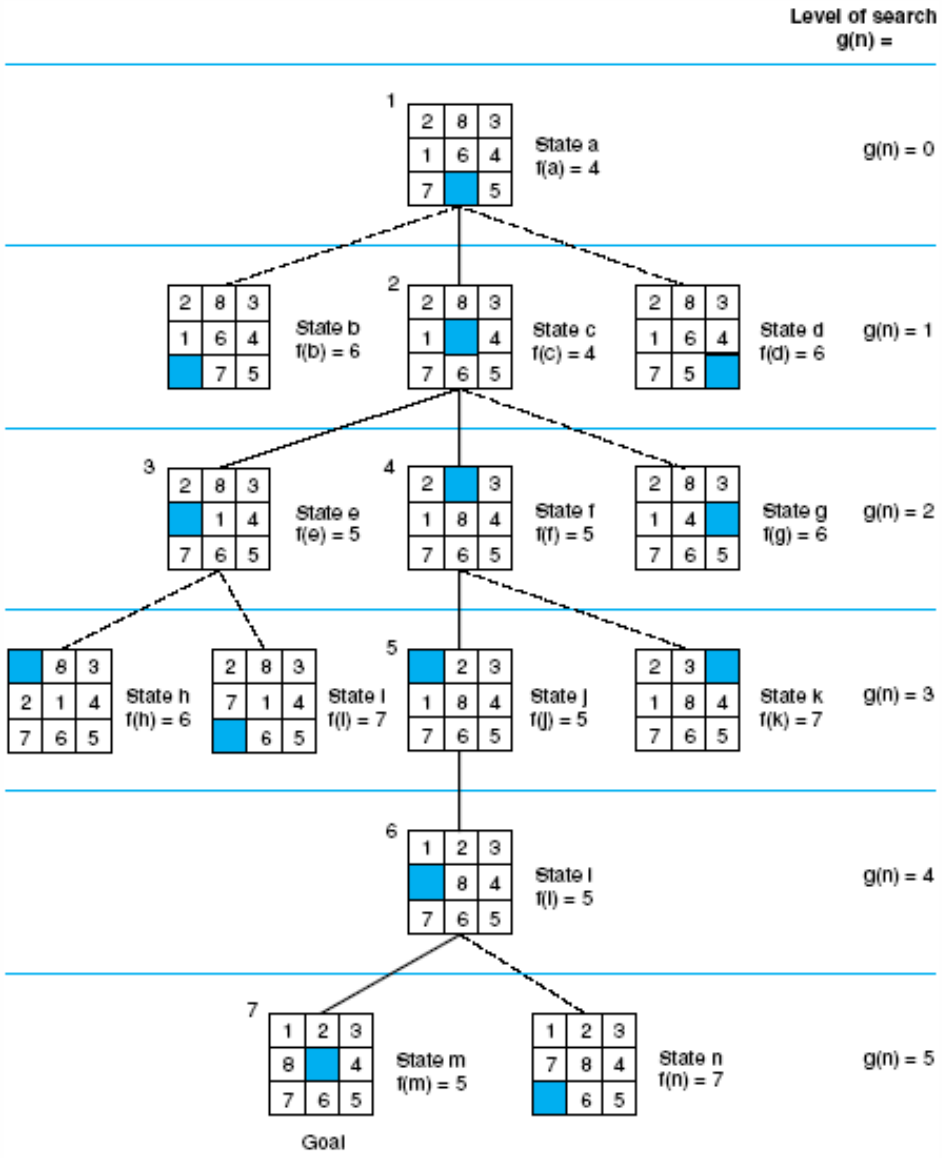
1. open = [A5]; closed = [ ]
2. evaluate A5; open = [B4,C4,D6]; closed = [A5]
3. evaluate B4; open = [C4,E5,F5,D6]; closed = [B4,A5]
4. evaluate C4; open = [H3,G4,E5,F5,D6]; closed = [C4,B4,A5]
5. evaluate H3; open = [O2,P3,G4,E5,F5,D6]; closed = [H3,C4,B4,A5]
6. evaluate O2; open = [P3,G4,E5,F5,D6]; closed = [O2,H3,C4,B4,A5]
7. evaluate P3; the solution is found!

A*: Esempio 8-Puzzle 1/2

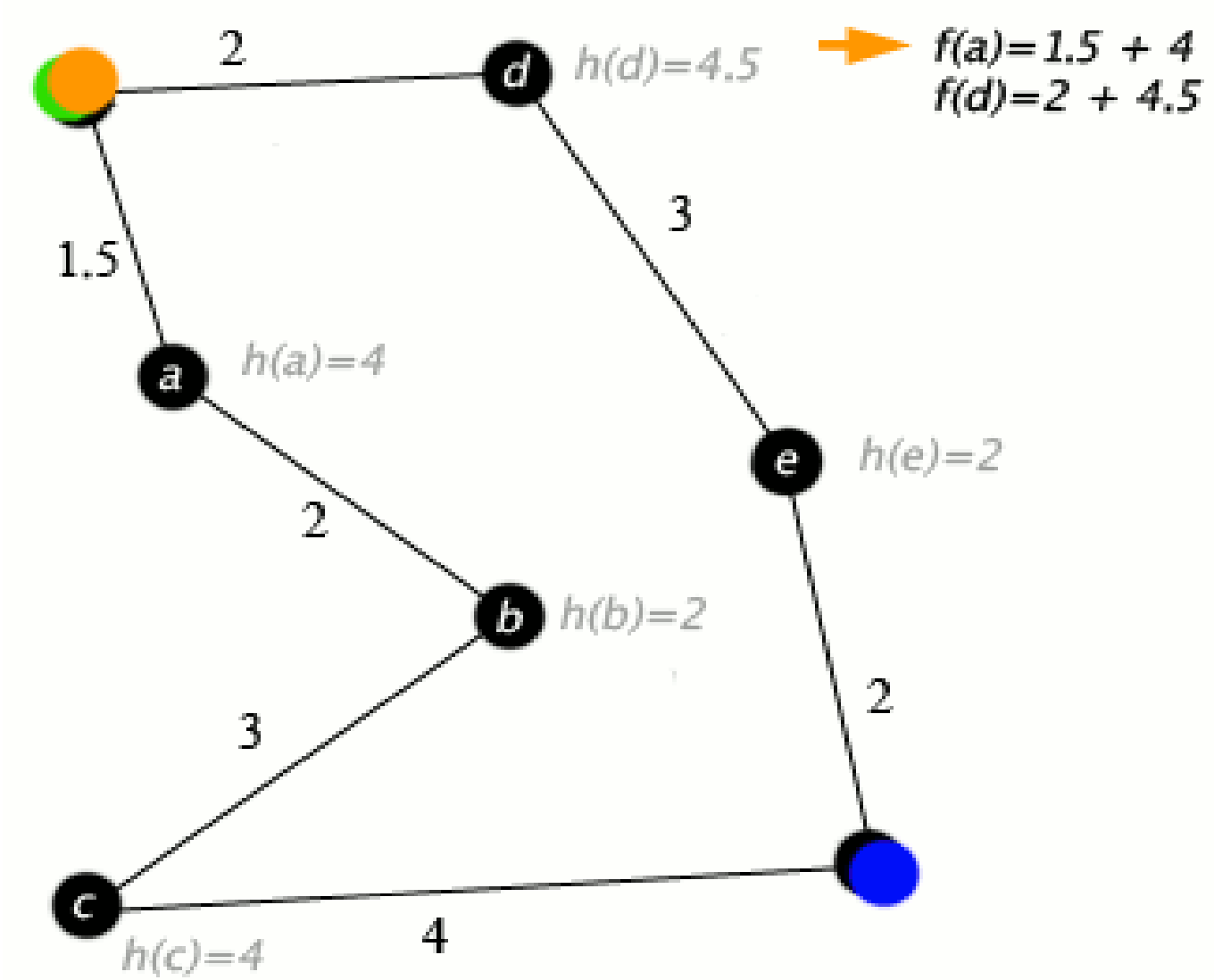


A*: Esempio 8-Puzzle 2/2

1. **open = [a4];**
   closed = [ ]
2. **open = [c4, b6, d6];**
   closed = [a4]
3. **open = [e5, f5, b6, d6, g6];**
   closed = [a4, c4]
4. **open = [f5, h6, b6, d6, g6, l7];**
   closed = [a4, c4, e5]
5. **open = [j5, h6, b6, d6, g6, k7, l7];**
   closed = [a4, c4, e5, f5]
6. **open = [l5, h6, b6, d6, g6, k7, l7];**
   closed = [a4, c4, e5, f5, j5]
7. **open = [m5, h6, b6, d6, g6, n7, k7, l7];**
   closed = [a4, c4, e5, f5, j5, l5]
8. **success, m = goal!**



A*: Esempio



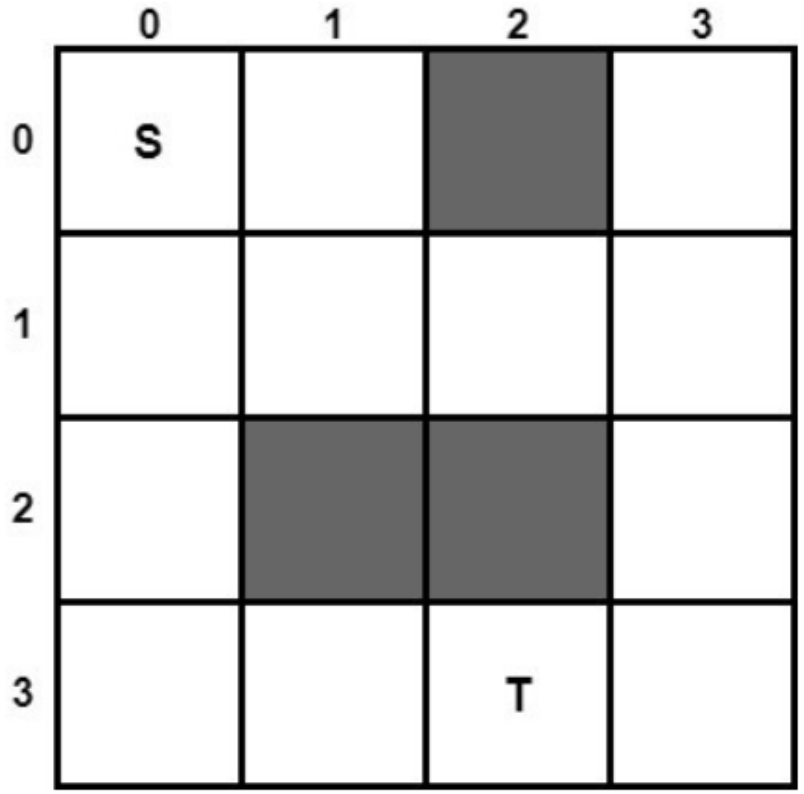
Fonte GIF:
https://en.m.wikipedia.org/wiki/A*_search_algorithm#/media/File%3AAstarExampleEn.gif

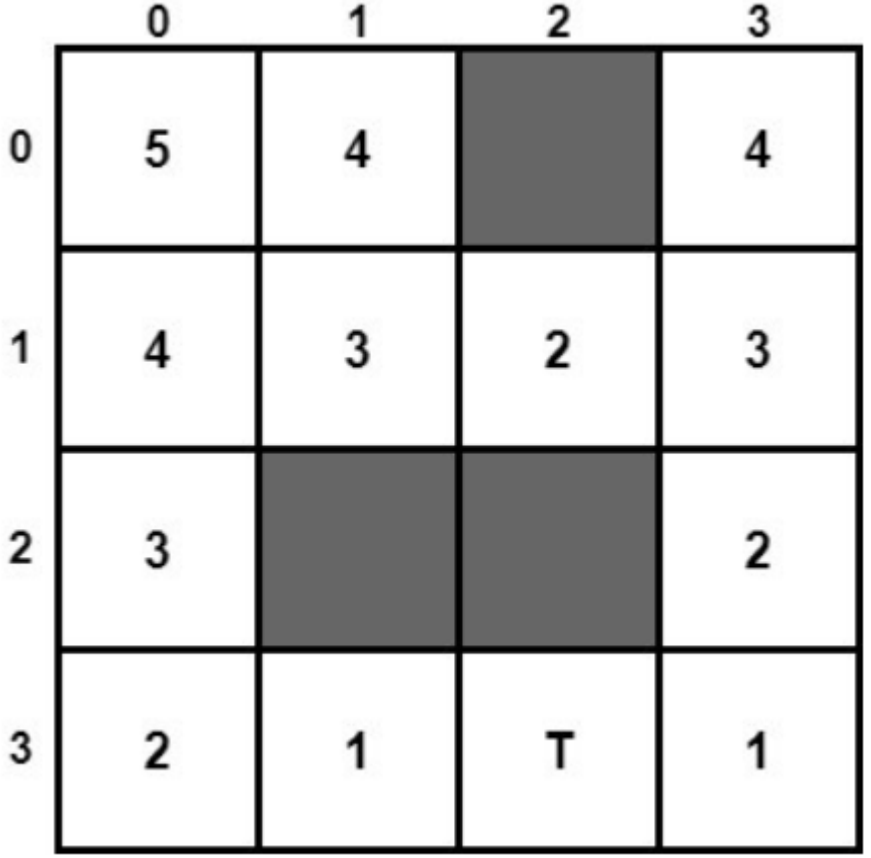
A*: Esempio

https://www.youtube.com/watch?v=zycfu79Y8Fw

A* Esempio Robot Navigation

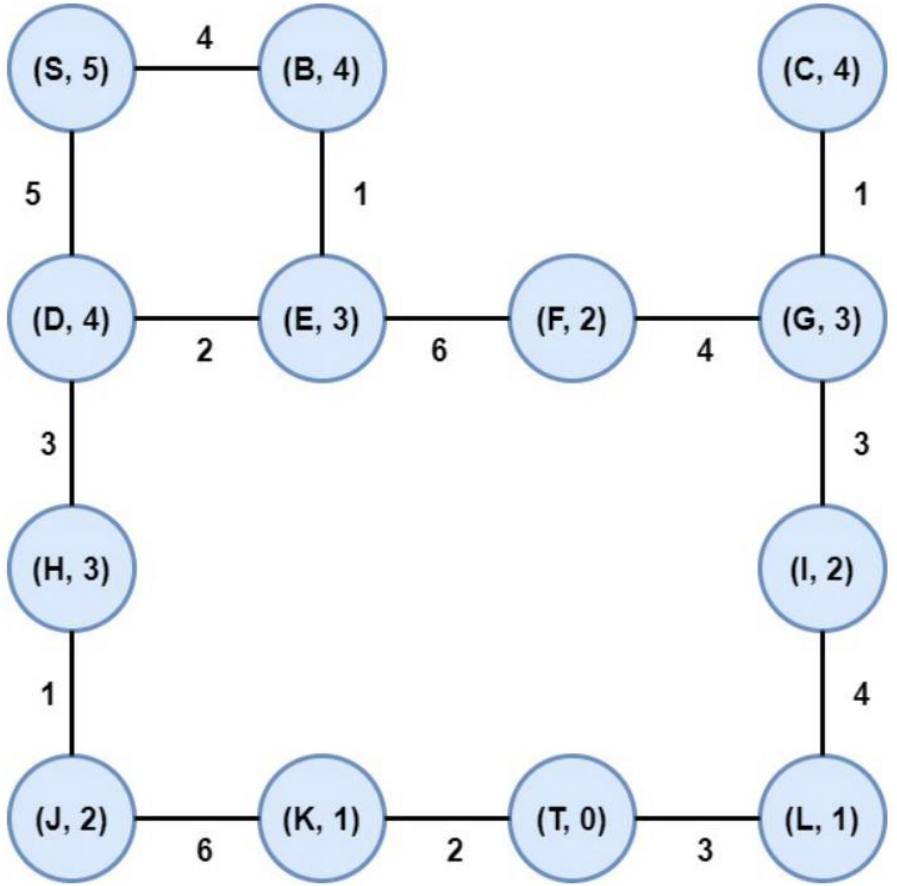
- Supponiamo di avere un robot e di volere che il robot navighi dal punto S in posizione (0, 0) al punto T in posizione (3, 2).
- I quadrati grigi sono ostacoli che il robot non può superare.
- Come funzione euristica, utilizzeremo la distanza di Manhattan (nel calcolo non si tiene conto degli ostacoli).



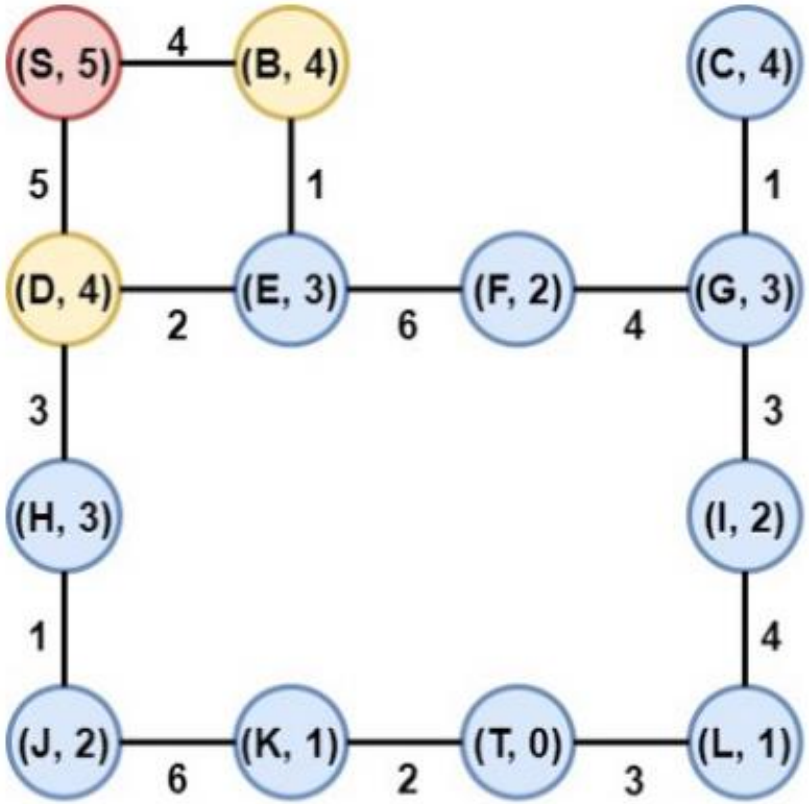


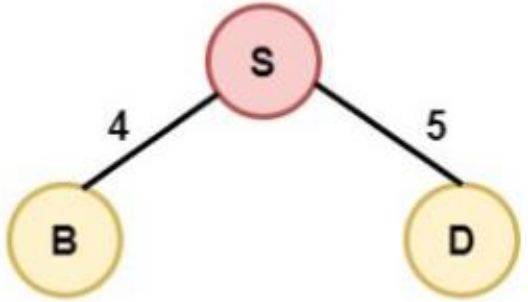
A* Esempio Robot Navigation

	0	1	2	3
0	5	4		4
1	4	3	2	3
2	3			2
3	2	1	T	1



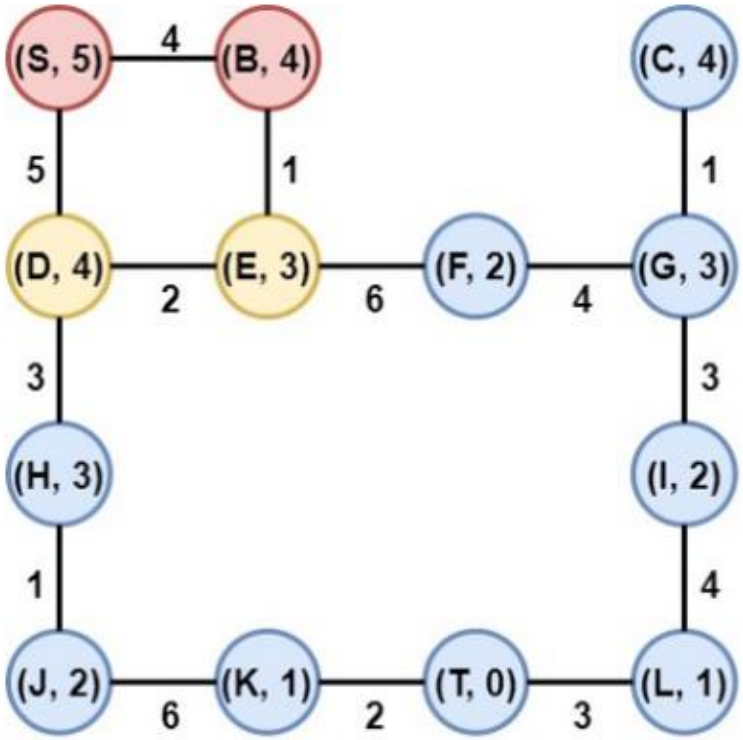
A* Robot Navigation: Inizializzazione

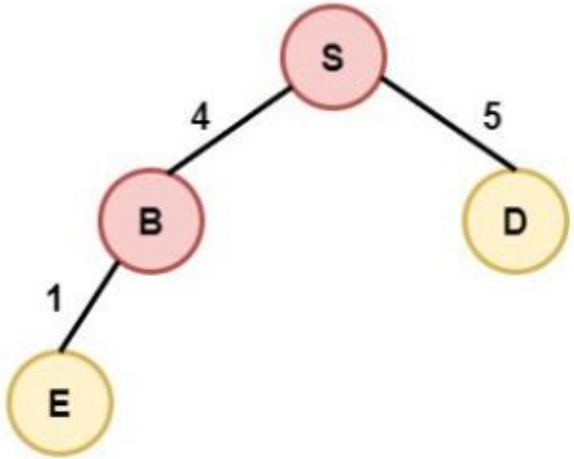




Pace	Opened	Closed
0	{(S,5)}	{-}
1	{(B,8),(D,9)}	{(S,5)}

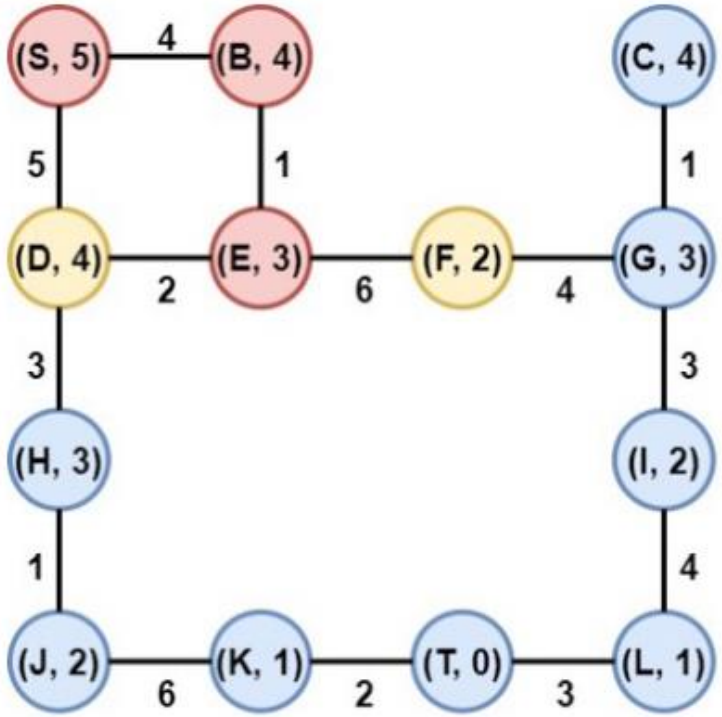
A* Robot Navigation: Nodo B Selezionato

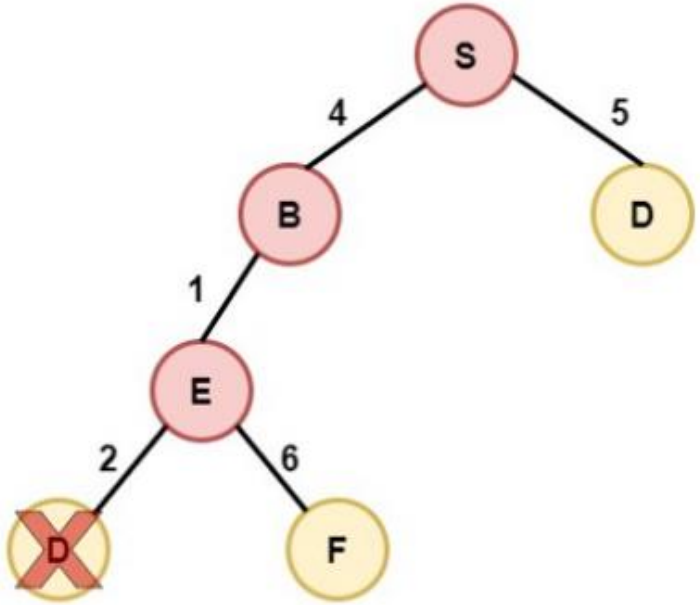




Pace	Opened	Closed
0	{(S,5)}	{-}
1	{(B,8),(D,9)}	{(S,5)}
2	{(E,8),(D,9)}	{(S,5),(B,8)}

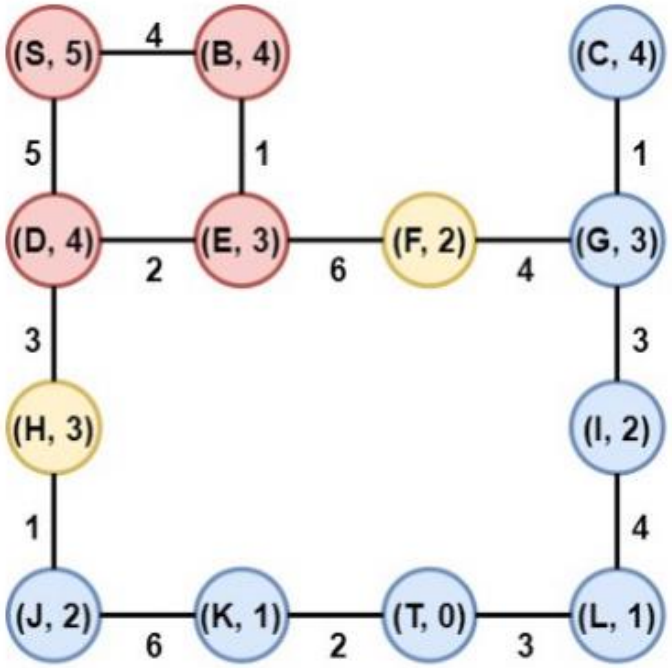
A* Robot Navigation: Nodo E Selezionato

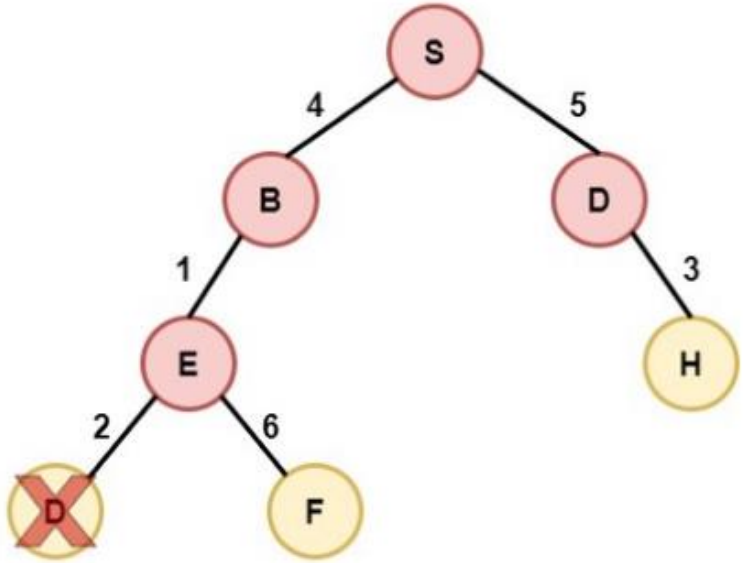




Pace	Opened	Closed
0	{(S,5)}	{-}
1	{(B,8),(D,9)}	{(S,5)}
2	{(E,8),(D,9)}	{(S,5),(B,8)}
3	{(F,13),(D,9)}	{(S,5),(B,8),(E,8)}

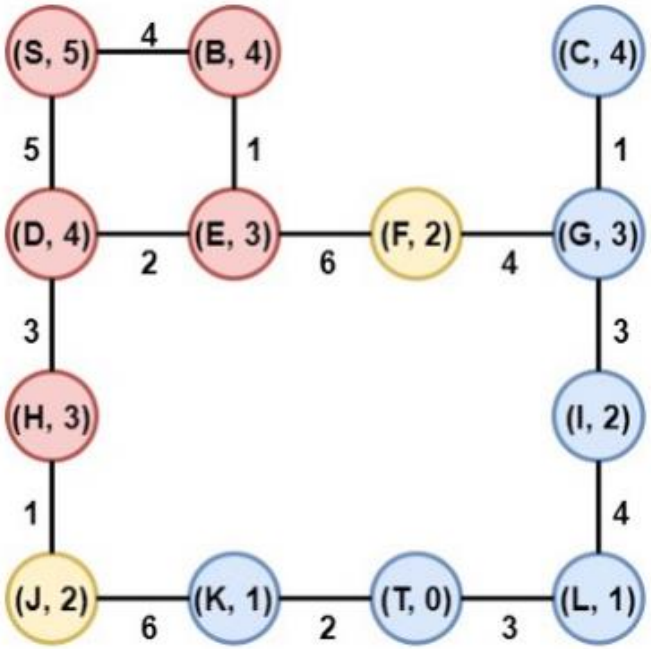
A* Robot Navigation: Nodo D Selezionato

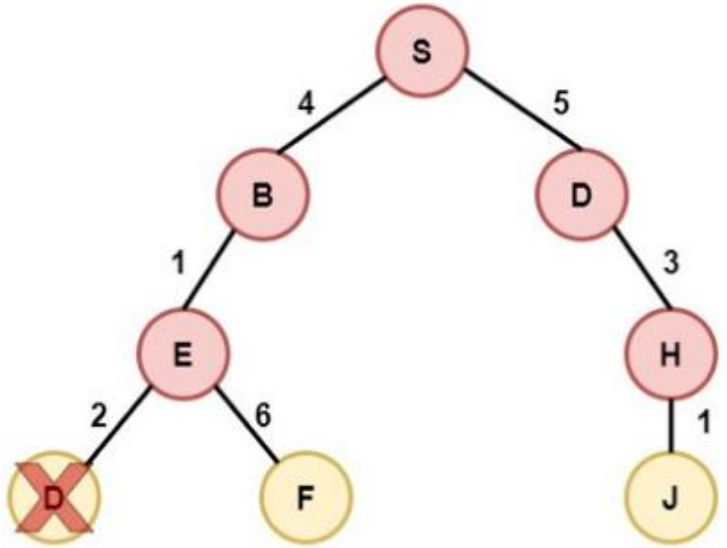




Pace	Opened	Closed
0	{(S,5)}	{-}
1	{(B,8),(D,9)}	{(S,5)}
2	{(E,8),(D,9)}	{(S,5),(B,8)}
3	{(F,13),(D,9)}	{(S,5),(B,8),(E,8)}
4	{(F,13),(H,11)}	{(S,5),(B,8),(E,8),(D,9)}

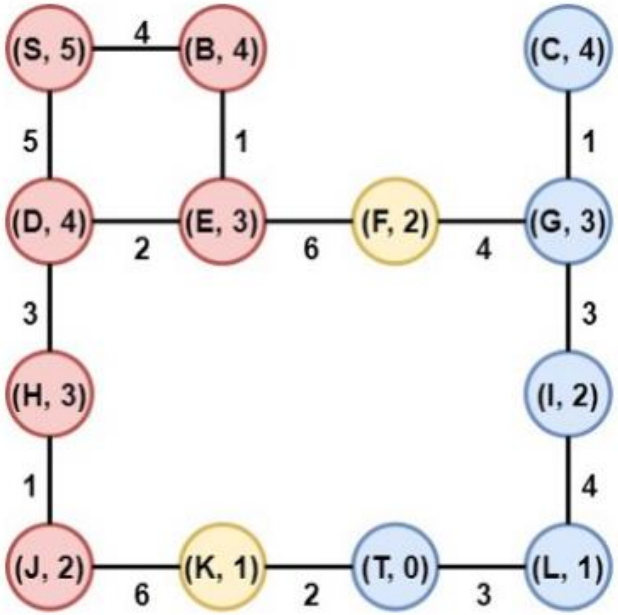
A* Robot Navigation: Nodo H Selezionato

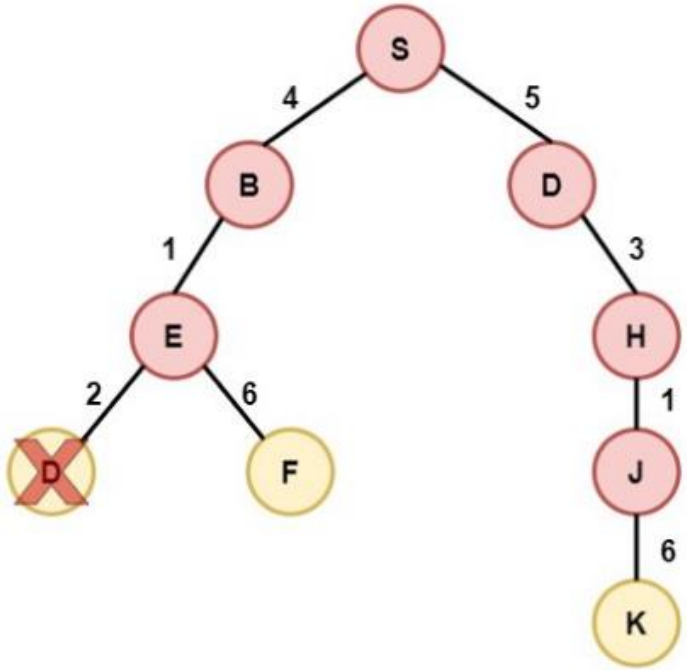




Pace	Opened	Closed
0	{(S,5)}	{-}
1	{(B,8),(D,9)}	{(S,5)}
2	{(E,8),(D,9)}	{(S,5),(B,8)}
3	{(F,13),(D,9)}	{(S,5),(B,8),(E,8)}
4	{(F,13),(H,11)}	{(S,5),(B,8),(E,8),(D,9)}
5	{(F,13),(J,11)}	{(S,5),(B,8),(E,8),(D,9),(H,11)}

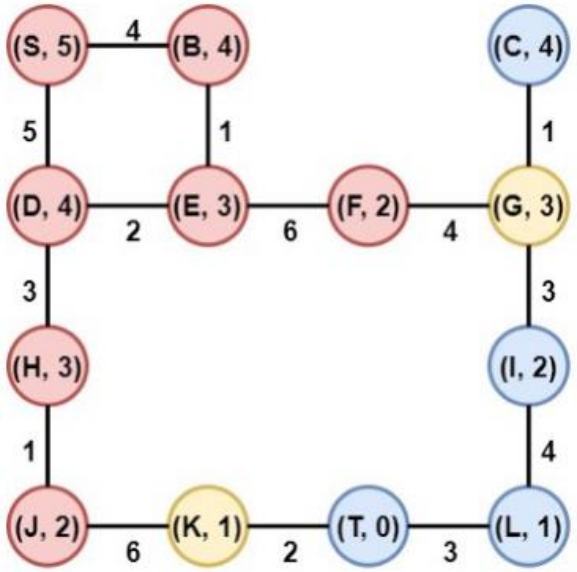
A* Robot Navigation: Nodo J Selezionato

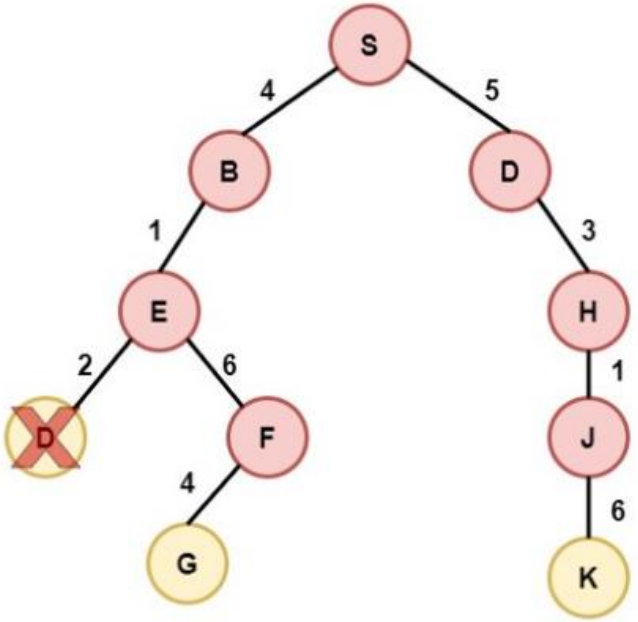




Pace	Opened	Closed
0	{(S,5)}	{-}
1	{(B,8),(D,9)}	{(S,5)}
2	{(E,8),(D,9)}	{(S,5),(B,8)}
3	{(F,13),(D,9)}	{(S,5),(B,8),(E,8)}
4	{(F,13),(H,11)}	{(S,5),(B,8),(E,8),(D,9)}
5	{(F,13),(J,11)}	{(S,5),(B,8),(E,8),(D,9),(H,11)}
6	{(F,13),(K,16)}	{(S,5),(B,8),(E,8),(D,9),(H,11),(J,11)}

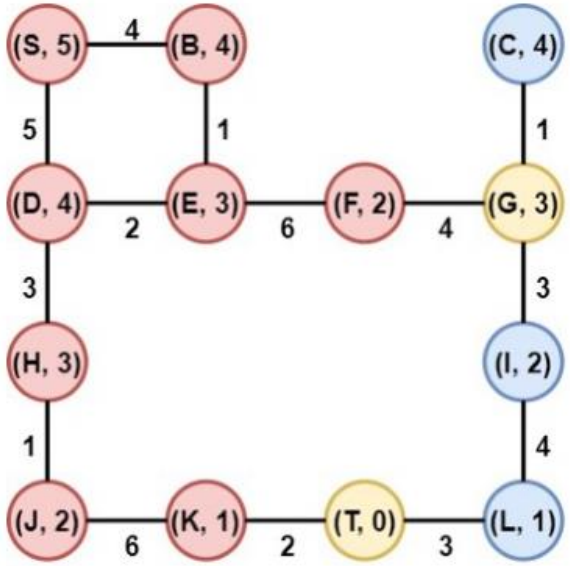
A* Robot Navigation: Nodo F Selezionato

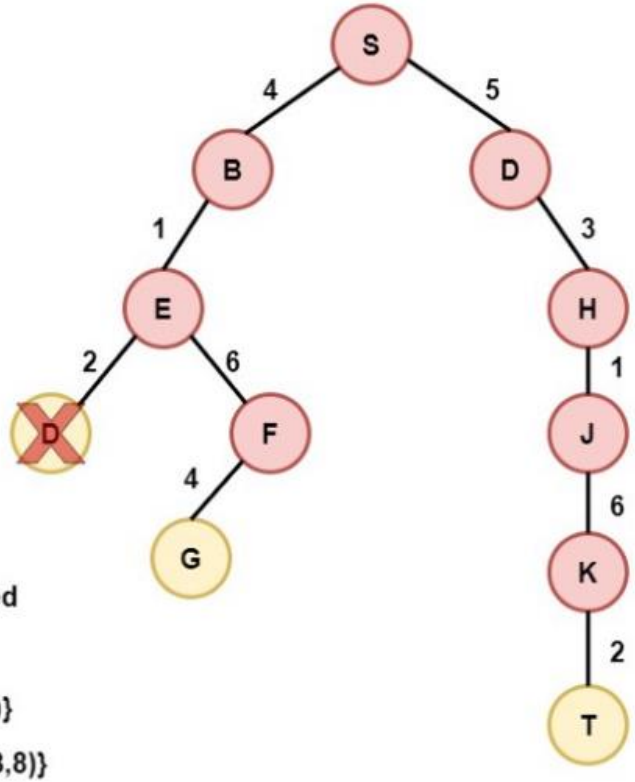




Pace	Opened	Closed
0	{(S,5)}	{-}
1	{(B,8),(D,9)}	{(S,5)}
2	{(E,8),(D,9)}	{(S,5),(B,8)}
3	{(F,13),(D,9)}	{(S,5),(B,8),(E,8)}
4	{(F,13),(H,11)}	{(S,5),(B,8),(E,8),(D,9)}
5	{(F,13),(J,11)}	{(S,5),(B,8),(E,8),(D,9),(H,11)}
6	{(F,13),(K,16)}	{(S,5),(B,8),(E,8),(D,9),(H,11),(J,11)}
7	{(K,16),(G,18)}	{(S,5),(B,8),(E,8),(D,9),(H,11),(J,11),(F,13)}

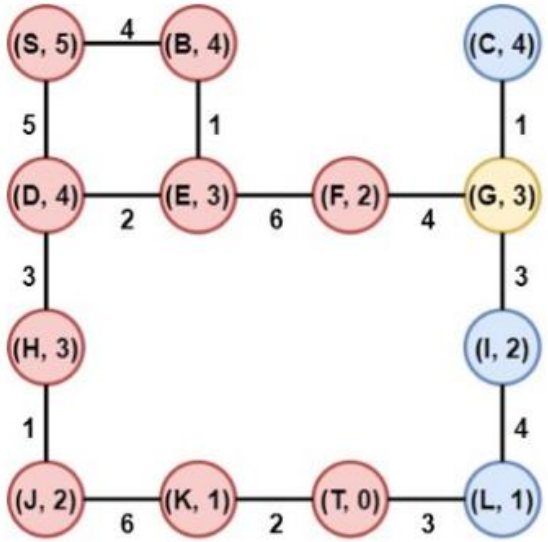
A* Robot Navigation: Nodo K Selezionato

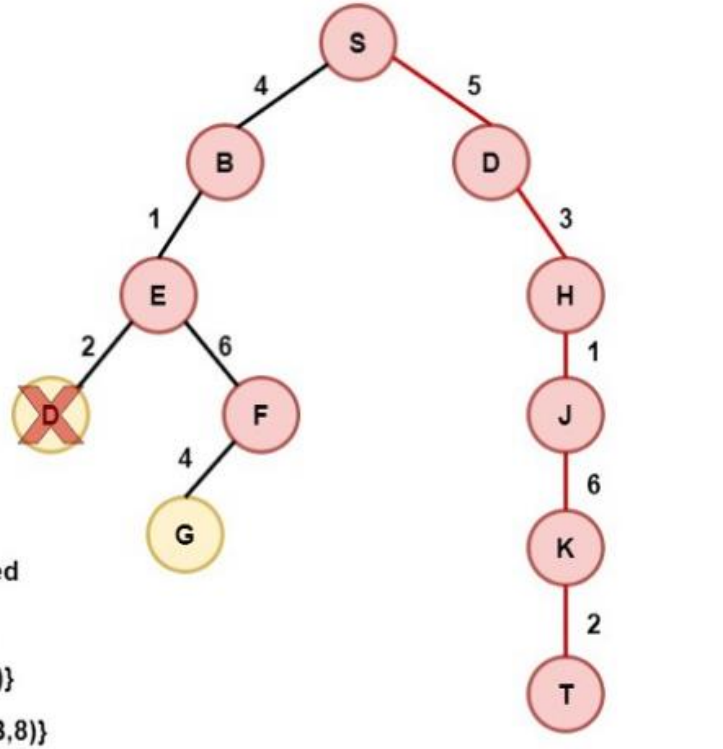




Pace	Opened	Closed
0	{(S,5)}	{-}
1	{(B,8),(D,9)}	{(S,5)}
2	{(E,8),(D,9)}	{(S,5),(B,8)}
3	{(F,13),(D,9)}	{(S,5),(B,8),(E,8)}
4	{(F,13),(H,11)}	{(S,5),(B,8),(E,8),(D,9)}
5	{(F,13),(J,11)}	{(S,5),(B,8),(E,8),(D,9),(H,11)}
6	{(F,13),(K,16)}	{(S,5),(B,8),(E,8),(D,9),(H,11),(J,11)}
7	{(K,16),(G,18)}	{(S,5),(B,8),(E,8),(D,9),(H,11),(J,11),(F,13)}
8	{(T,17),(G,18)}	{(S,5),(B,8),(E,8),(D,9),(H,11),(J,11),(F,13),(K,16)}

A* Robot Navigation: Nodo T Selezionato





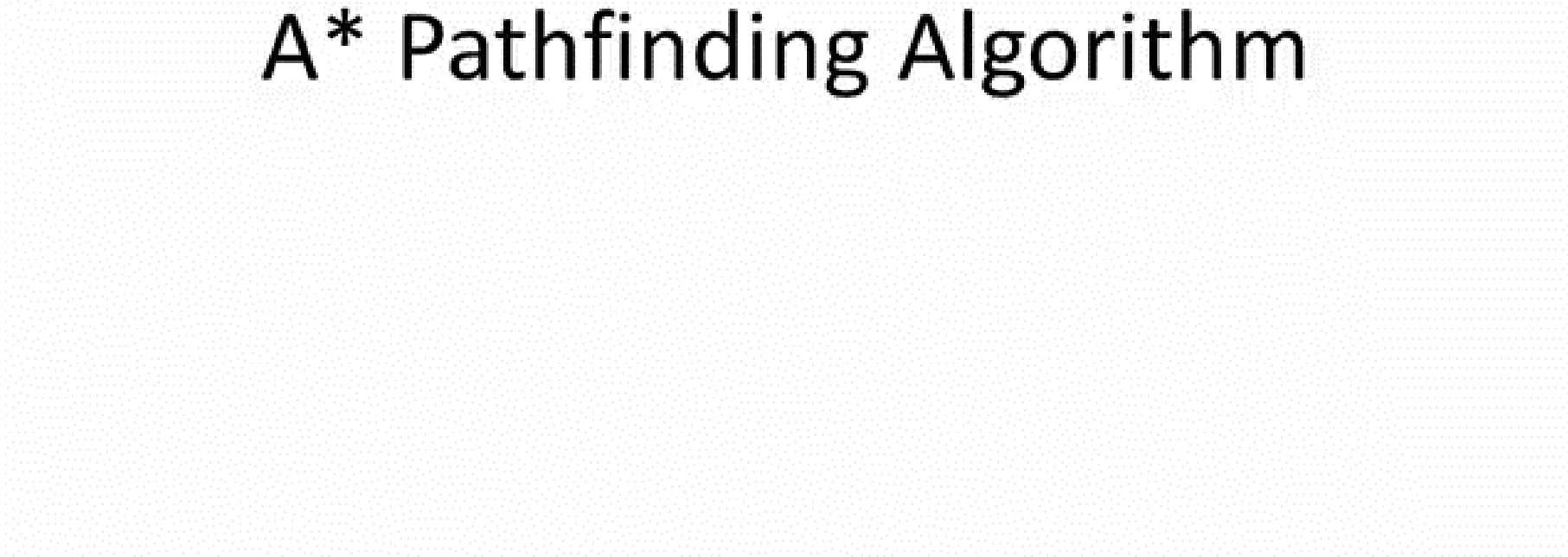
Pace	Opened	Closed
0	{(S,5)}	{-}
1	{(B,8),(D,9)}	{(S,5)}
2	{(E,8),(D,9)}	{(S,5),(B,8)}
3	{(F,13),(D,9)}	{(S,5),(B,8),(E,8)}
4	{(F,13),(H,11)}	{(S,5),(B,8),(E,8),(D,9)}
5	{(F,13),(J,11)}	{(S,5),(B,8),(E,8),(D,9),(H,11)}
6	{(F,13),(K,16)}	{(S,5),(B,8),(E,8),(D,9),(H,11),(J,11)}
7	{(K,16),(G,18)}	{(S,5),(B,8),(E,8),(D,9),(H,11),(J,11),(F,13)}
8	{(T,17),(G,18)}	{(S,5),(B,8),(E,8),(D,9),(H,11),(J,11),(F,13),(K,16)}
9	{(G,18)}	{(S,5),(B,8),(E,8),(D,9),(H,11),(J,11),(F,13),(K,16),(T,17)}

Path : S - D - H - J - K - T

Total Cost: 17

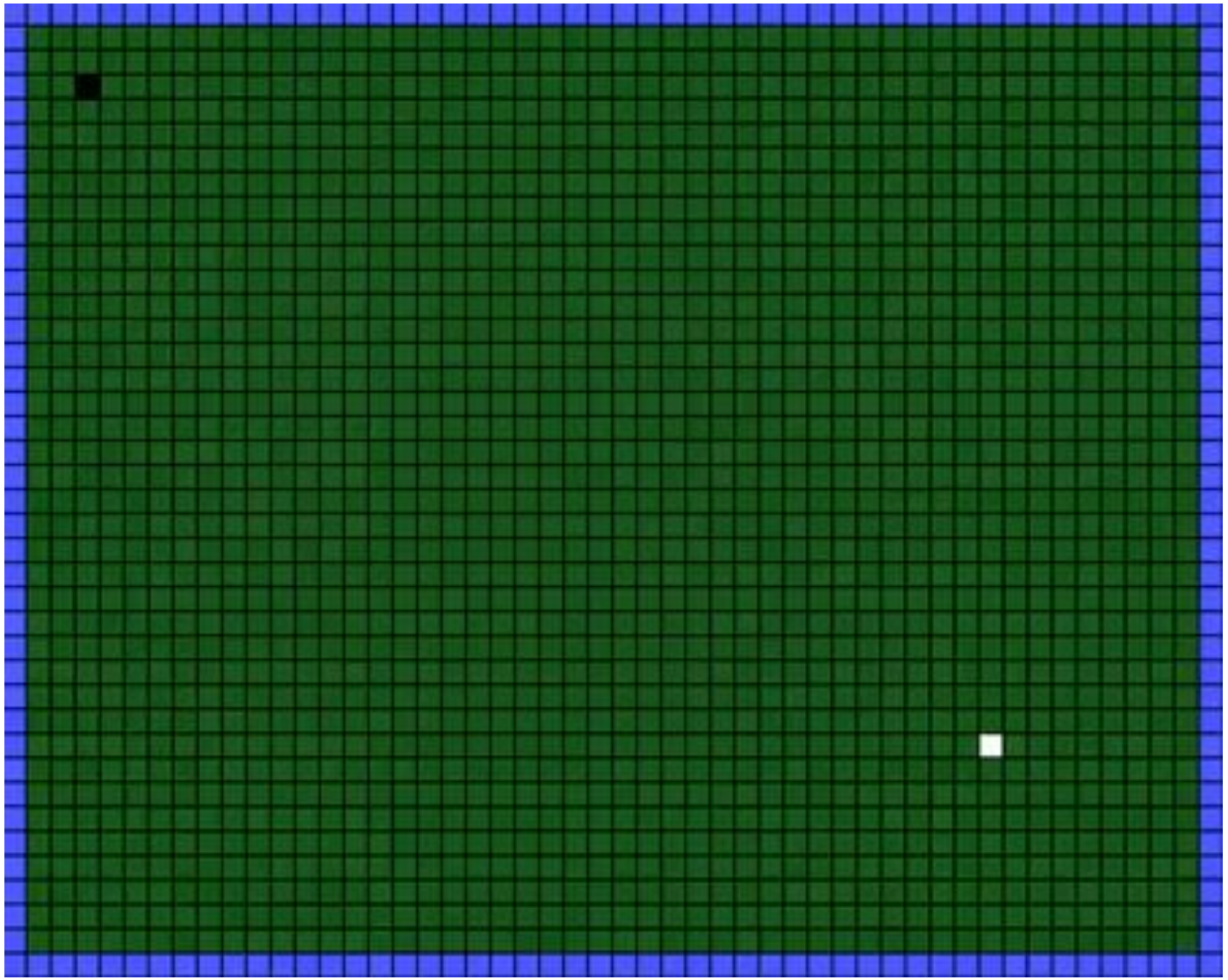
A* Esempio

https://www.youtube.com/watch?v=eSOJ3ARN5FM

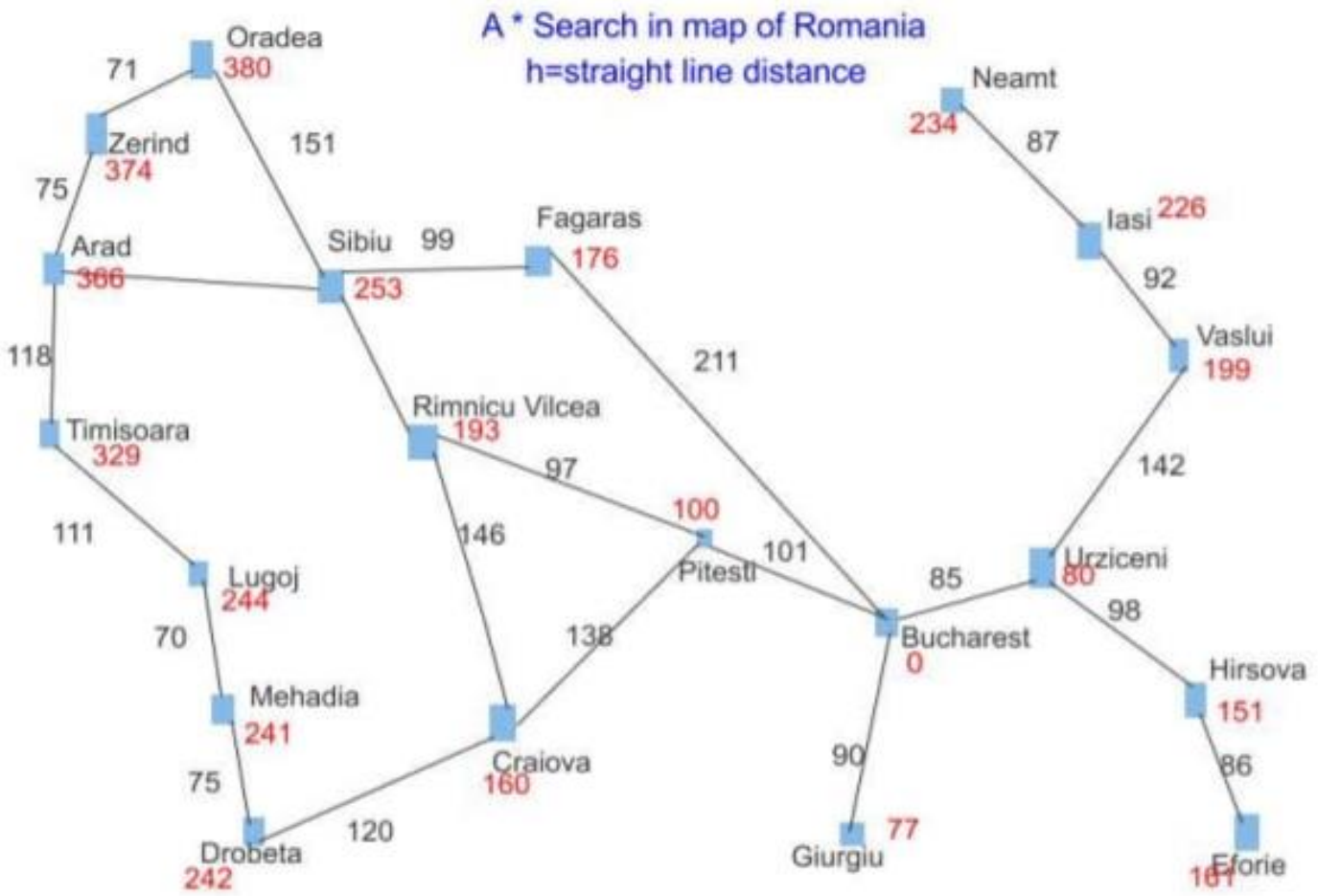


Astar

https://www.youtube.com/watch?v=19h1g22hby8

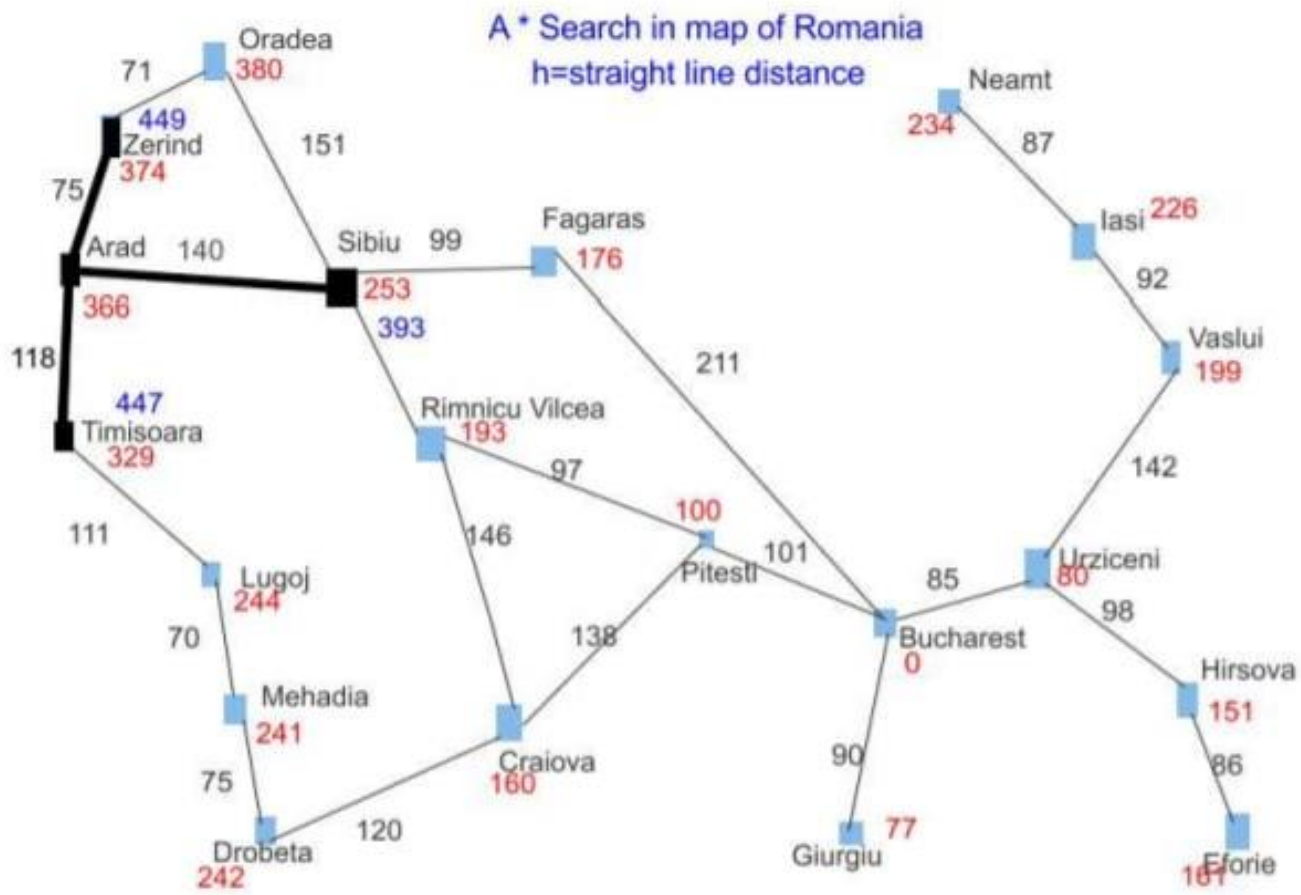


Map Navigation con Astar 1/6



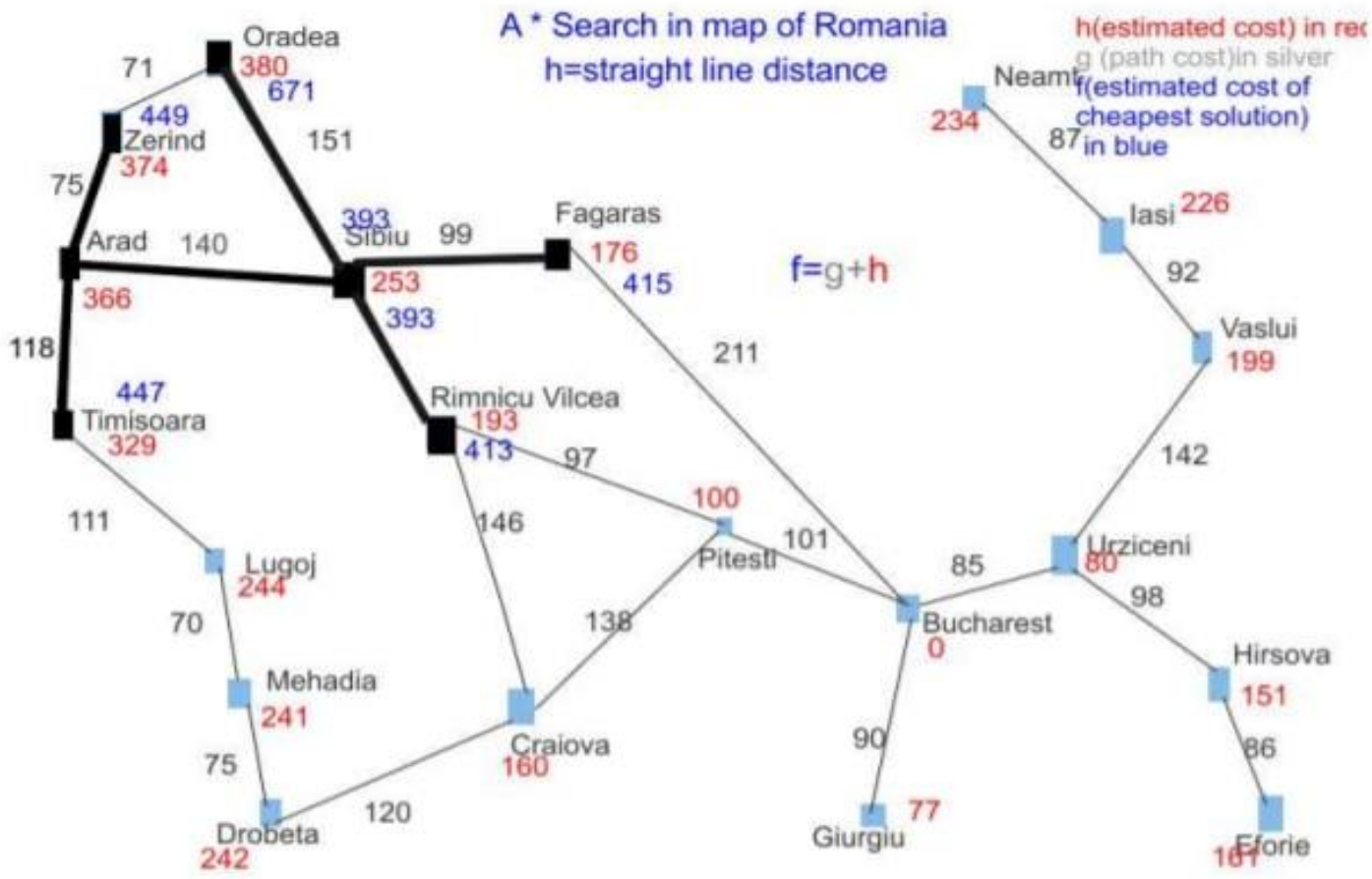
- **Start**: Arad;
- **Goal**: Bucarest;
- In rosso i valori della funzione euristica;
- In grigio i valori di costo del percorso;
- In blu i valori di f(n);
- h(n) indica la distanza in linea d'aria (in Km) da Bucarest;

Map Navigation con Astar 2/6



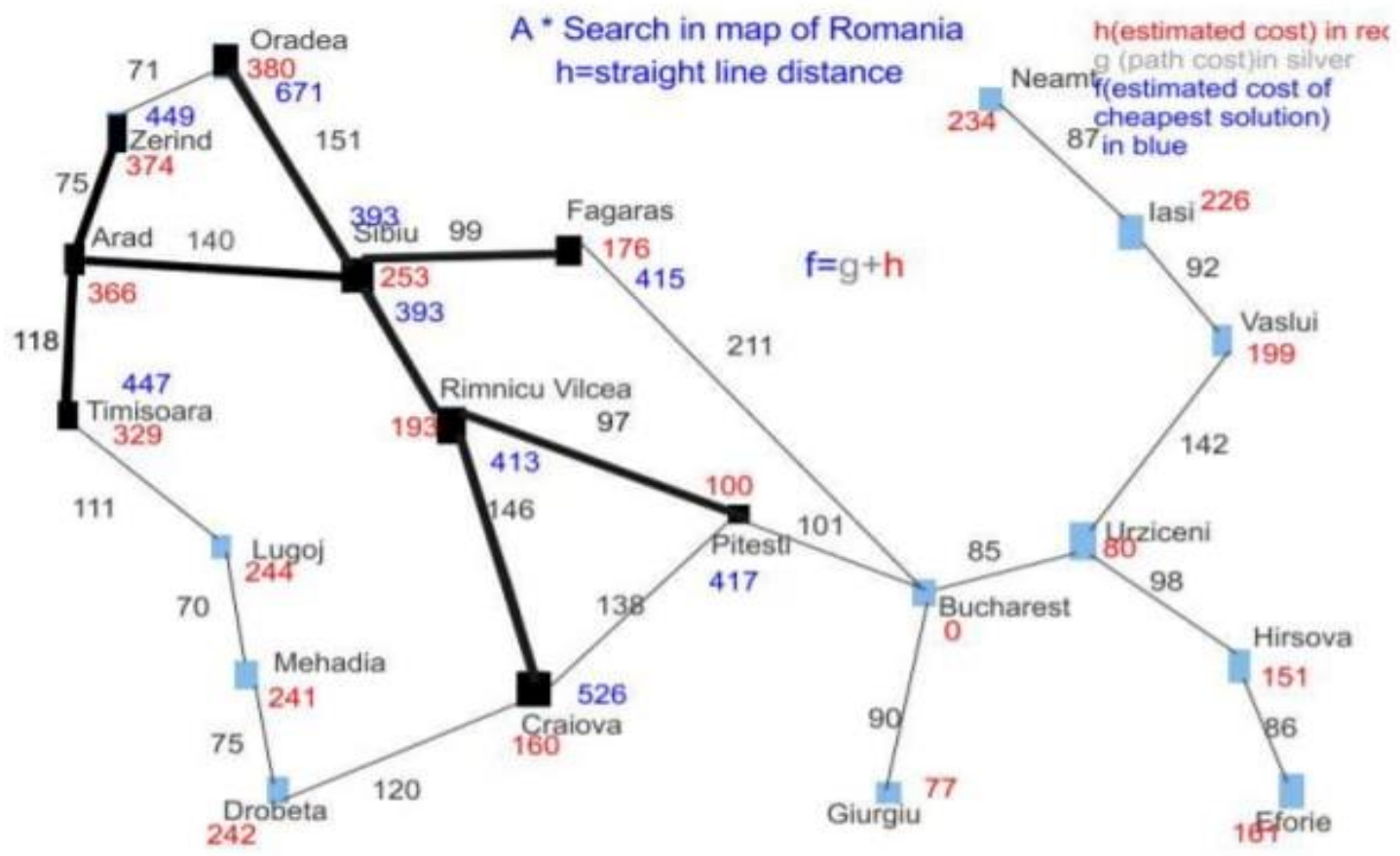
- **Start**: Arad;
- **Goal**: Bucarest;
- In **rosso** i valori della funzione euristica;
- In **nero** i valori di costo del percorso;
- In **blu** i valori di f(n);
- h(n) indica la distanza in linea d'aria (in Km) da Bucarest;

Map Navigation con Astar 3/6



- **Start**: Arad;
- **Goal**: Bucarest;
- In **rosso** i valori della funzione euristica;
- In **nero** i valori di costo del percorso;
- In **blu** i valori di f(n);
- h(n) indica la distanza in linea d'aria (in Km) da Bucarest;

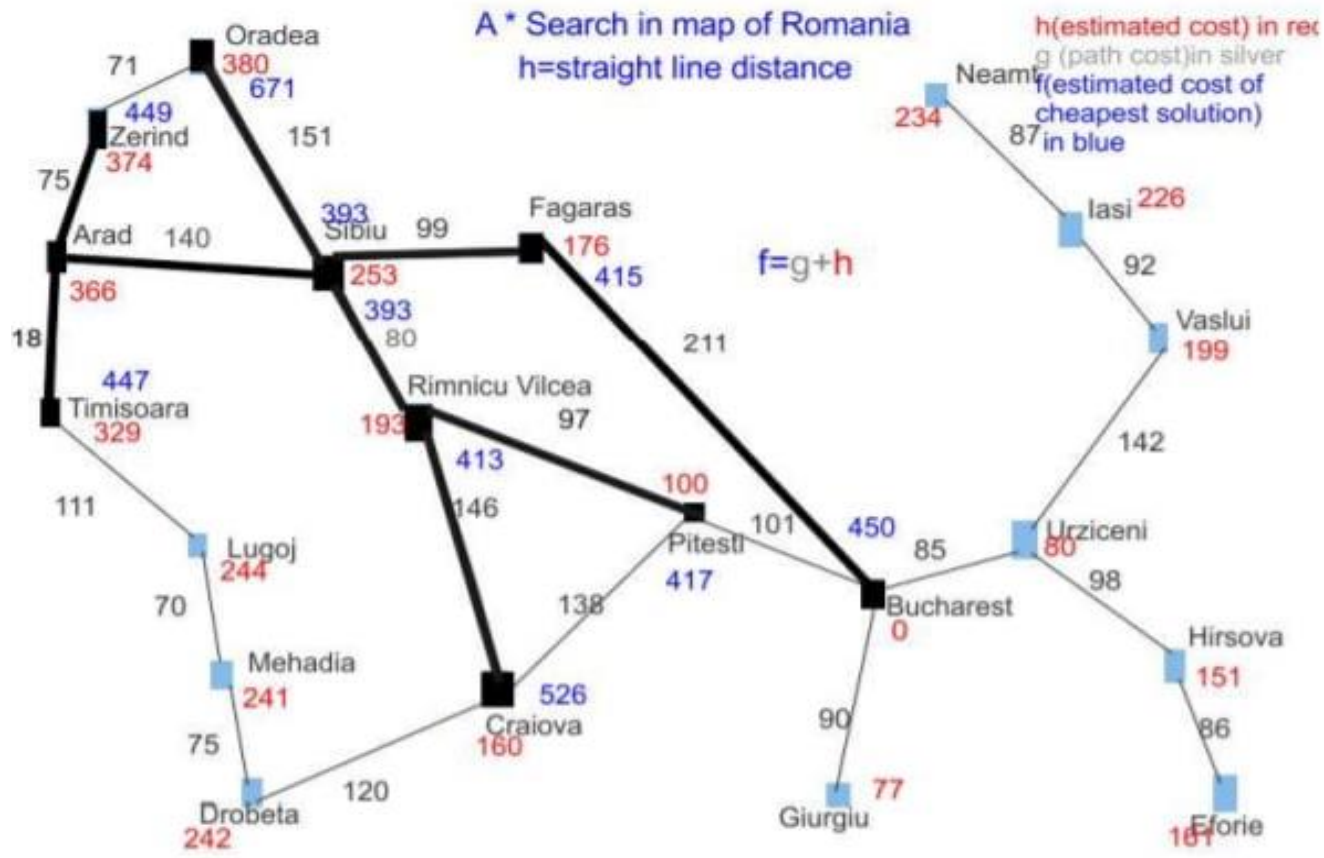
Map Navigation con Astar 4/6



- **Start**: Arad;
- **Goal**: Bucarest;
- In **rosso** i valori della funzione euristica;
- In **nero** i valori di costo del percorso;
- In **blu** i valori di f(n);
- h(n) indica la distanza in linea d'aria (in Km) da Bucarest;

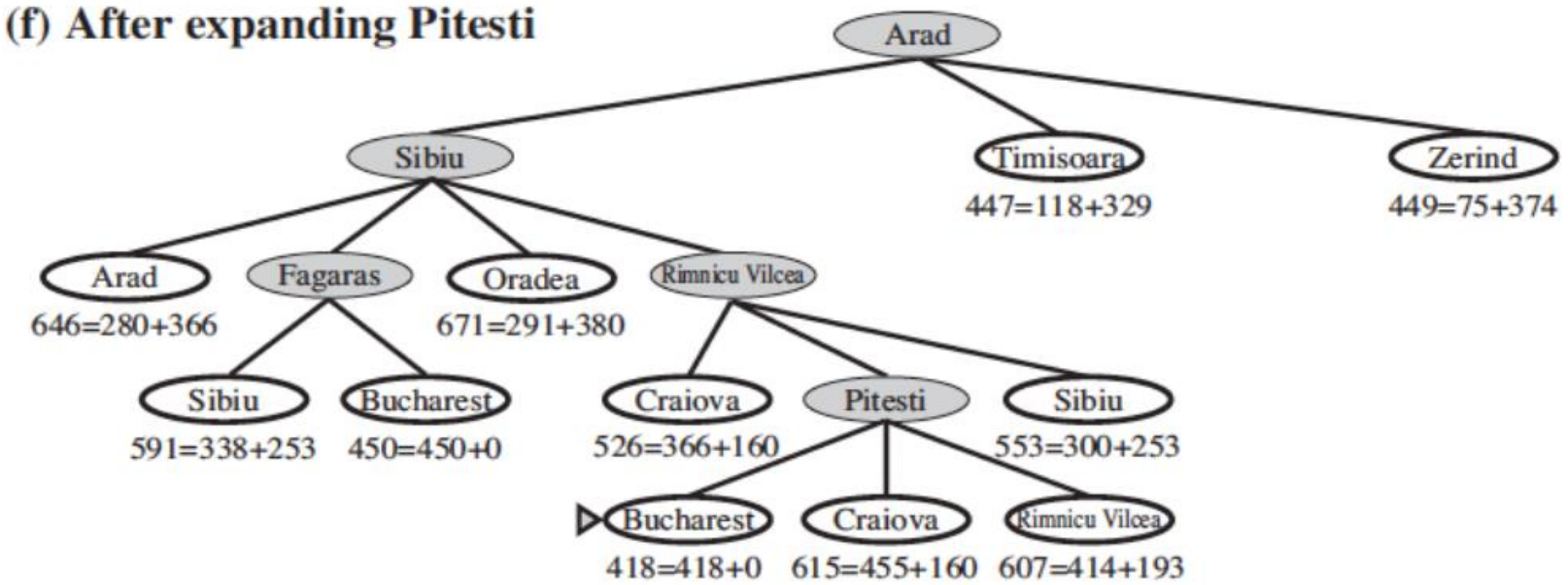
ATTENZIONE: Fagaras presenta una f(n) migliore!

Map Navigation con Astar 5/6



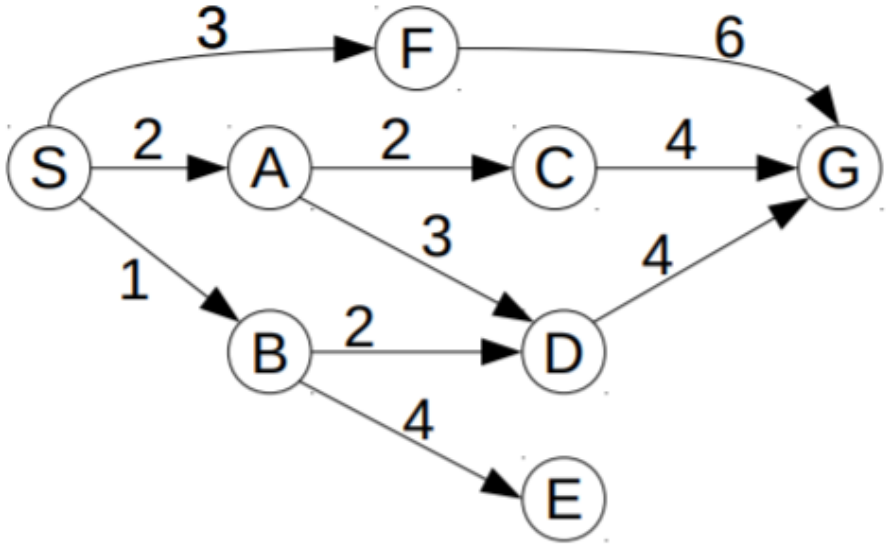
- **Start**: Arad;
- **Goal**: Bucarest;
- In **rosso** i valori della funzione euristica;
- In **nero** i valori di costo del percorso;
- In **blu** i valori di f(n);
- h(n) indica la distanza in linea d'aria (in Km) da Bucarest;

Map Navigation con Astar 6/6



Esercitazione Finale

state space



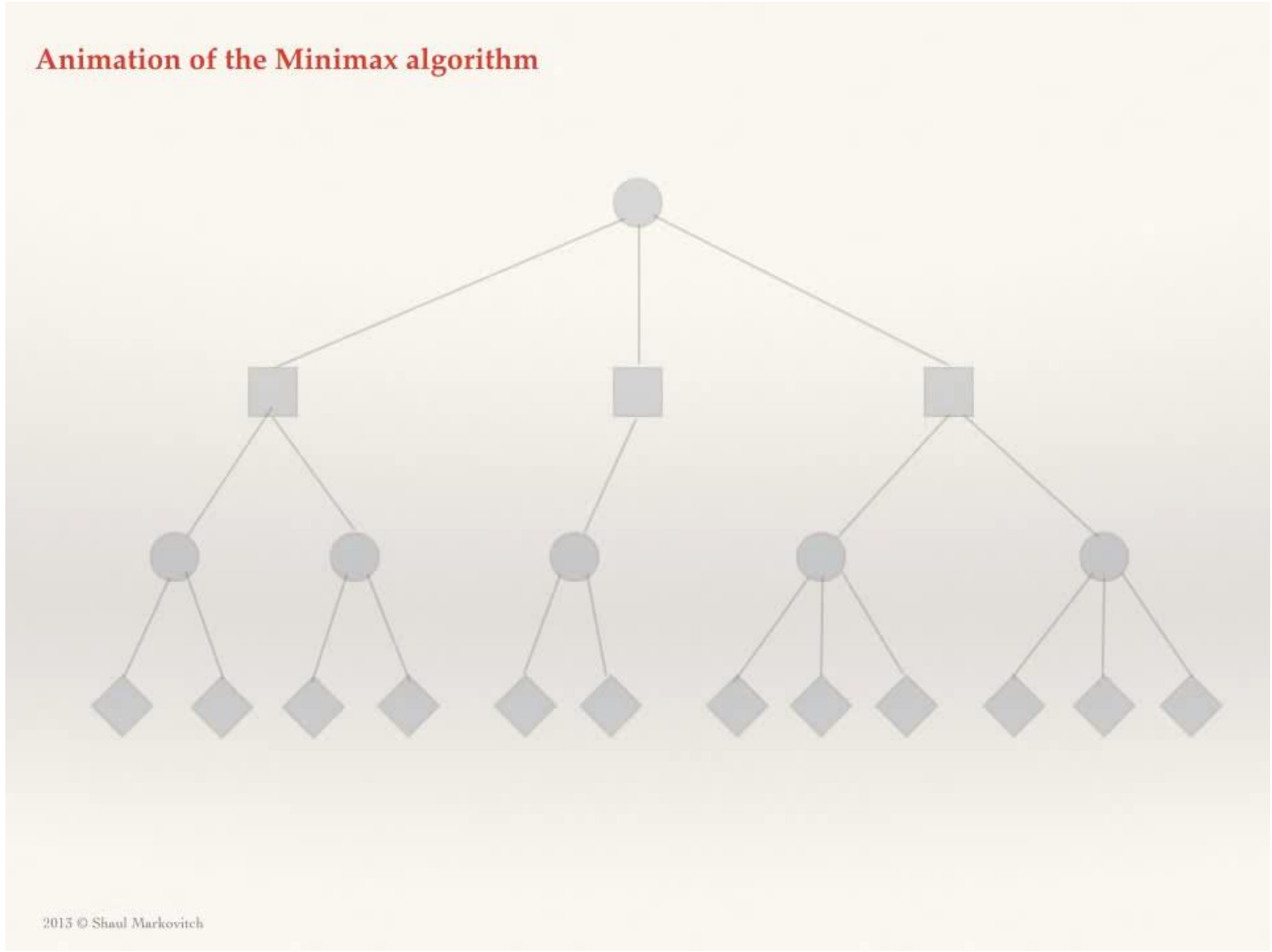
heuristic function (goal state: G)

S	A	B	C	D	E	F	G
6	4	5	2	2	8	4	0

- Trovare i primi 3 paths che dal nodo S conducono al nodo G usando gli algoritmi **BFS** e **DFS**;
- Trovare il path ottimo o (subottimo) che dal nodo S conduce al nodo G usando gli algoritmi **Best First Search**, **UCS**, **Astar**;

MiniMax

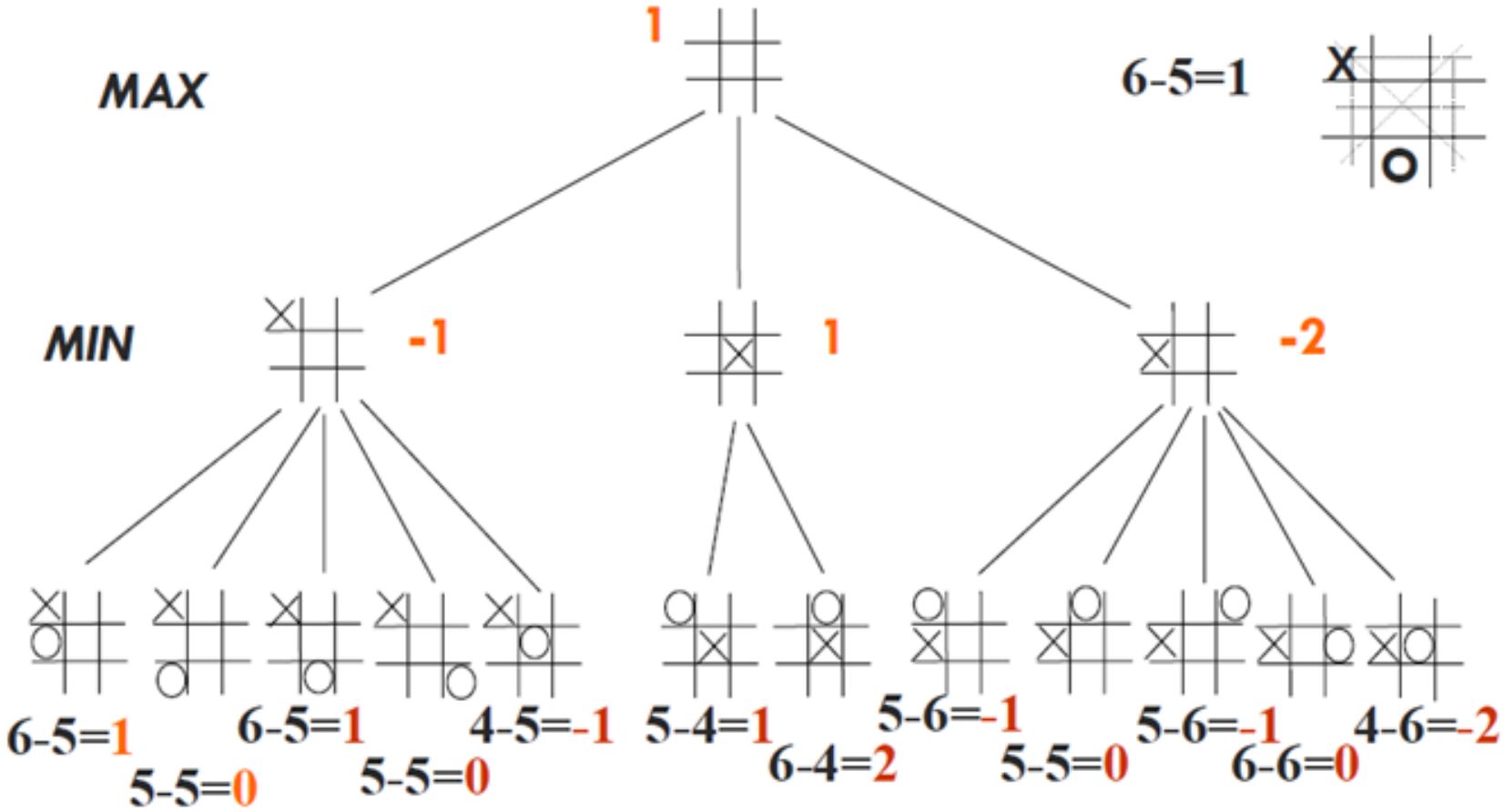
https://www.youtube.com/watch?v=zDskcx8FStA



Euristiche Per Giochi

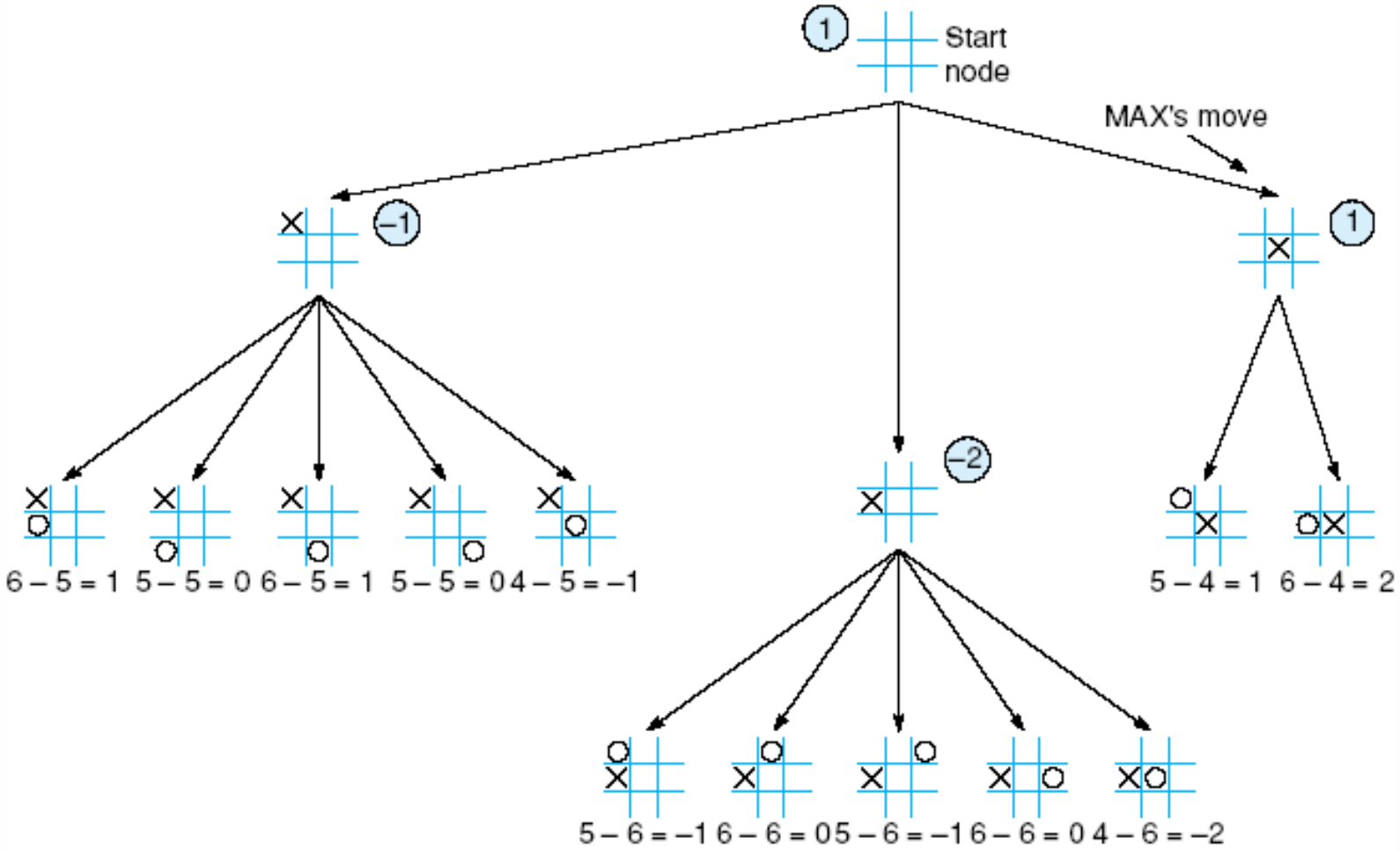
- **Funzioni lineari pesate**: $w_1 f_1 + w_2 f_2 + \ldots + w_n f_n$
  - **Negli scacchi**: 1 punto x Pedone, 3 x Alfiere, 3 x Cavallo, 5 x Torre, 9 x Regina;
  - **Nella dama**: 1 punto x pezzo; 10 punti per le dame;
  - **Nel tris**: conteggio numero righe/colonne/diagonali per X/O.

MiniMax Euristico: Esempio 1



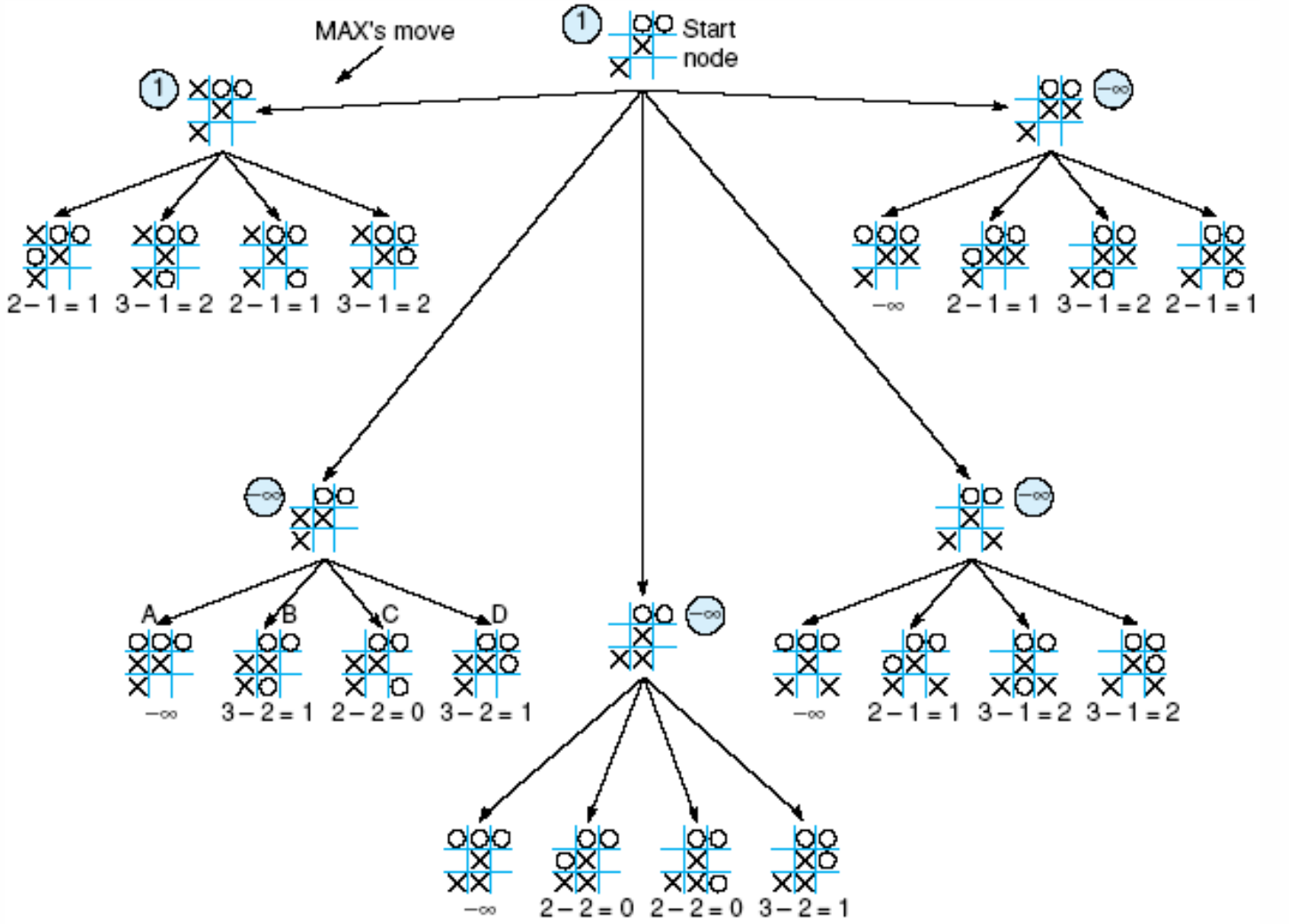
- Profondità P = 2;
- **funzione euristica: f(n)= O(n) - X(n)**;
- **X(n)** indica il numero di righe/colonne/diagonali aperte per il **simbolo X;**
- **O(n)** indica il numero di righe/colonne/diagonali aperte per il **simbolo O**.

MiniMax Euristico: Esempio 2



- Profondità P = 2;
- **funzione euristica: f(n)= O(n) - X(n)**;
- **X(n)** indica il numero di righe/colonne/diagonali aperte per il **simbolo X;**
- **O(n)** indica il numero di righe/colonne/diagonali aperte per il **simbolo O**.

MiniMax Euristico: Esempio 3



- Profondità P = 2;
- **funzione euristica: f(n)= O(n) - X(n)**;
- **X(n)** indica il numero di righe/colonne/diagonali aperte per il **simbolo X;**
- **O(n)** indica il numero di righe/colonne/diagonali aperte per il **simbolo O**.

Alpha Beta Pruning

https://www.youtube.com/watch?v=X_rfBIjqd-I

[alpha, beta]

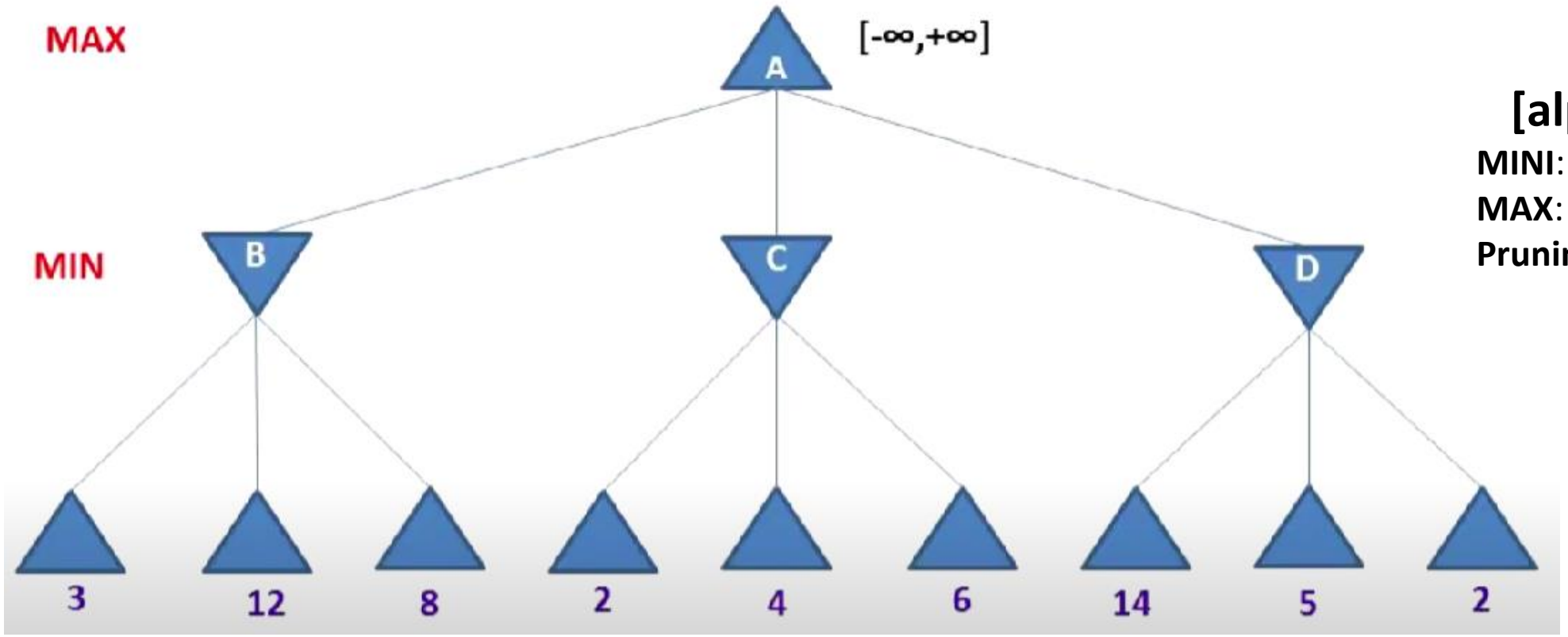
MINI: registra in **beta;**

MAX: registra in **alpha;**

Pruning se beta<=alpha

Esempio 3 Alpha-Beta Pruning 1/12

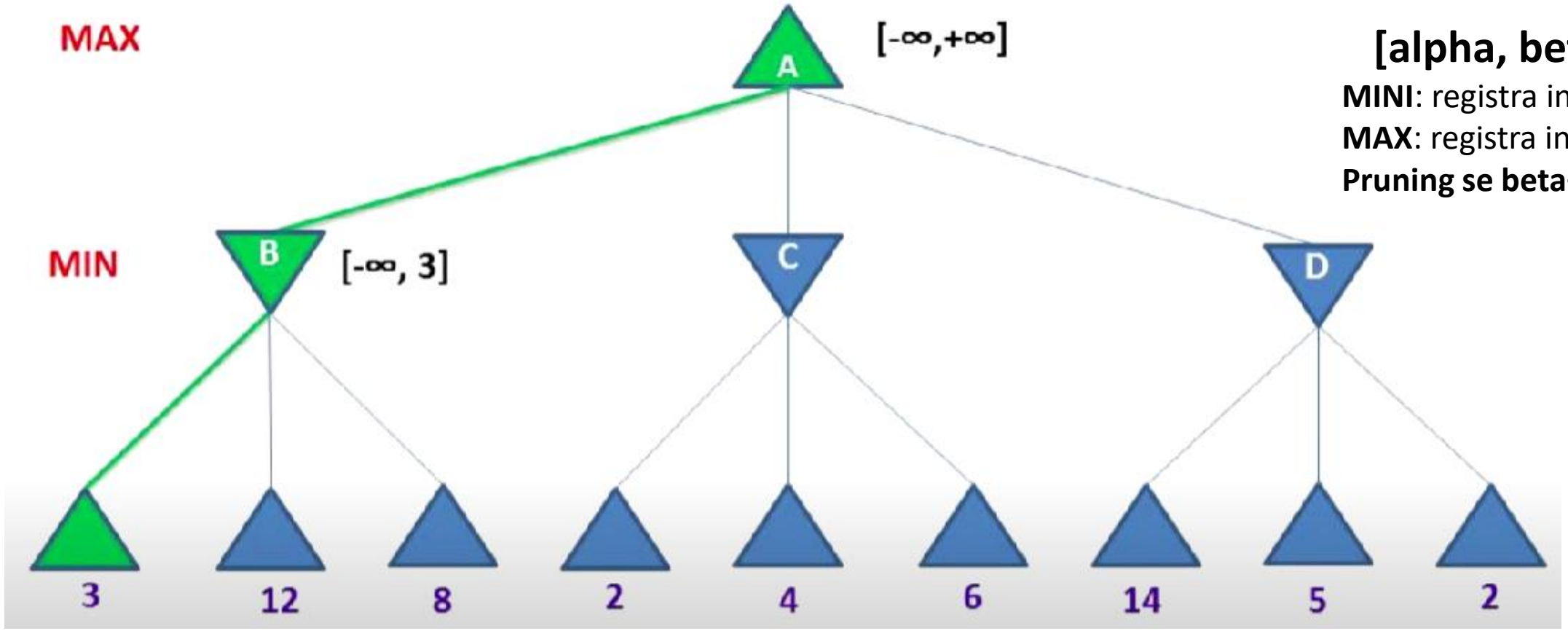
- **Fonte:** https://www.youtube.com/watch?v=X_rfBIjqd-I



[alpha, beta]

MINI: registra in **beta;**
MAX: registra in **alpha;**
Pruning se beta<=alpha

Esempio 3 Alpha-Beta Pruning 2/12



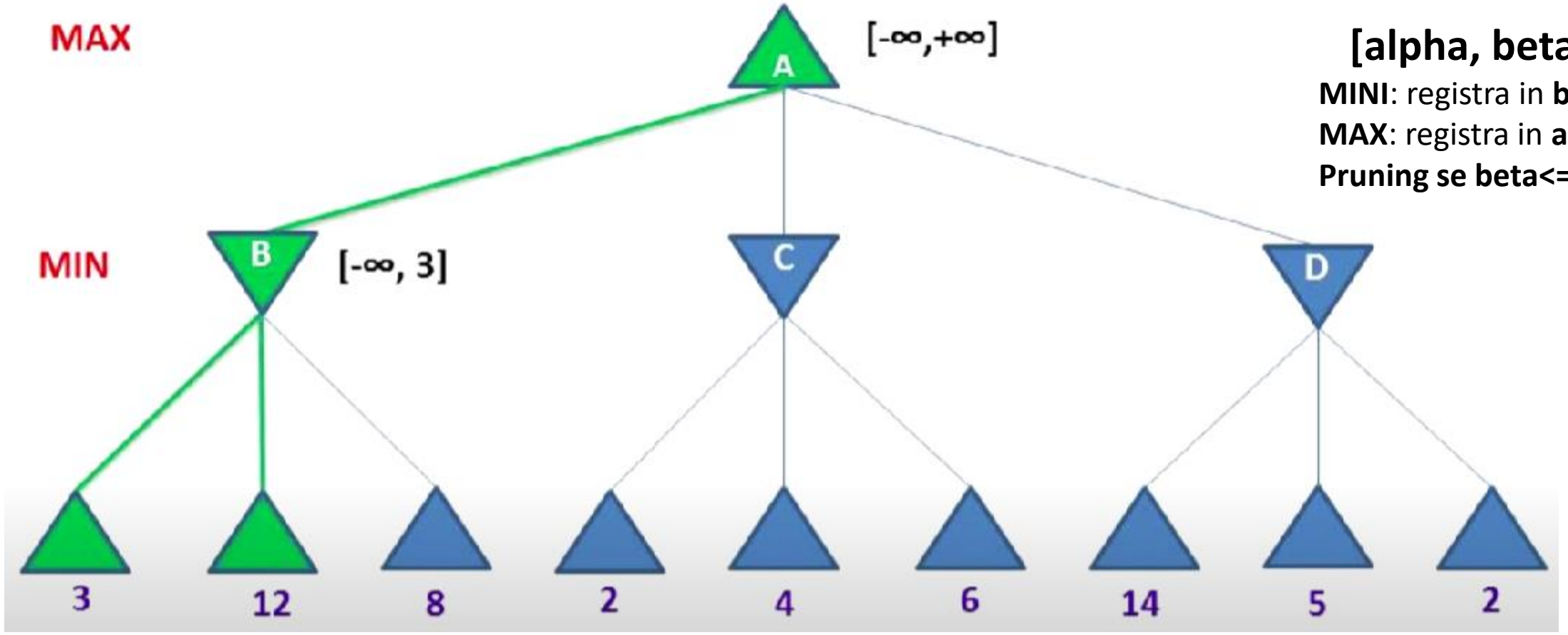
[alpha, beta]

MINI: registra in **beta;**

MAX: registra in **alpha;**

Pruning se beta<=alpha

Esempio 3 Alpha-Beta Pruning 3/12



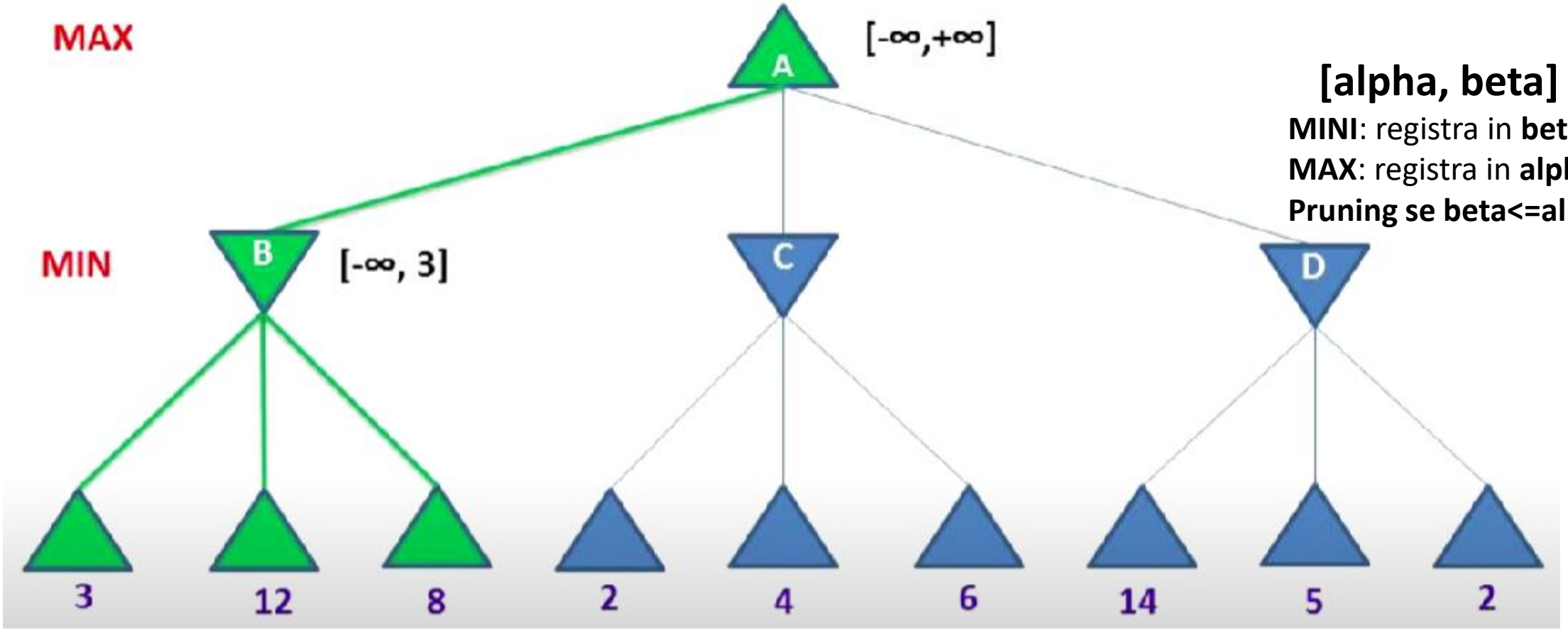
[alpha, beta]

MINI: registra in **beta;**

MAX: registra in **alpha;**

Pruning se beta<=alpha

Esempio 3 Alpha-Beta Pruning 4/12



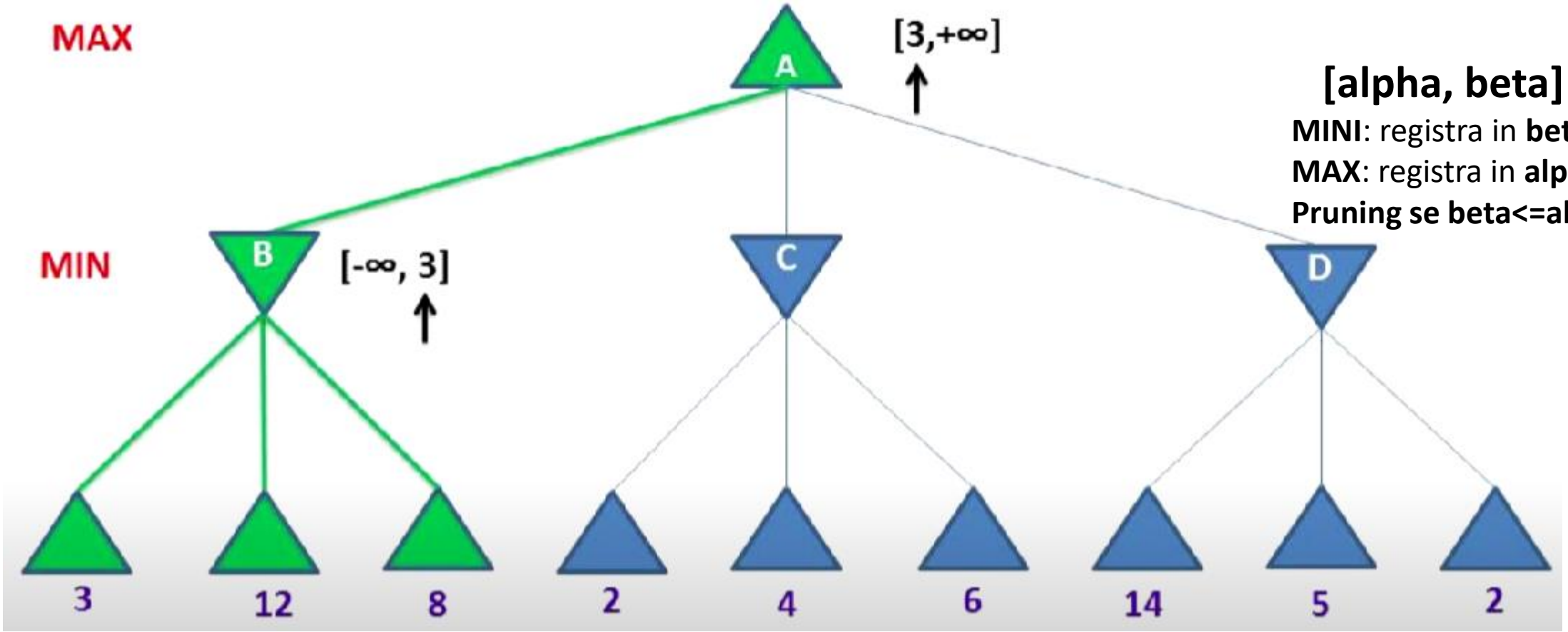
[alpha, beta]

MINI: registra in **beta;**

MAX: registra in **alpha;**

Pruning se beta<=alpha

Esempio 3 Alpha-Beta Pruning 5/12



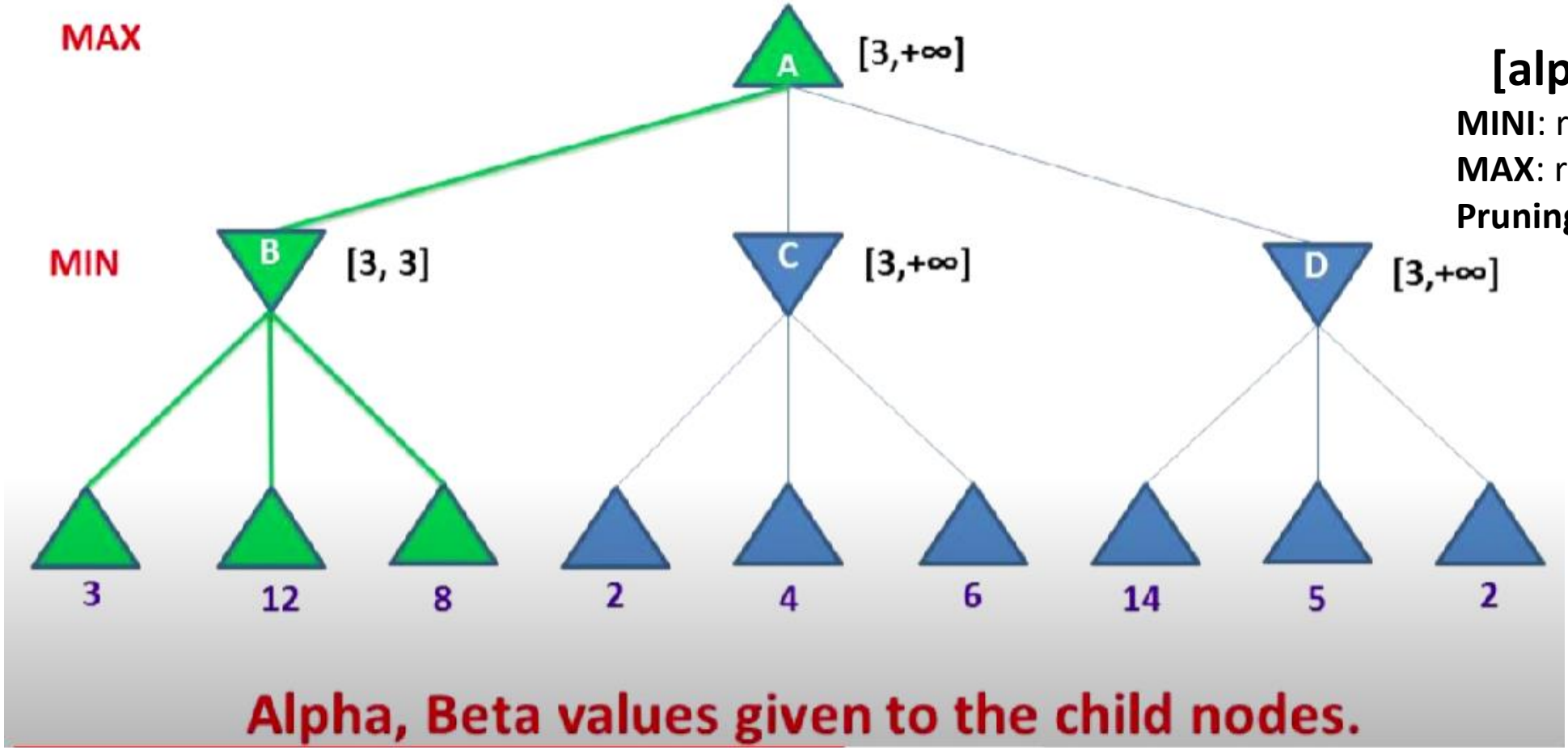
[alpha, beta]

MINI: registra in **beta;**

MAX: registra in **alpha;**

Pruning se beta<=alpha

Esempio 3 Alpha-Beta Pruning 6/12



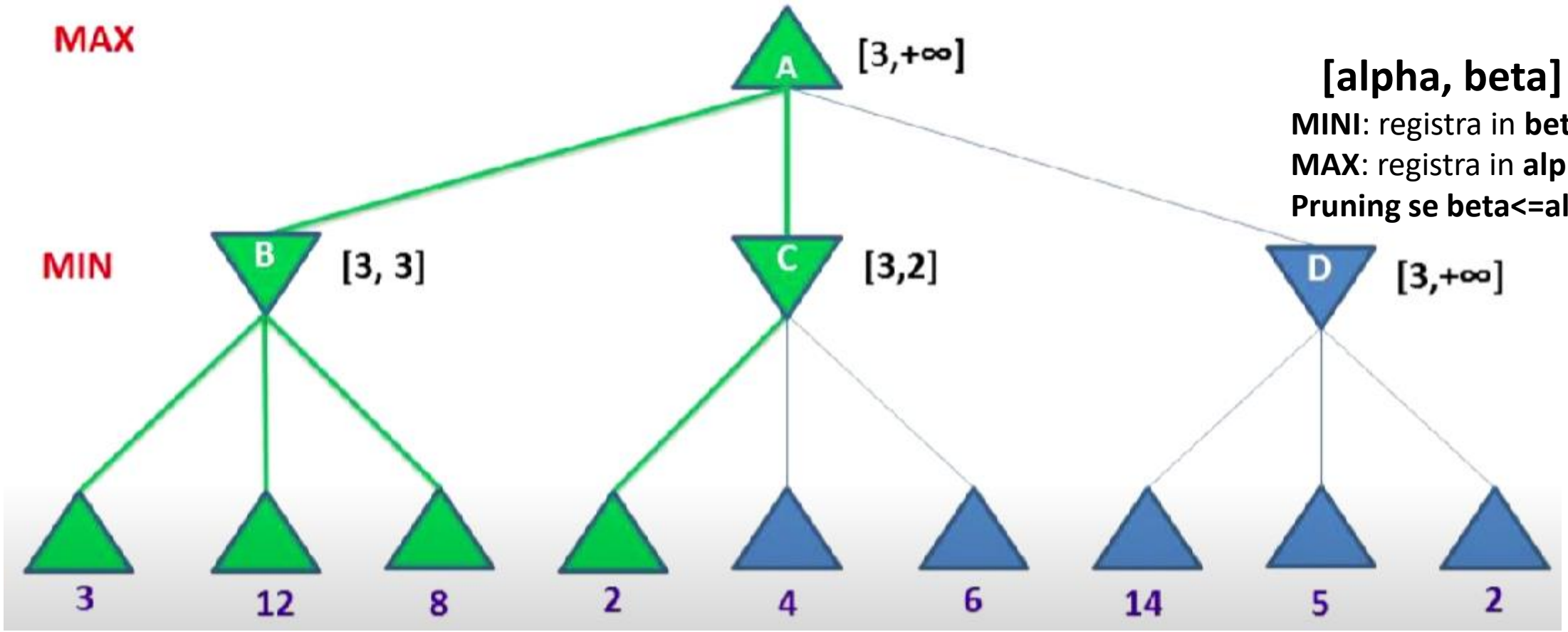
[alpha, beta]

MINI: registra in **beta;**

MAX: registra in **alpha;**

Pruning se beta<=alpha

Esempio 3 Alpha-Beta Pruning 7/12



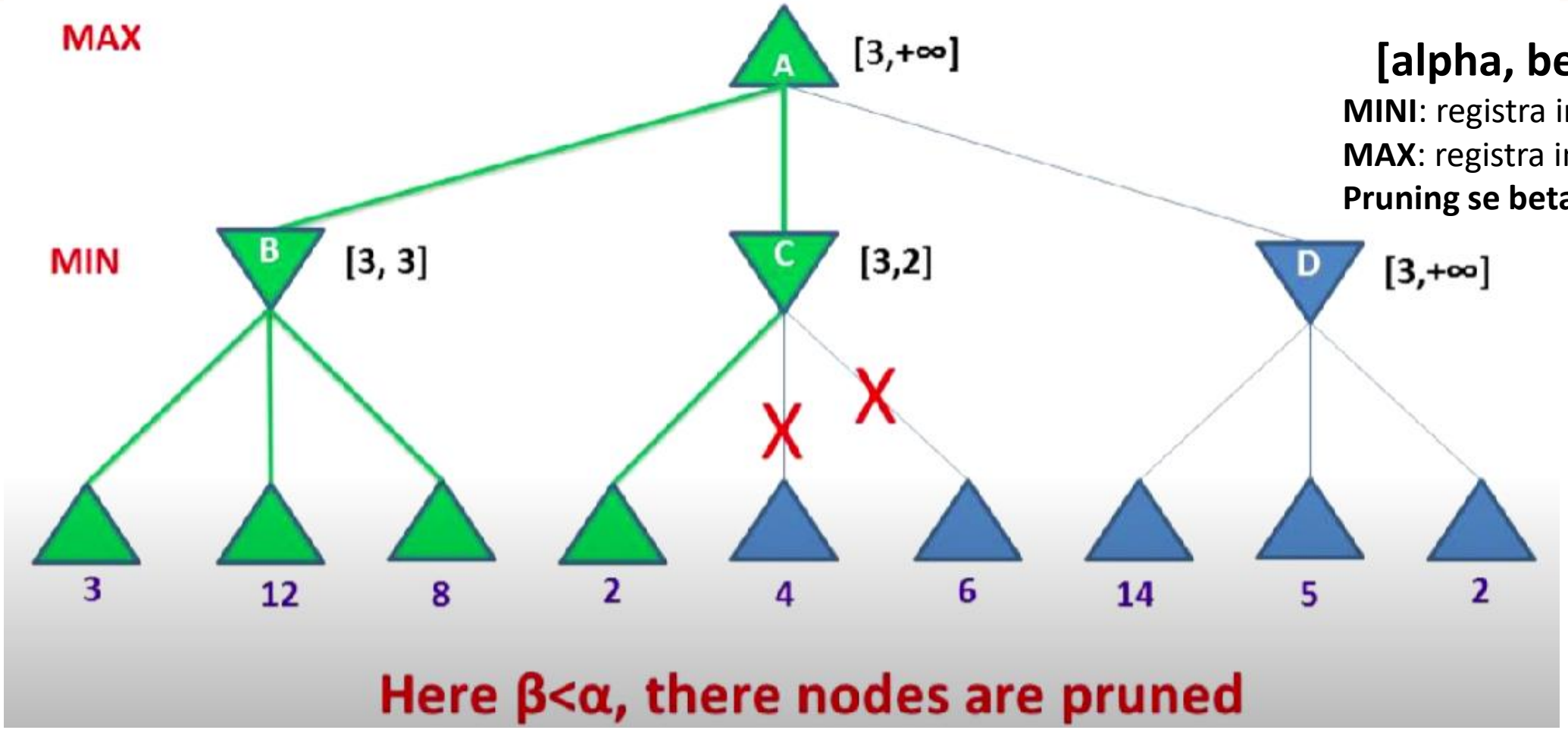
[alpha, beta]

MINI: registra in **beta;**

MAX: registra in **alpha;**

Pruning se beta<=alpha

Esempio 3 Alpha-Beta Pruning 8/12



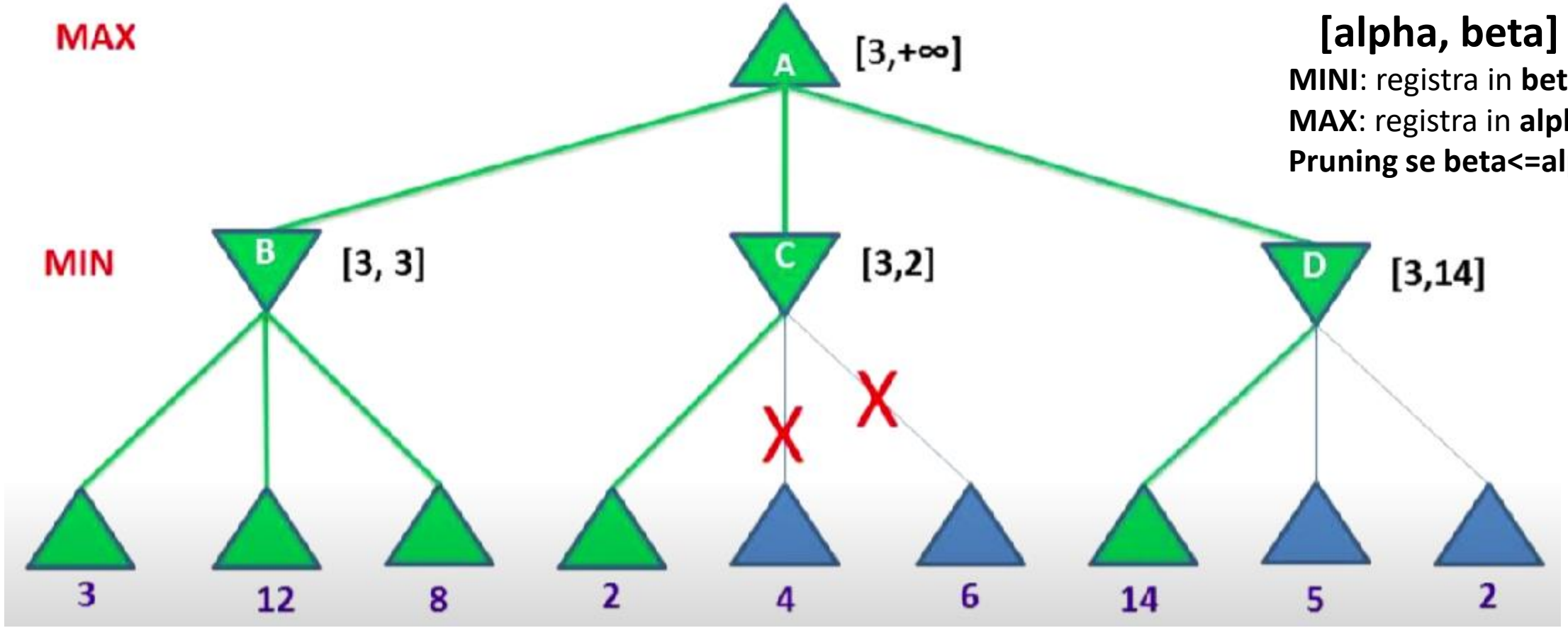
[alpha, beta]

MINI: registra in **beta;**

MAX: registra in **alpha;**

Pruning se beta<=alpha

Esempio 3 Alpha-Beta Pruning 9/12



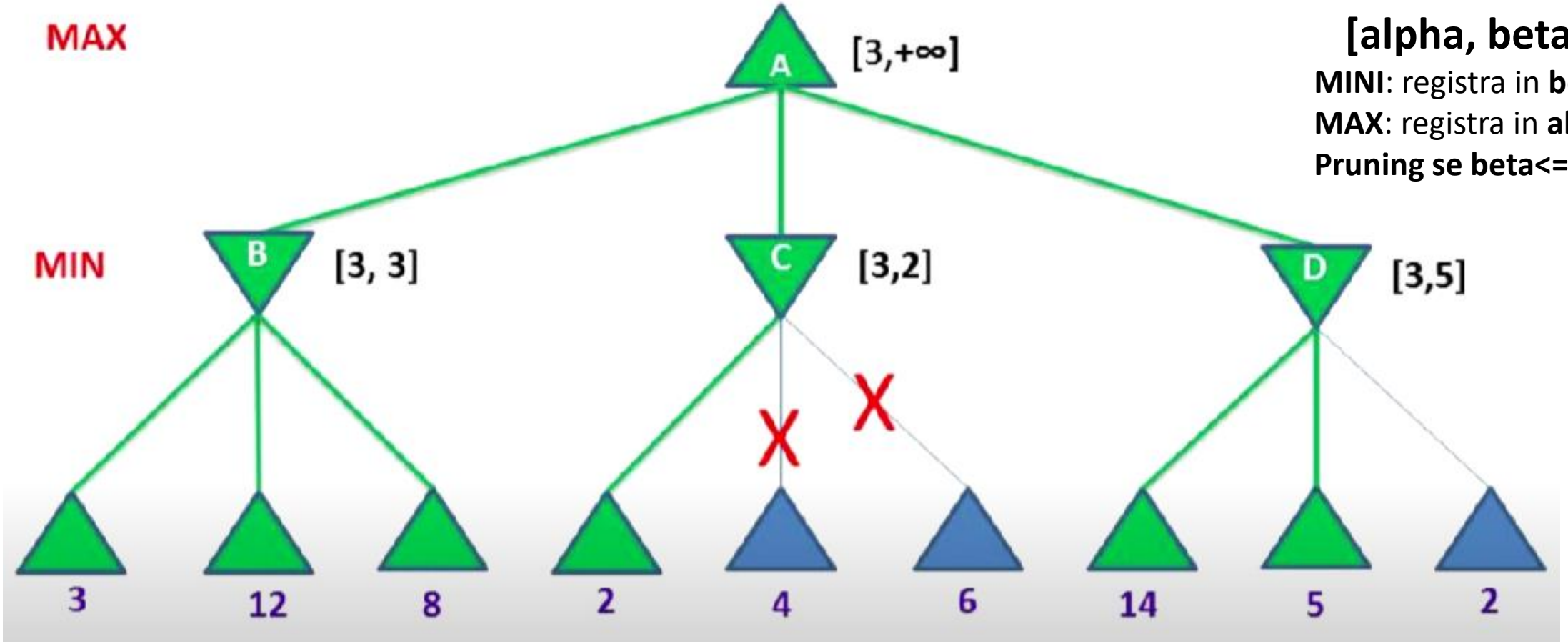
[alpha, beta]

MINI: registra in **beta;**

MAX: registra in **alpha;**

Pruning se beta<=alpha

Esempio 3 Alpha-Beta Pruning 10/12



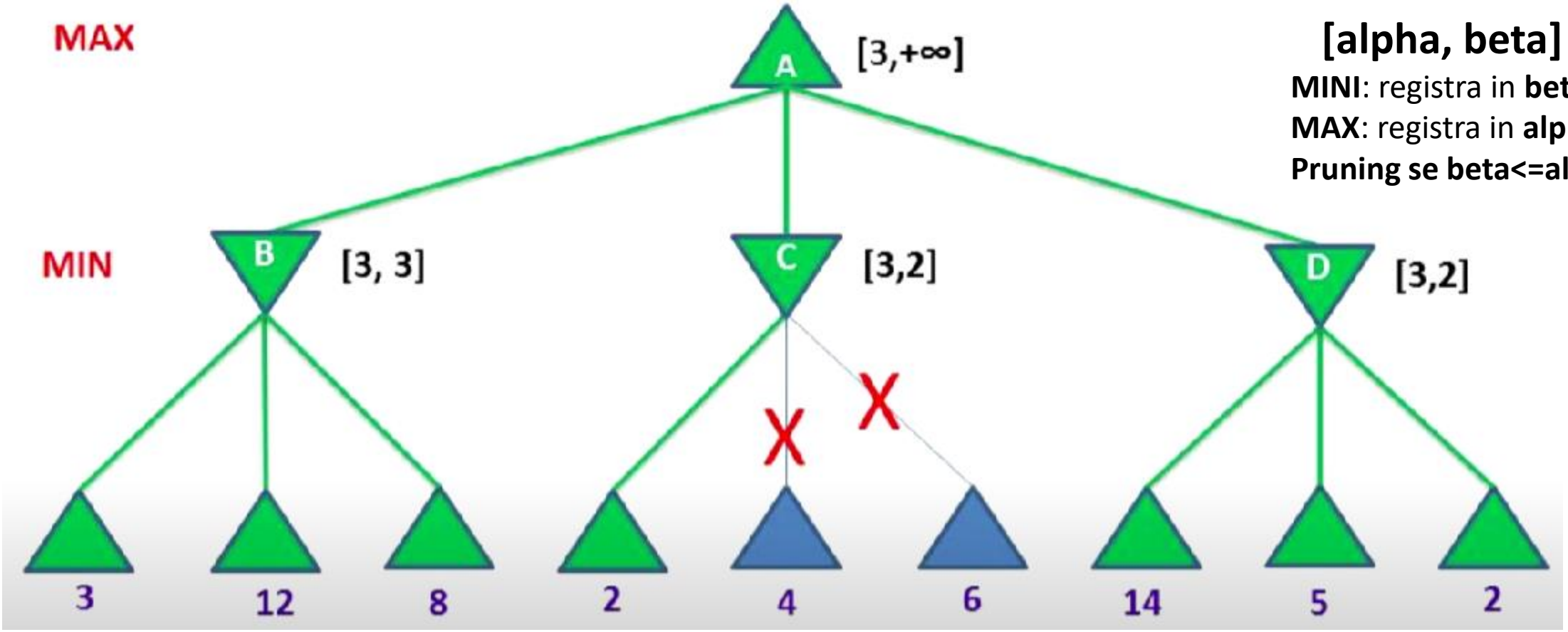
[alpha, beta]

MINI: registra in **beta;**

MAX: registra in **alpha;**

Pruning se beta<=alpha

Esempio 3 Alpha-Beta Pruning 11/12



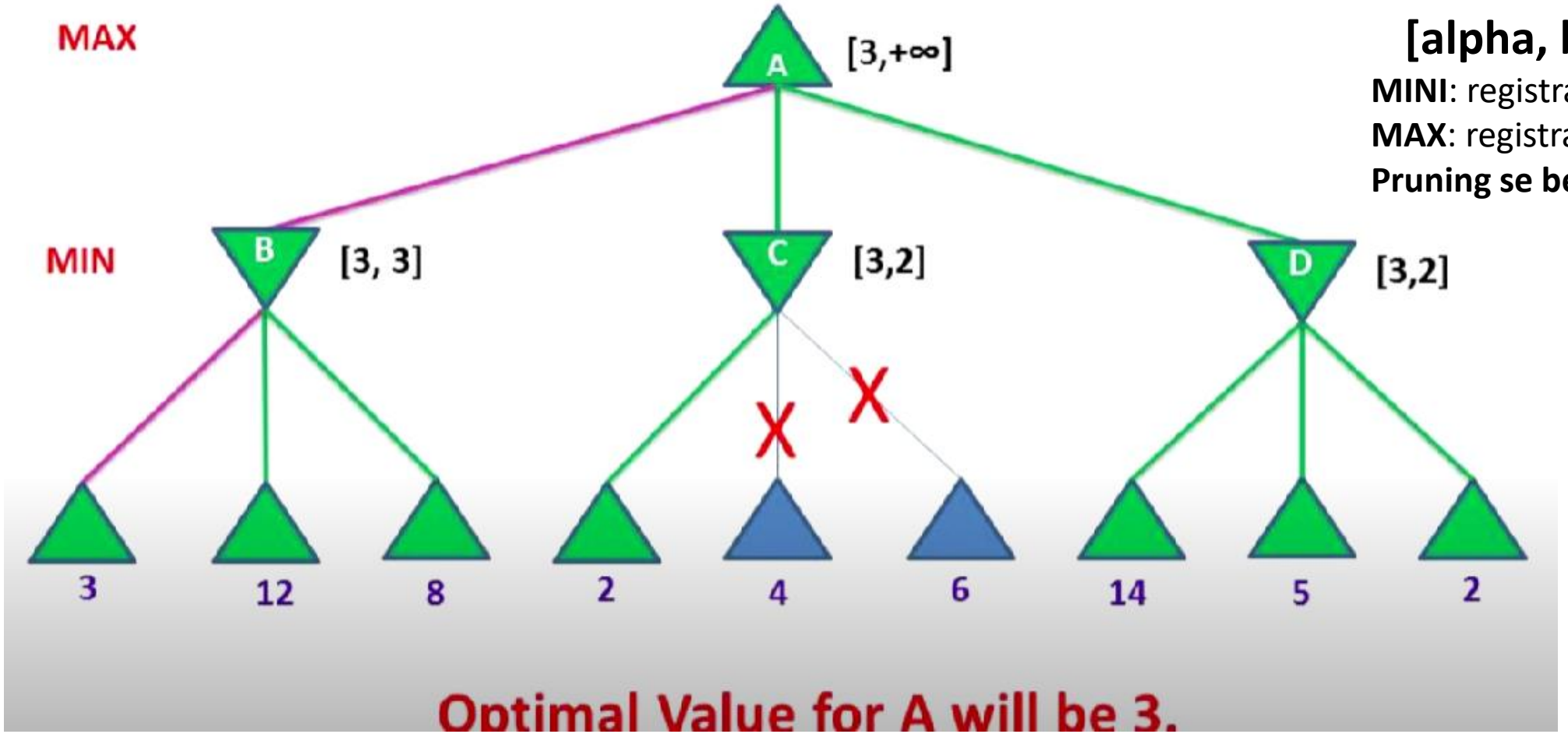
[alpha, beta]

MINI: registra in **beta;**

MAX: registra in **alpha;**

Pruning se beta<=alpha

Esempio 3 Alpha-Beta Pruning 12/12



[alpha, beta]

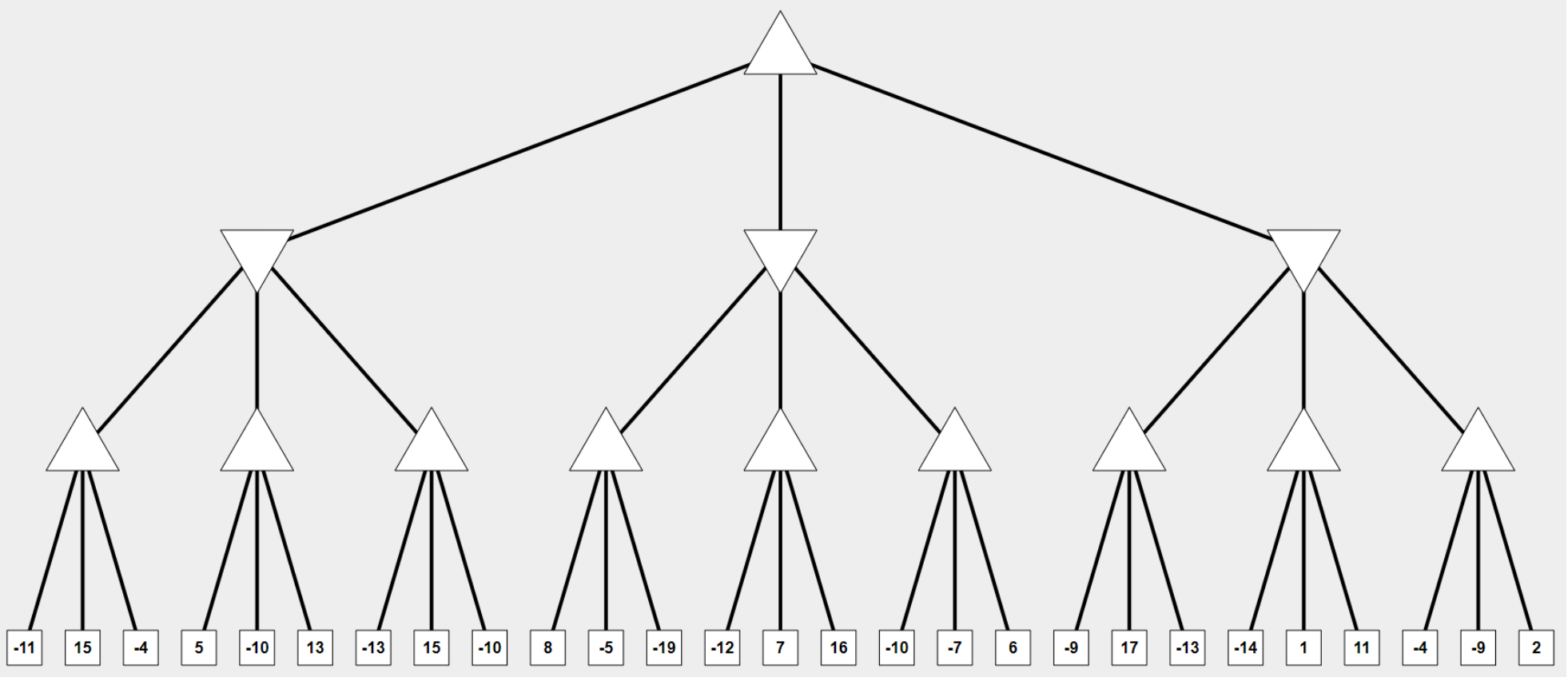
MINI: registra in **beta;**

MAX: registra in **alpha;**

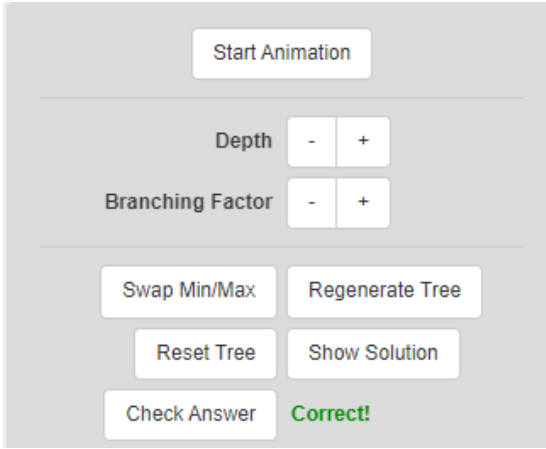
Pruning se beta<=alpha

Esercizio 1 Alpha-Beta Pruning 1/2

- **Tool Online**: https://pascscha.ch/info2/abTreePractice/

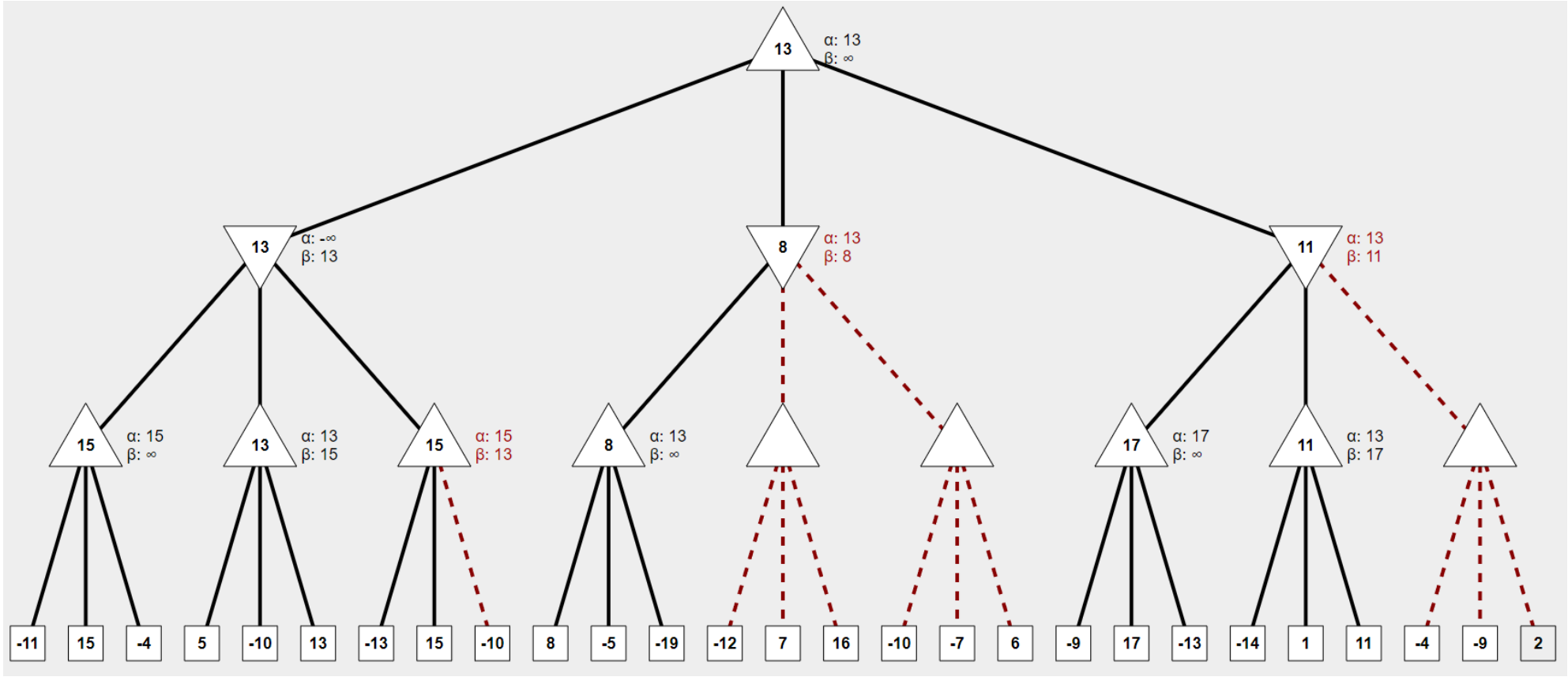


MINI: registra in **beta;**
MAX: registra in **alpha;**
Pruning se beta<=alpha



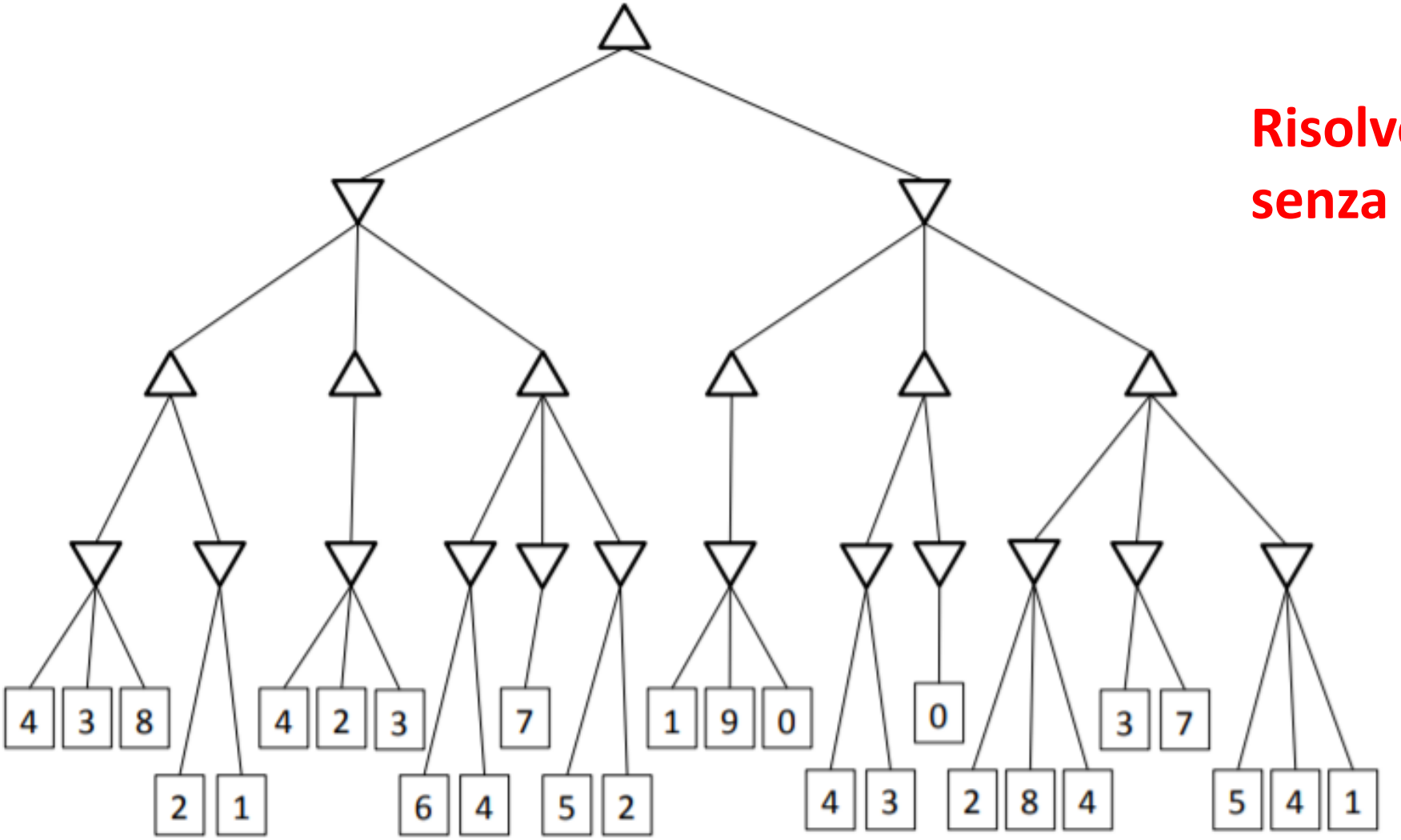
Esercizio 1 Alpha-Beta Pruning 2/2

- **Tool Online**: https://pascscha.ch/info2/abTreePractice/



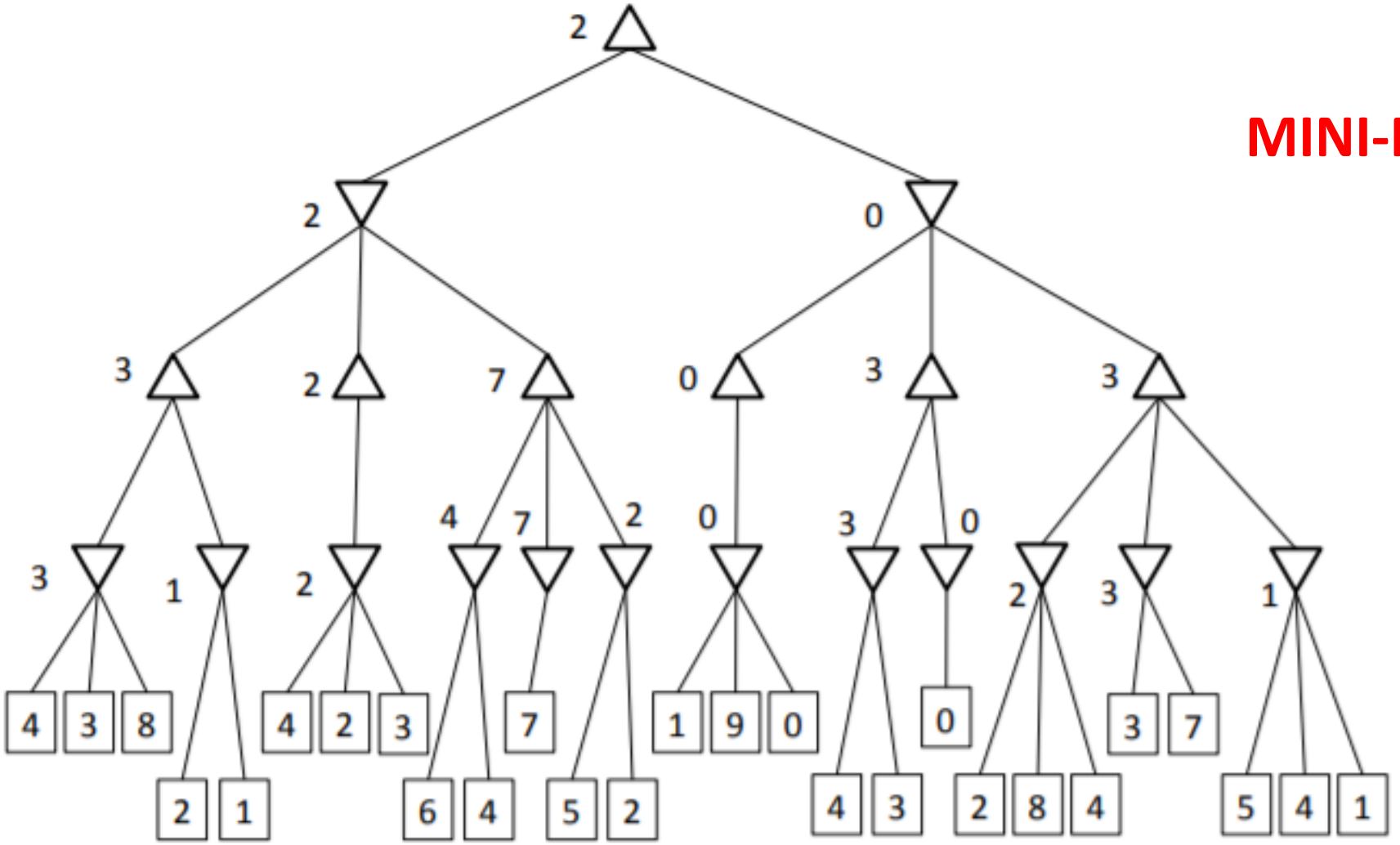
MINI: registra in **beta;**
MAX: registra in **alpha;**
Pruning se beta<=alpha

Esercizio 2 Alpha-Beta Pruning 1/3



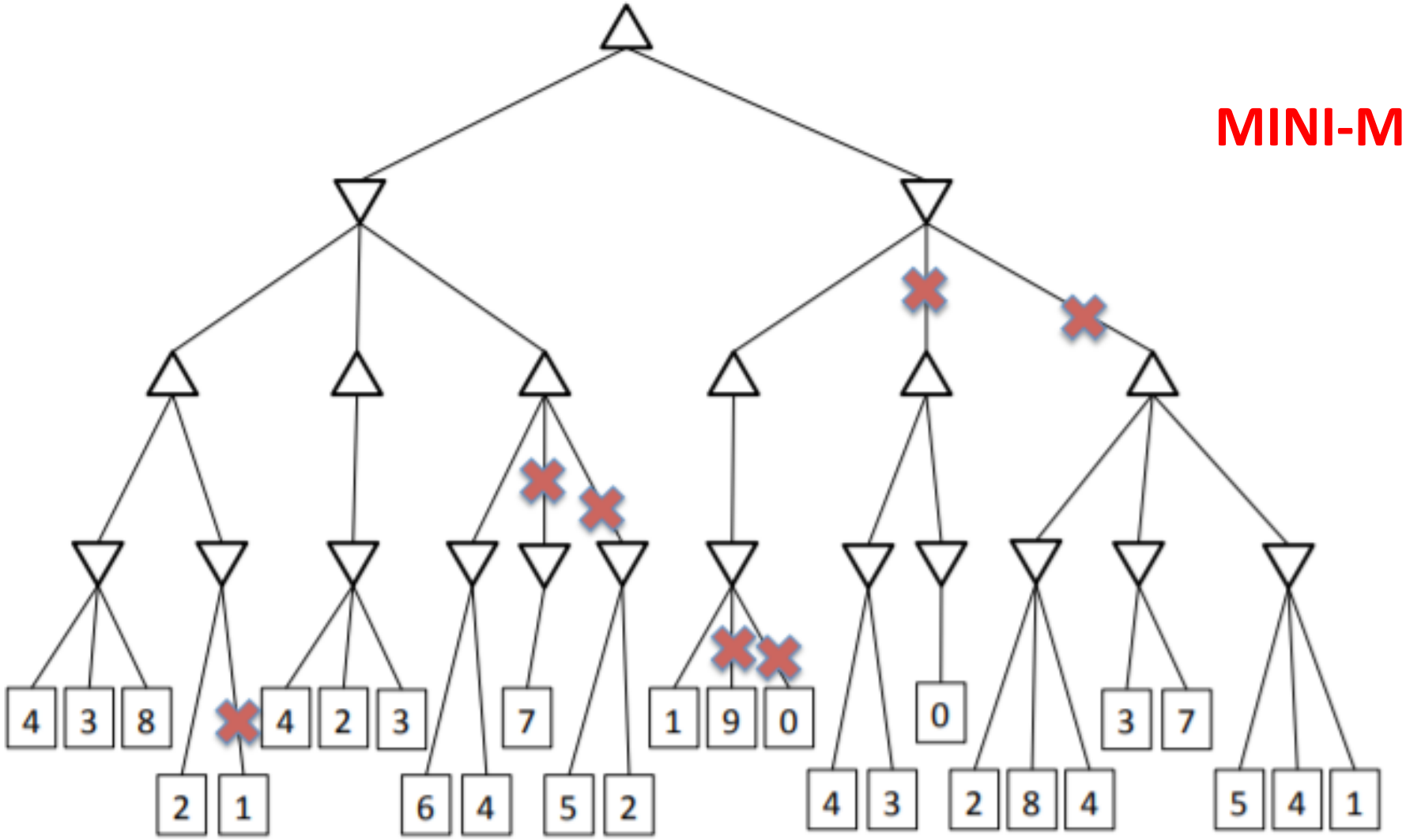
Risolvere l'esercizio con e senza Pruning

Esercizio 2 Alpha-Beta Pruning 2/3



MINI-MAX senza Pruning

Esercizio 2 Alpha-Beta Pruning 3/3



MINI-MAX con Pruning