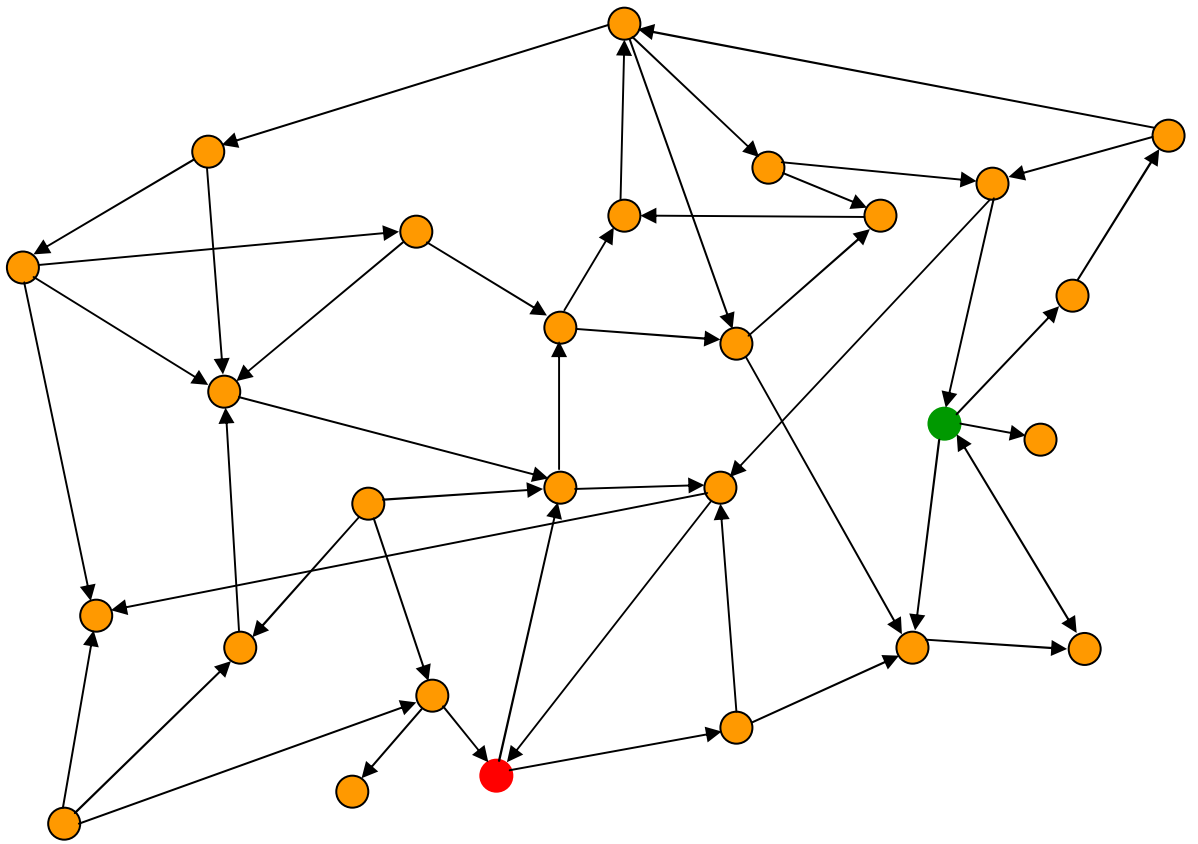
Intelligenza Artificiale

Metodi di ricerca

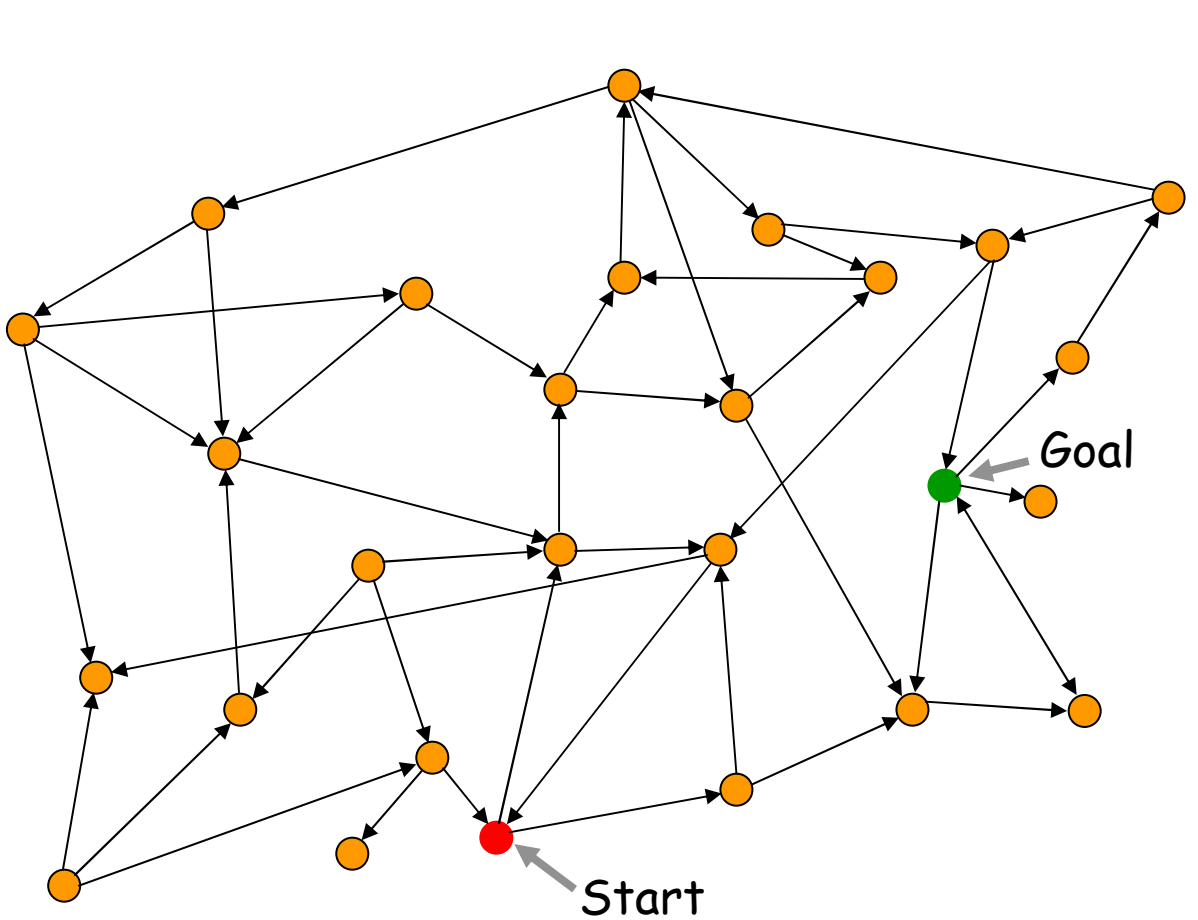
Marco Piastra

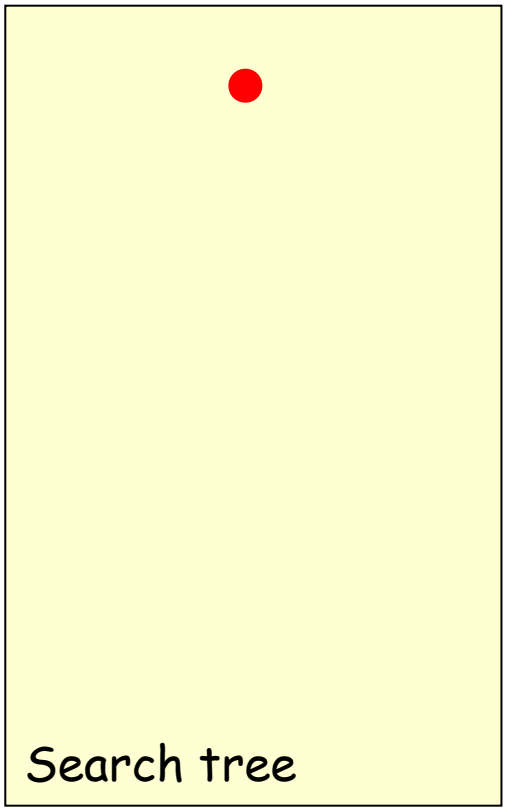
Ricerca nello spazio degli stati (disegno di J.C. Latombe)



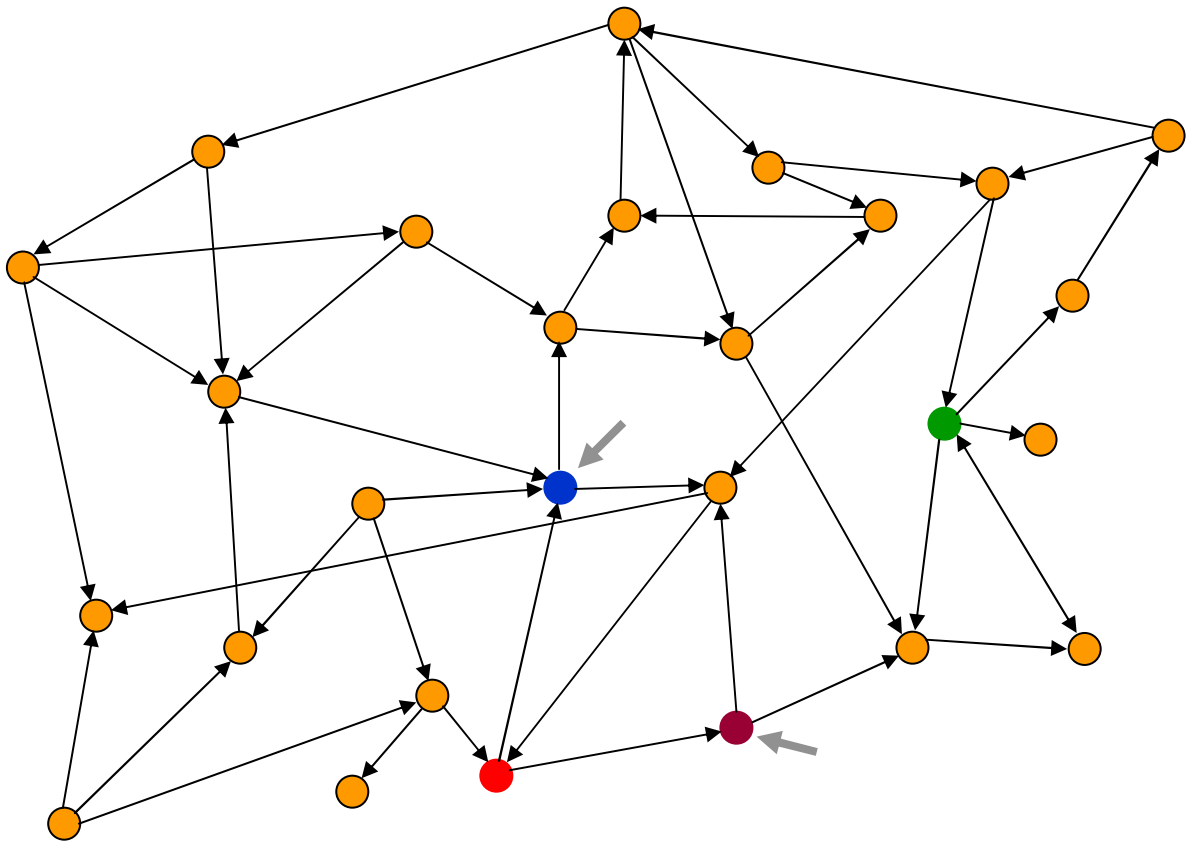
- I nodi rappresentano uno **stato**
- Gli archi (orientati) una **transizione** (potrebbero avere un **costo**)
- La tecnica consiste nell'esplorazione del grafo
- Non sempre è possibile/necessario costruire il grafo completo
- La tecnica deve essere efficiente

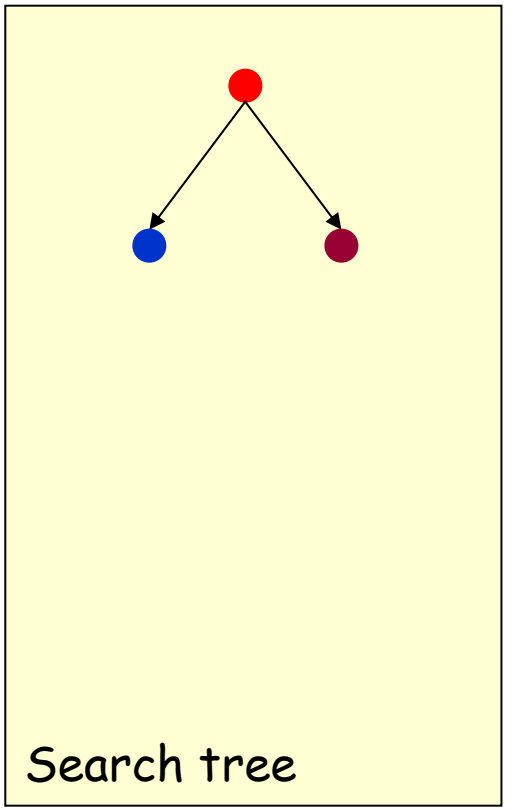
Ricerca nello spazio degli stati (disegno di J.C. Latombe)



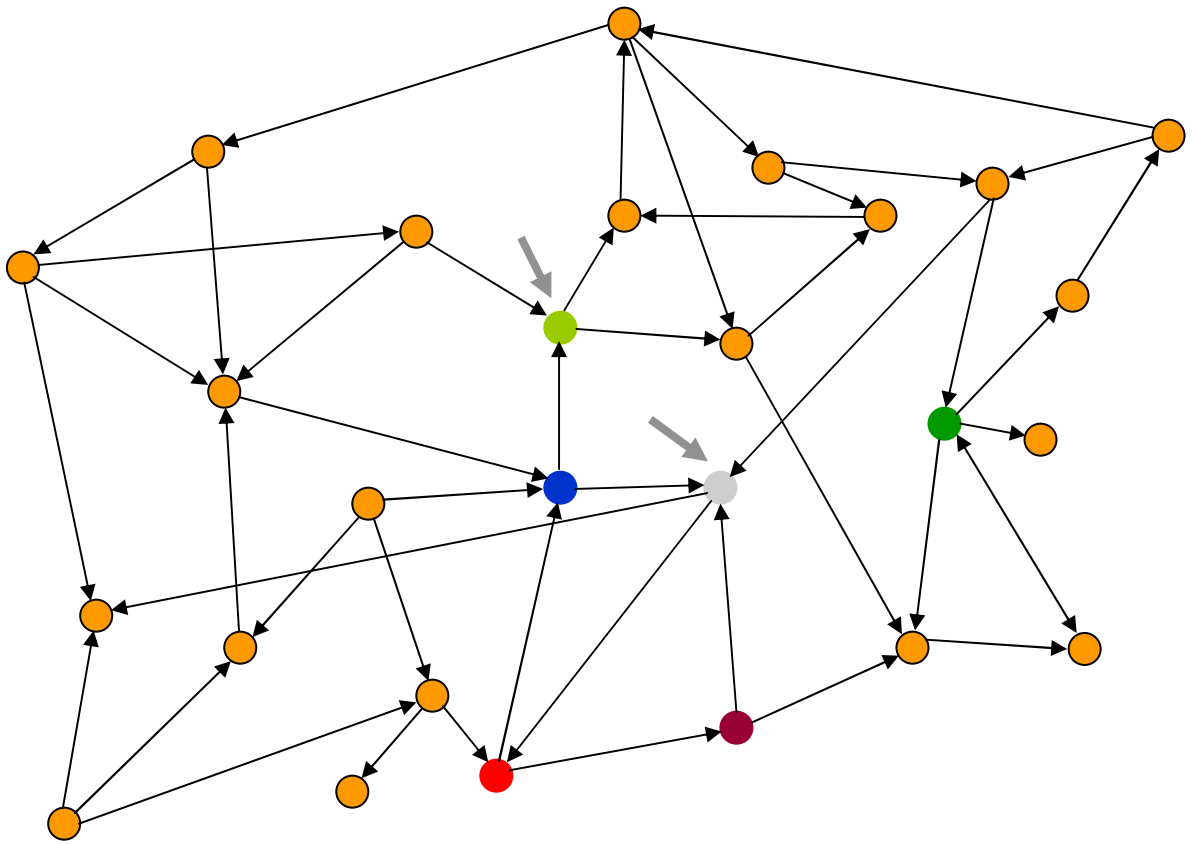


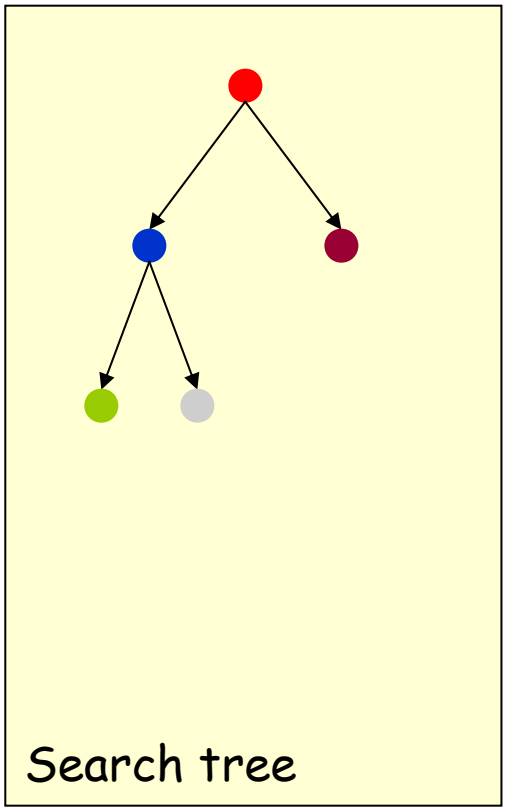
Ricerca nello spazio degli stati (disegno di J.C. Latombe)



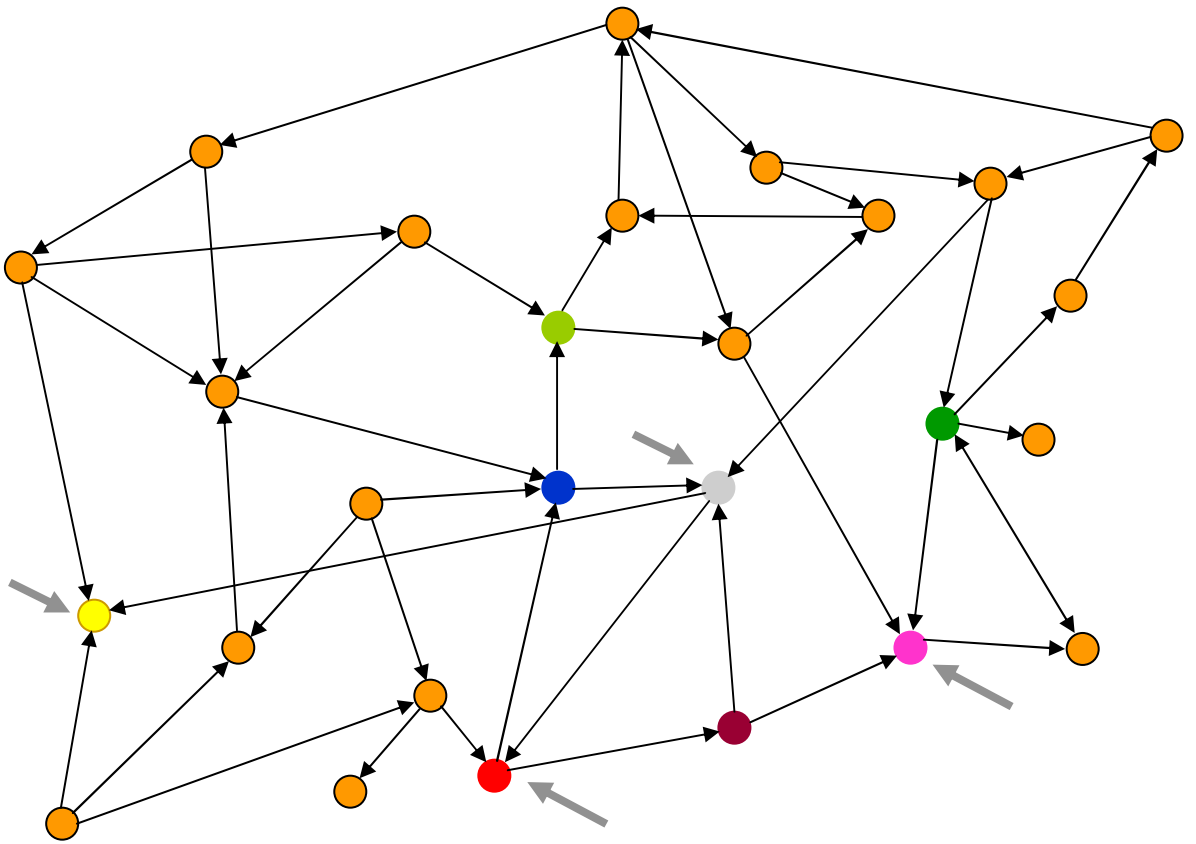


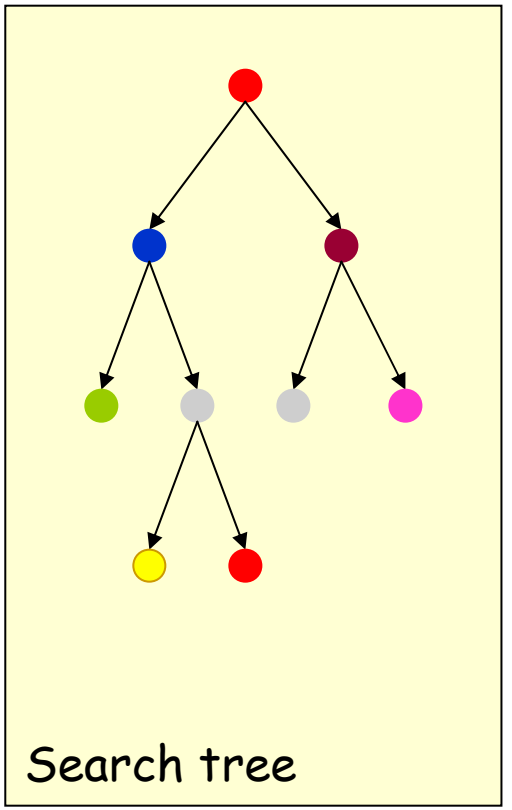
Ricerca nello spazio degli stati (disegno di J.C. Latombe)



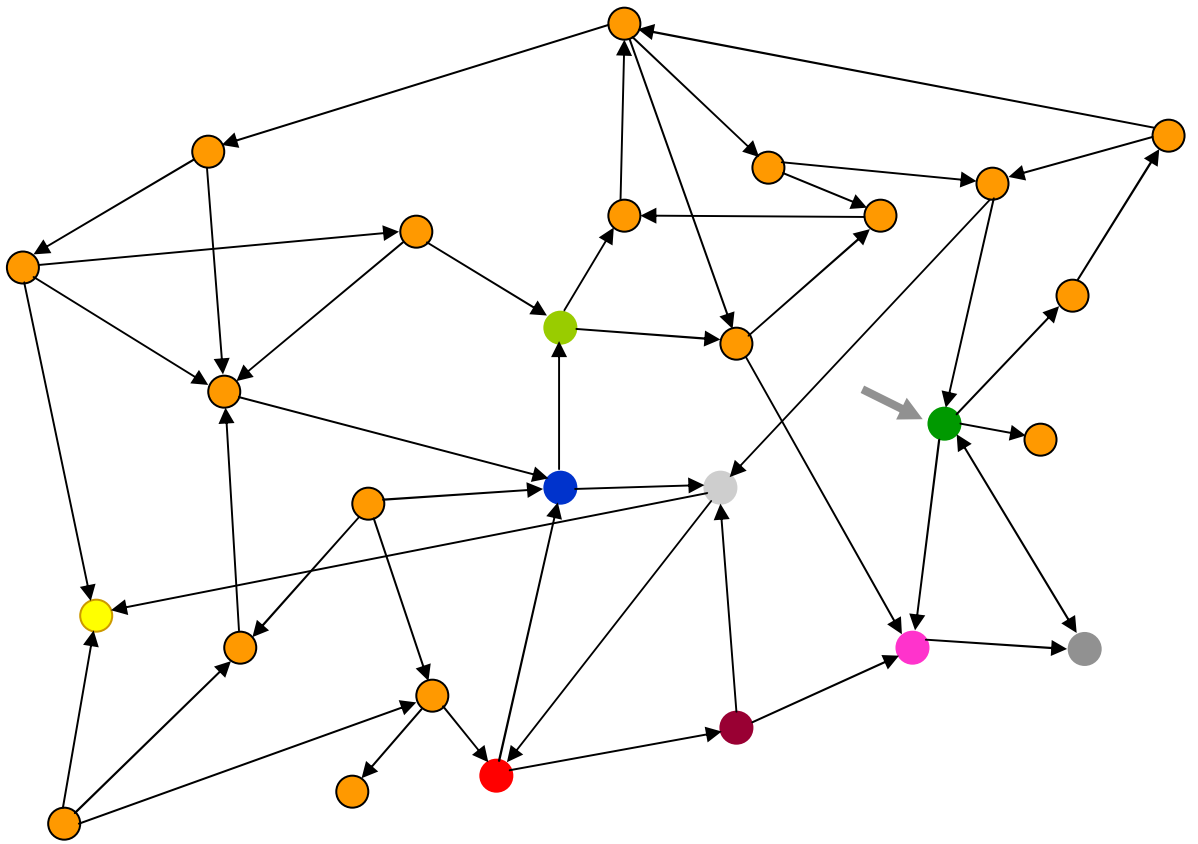


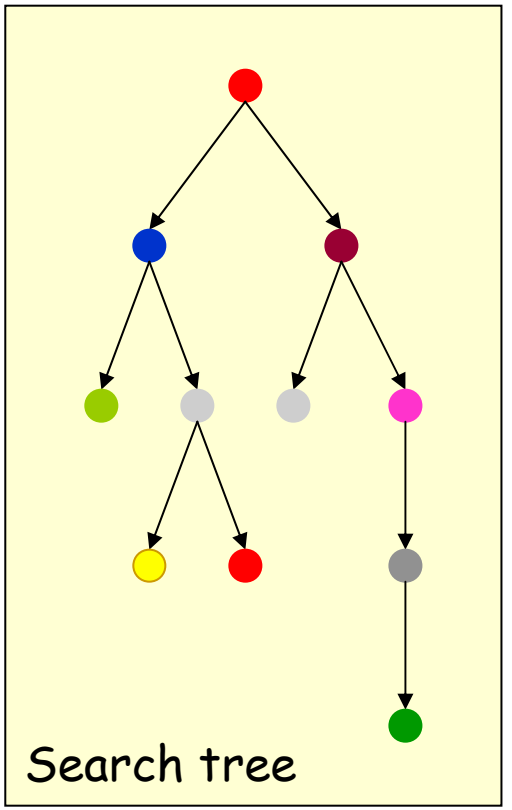
Ricerca nello spazio degli stati (disegno di J.C. Latombe)



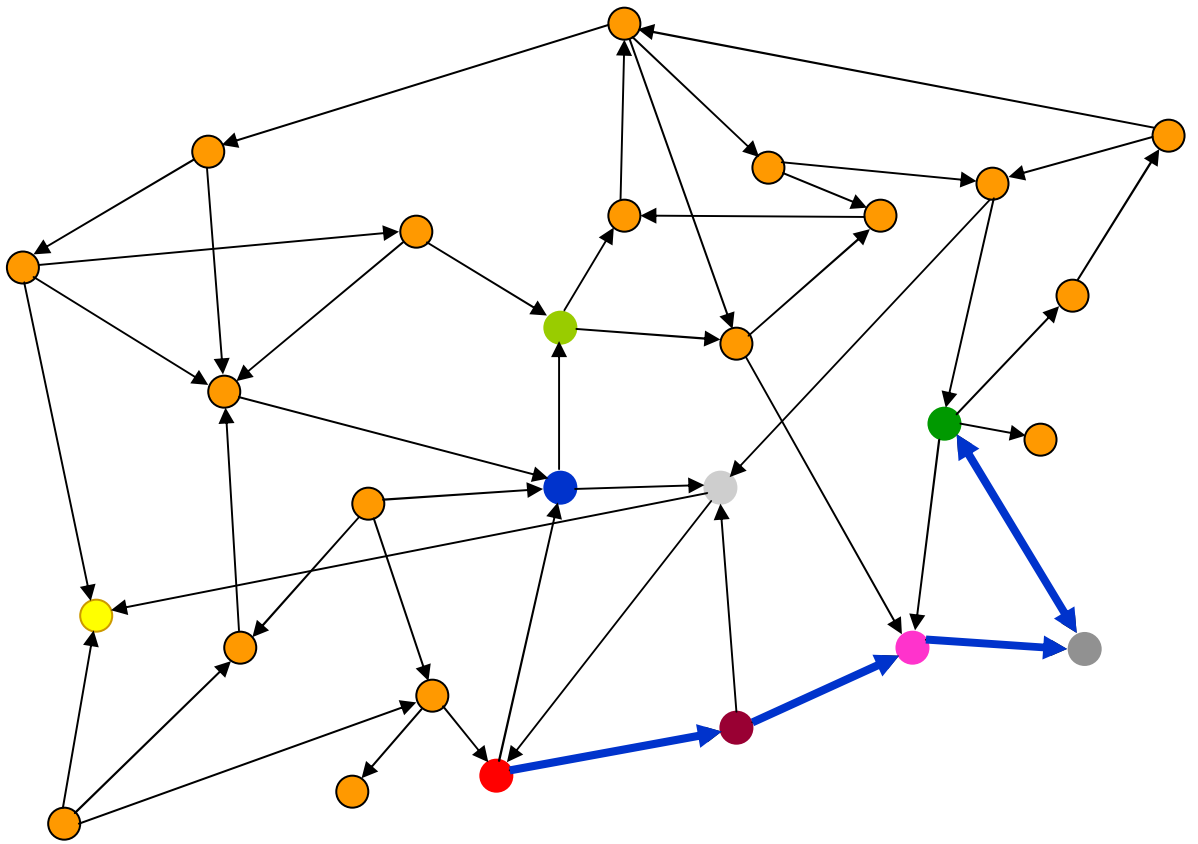


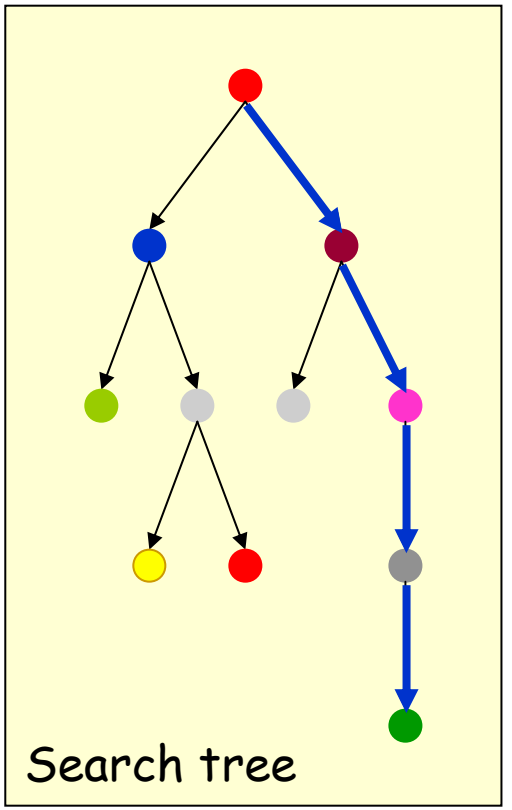
Ricerca nello spazio degli stati (disegno di J.C. Latombe)





Ricerca nello spazio degli stati (disegno di J.C. Latombe)





Ricerca nello spazio degli stati

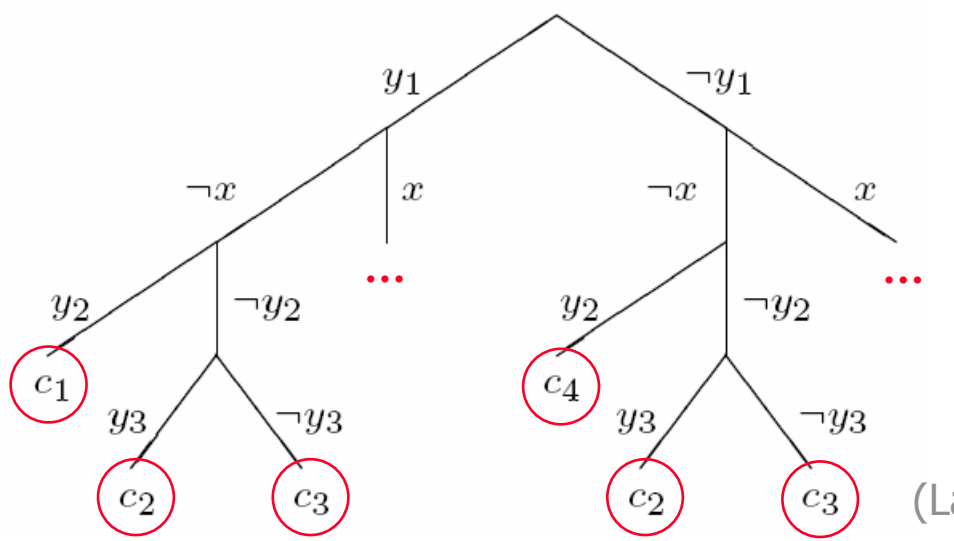
- **Stato**: s
  - Può essere una situazione o uno stato di informazione
- **Spazio degli stati**: S
  - Insieme di tutti gli stati possibili
- **Successori** di ciascuno stato: $Successors(s)$
  - $Successors(.) : S \rightarrow 2^S$

- Algoritmo di ricerca (algoritmo base)
  - Generazione di un **albero**
    1) A partire da uno stato iniziale s_c
    2) Si crea un nuovo nodo scegliendo uno stato s nell'insieme $Successors(s_c)$
    3) Torna a 2)
  - Terminazione (due possibilità)
    - Raggiunto lo stato goal (basta una soluzione)
    - Completa esplorazione dello spazio degli stati (si cercano tutte le soluzioni)

Un problema tipico: SAT

- Mostrare che una formula proposizionale è soddisfacibile
  - Stato s: assegnazioni di valore di verità a variabili proposizionali (s_0 è lo stato nullo: nessuna assegnazione)
  - $Successors(s)$: possibili assegnazioni ad una variabile (nell'esempio si usa l'ordine di apparizione)
  - Vincoli: le clausole c_1, c_2, c_3, c_4 (devono essere tutte soddisfatte)

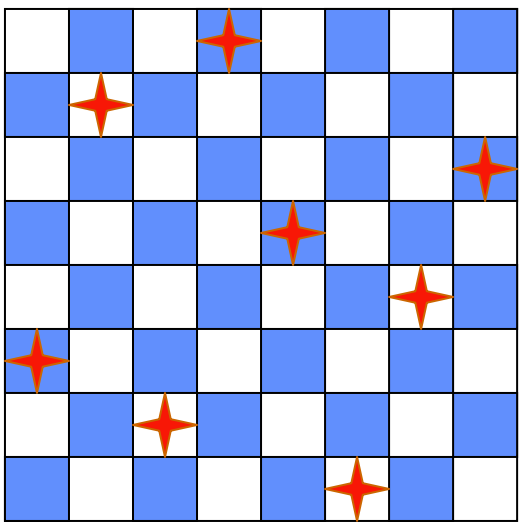
$$\big(\underbrace{(\neg y_1 \vee x \vee \neg y_2)}_{c_1} \wedge \underbrace{(y_2 \vee \neg y_3)}_{c_2} \wedge \underbrace{(y_2 \vee y_3)}_{c_3} \wedge \underbrace{(y_1 \vee \neg x \vee \neg y_2)}_{c_4}\big)$$



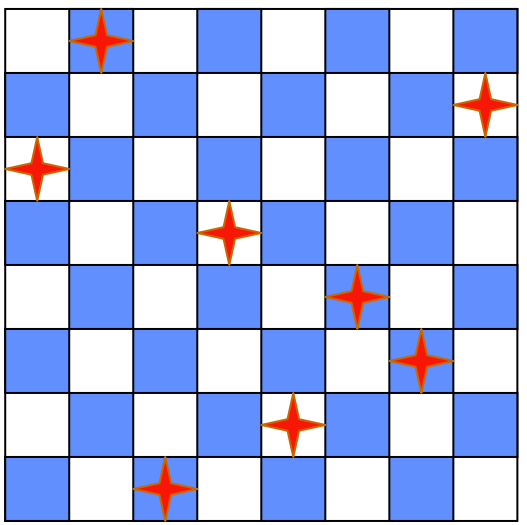
(La formula non è soddisfacibile)

Codifica dello spazio degli stati

- Problema: le otto regine
  - Piazzare 8 regine su una scacchiera 8 x 8 in modo che nessuna possa attaccare l'altra

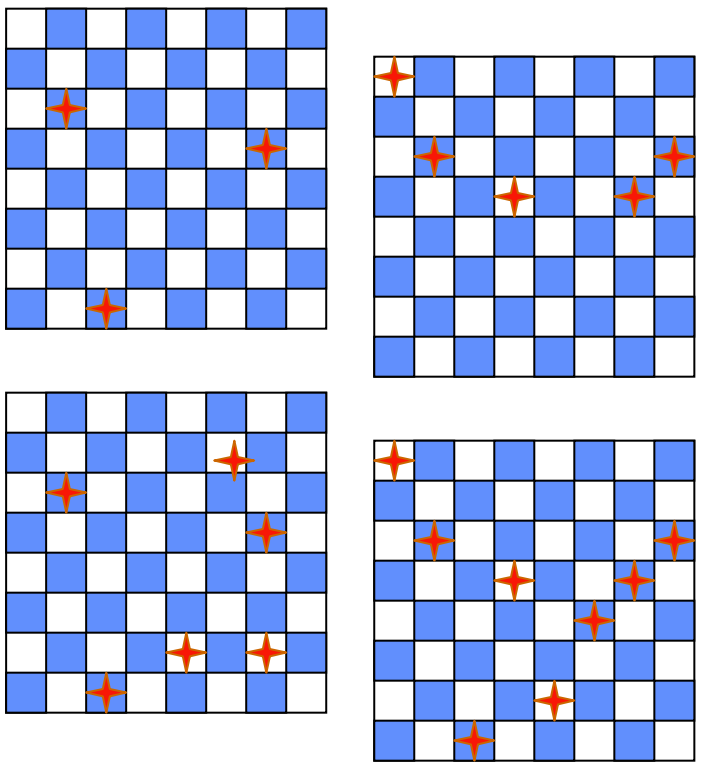


Una soluzione



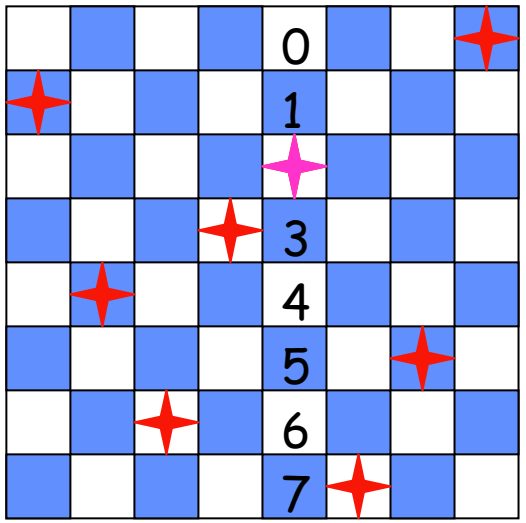
Una non-soluzione

Codifica dello spazio degli stati (2)



- Prima ipotesi di rappresentazione
  - Stati: tutte le disposizioni di 0, 1, 2, 3, 4, 5, 6, 7, 8 regine sulla scacchiera
  - Stato iniziale: 0 regine sulla scacchiera
  - Successors(s): tutti gli stati che si possono ottenere da s aggiungendo una regina
  - Goal (non unico): stato in cui 8 regine non possono attaccarsi
- Complessità:
  - Circa 3 x 10^{14} stati …

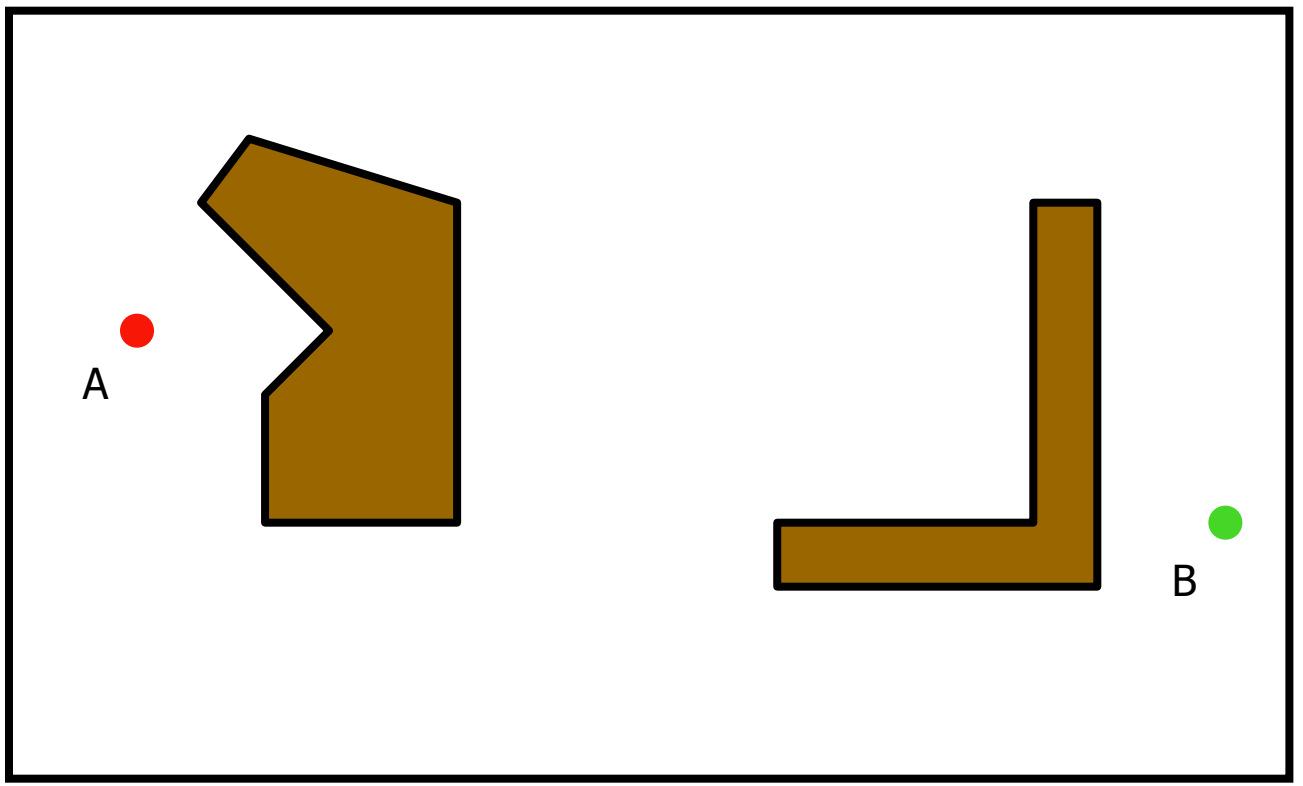
Codifica dello spazio degli stati (2)



- Seconda ipotesi di rappresentazione
  - (Due regine nella stessa colonna possono attaccarsi)
  - Stati: tutte le disposizioni di 8 regine in cui ciascuna regina occupa una colonna
  - Stato iniziale: qualsiasi
  - $Successors(s)$: tutti gli stati che si possono ottenere da s spostando una regina nella propria colonna
- Complessità:
  - Circa 1.6 x 10^7 stati
  - La prima ipotesi era generale (andava bene per il problema delle n regine, con n $\leq$ 8)
  - La seconda ipotesi sfrutta meglio i requisiti del problema

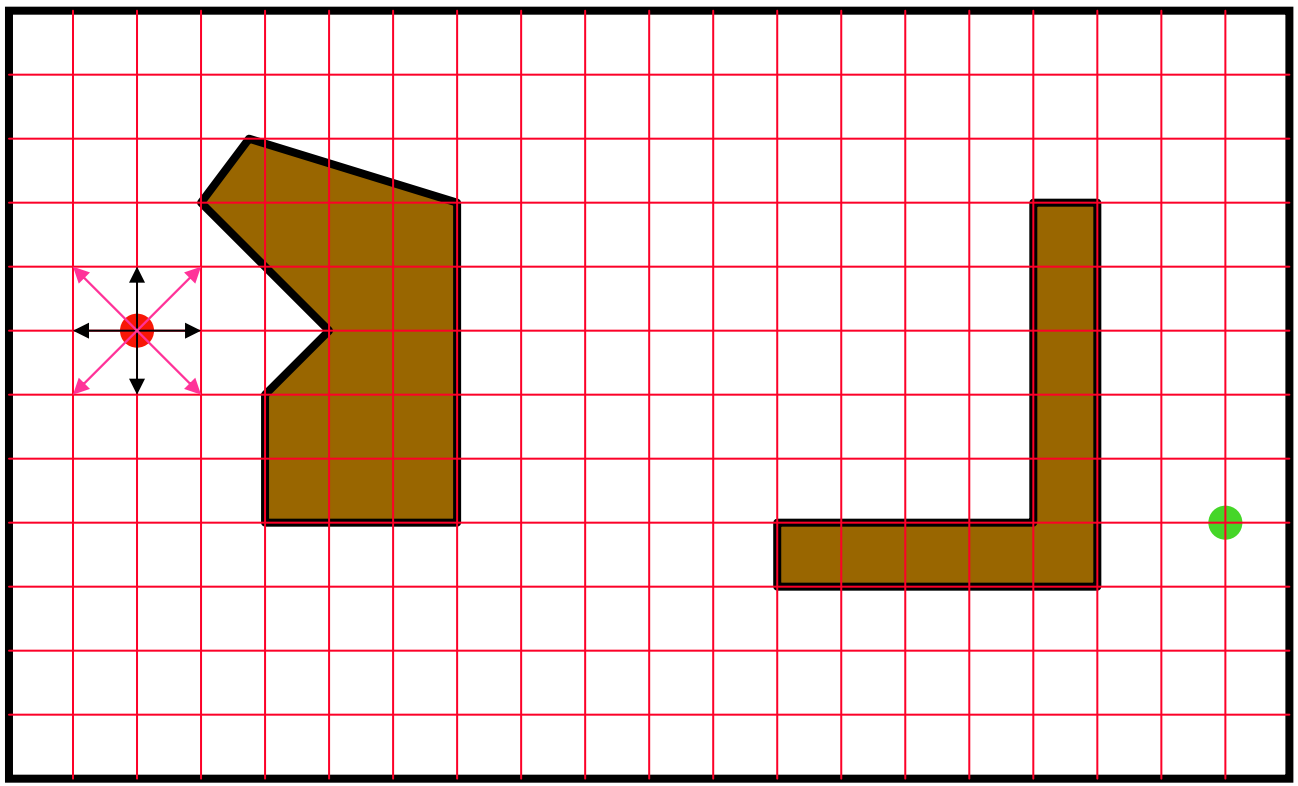
Pianificazione come ricerca (disegno di J.C. Latombe)

– Un robot deve andare da A a B, in una stanza con ostacoli



Pianificazione come ricerca (disegno di J.C. Latombe)

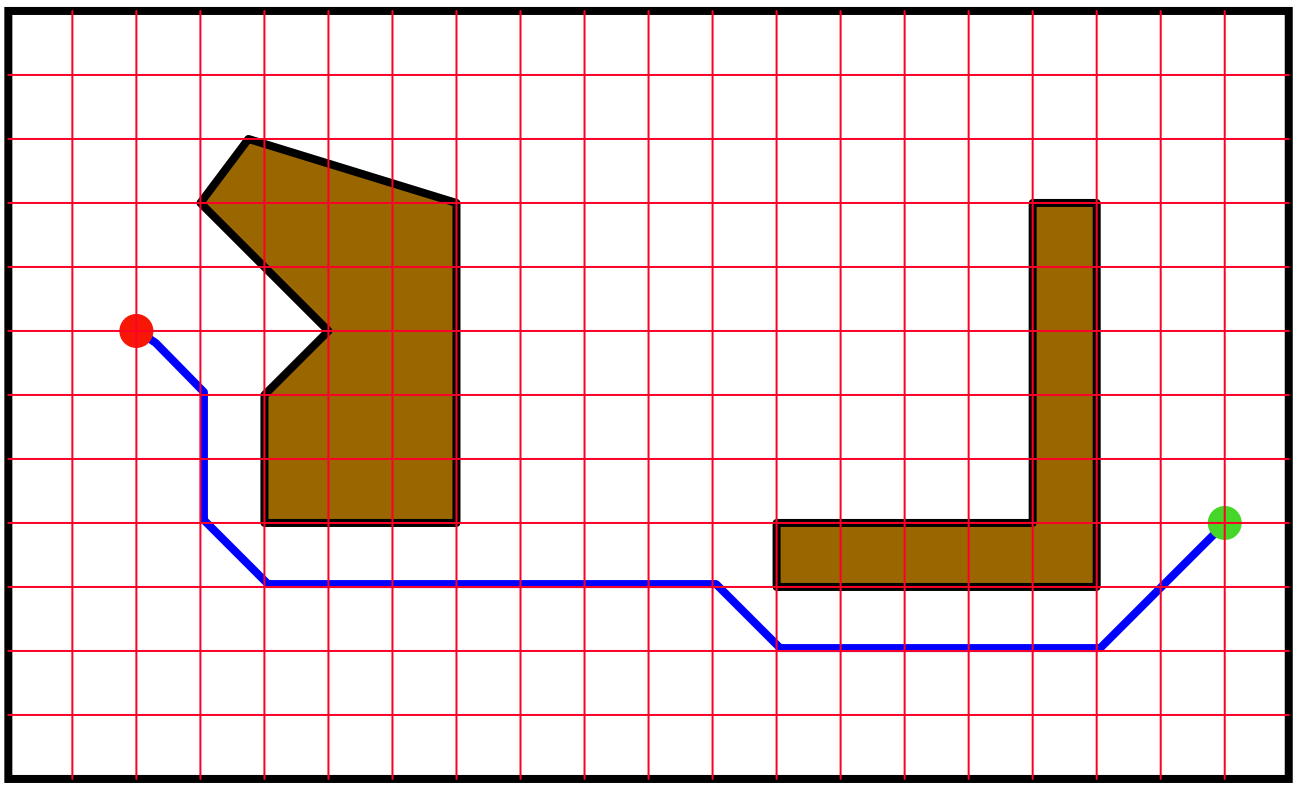
– Stati: punti di una griglia discreta 19 x 11 (~ 2 x 10^2)



– Costi: movimento orizzontale o verticale 1, diagonale $\sqrt{2}$

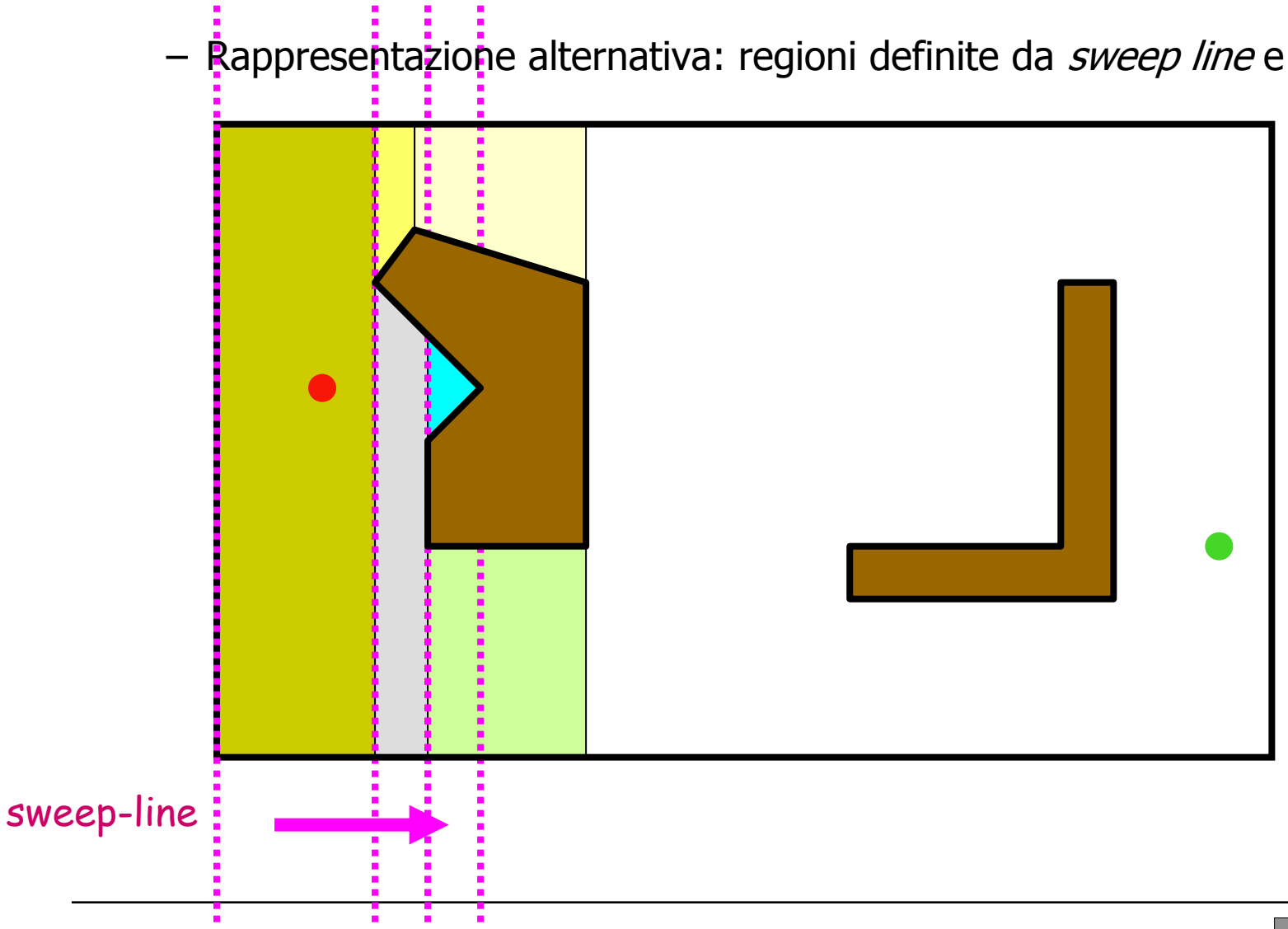
Pianificazione come ricerca (disegno di J.C. Latombe)

- Soluzione ottimale (data la discretizzazione)



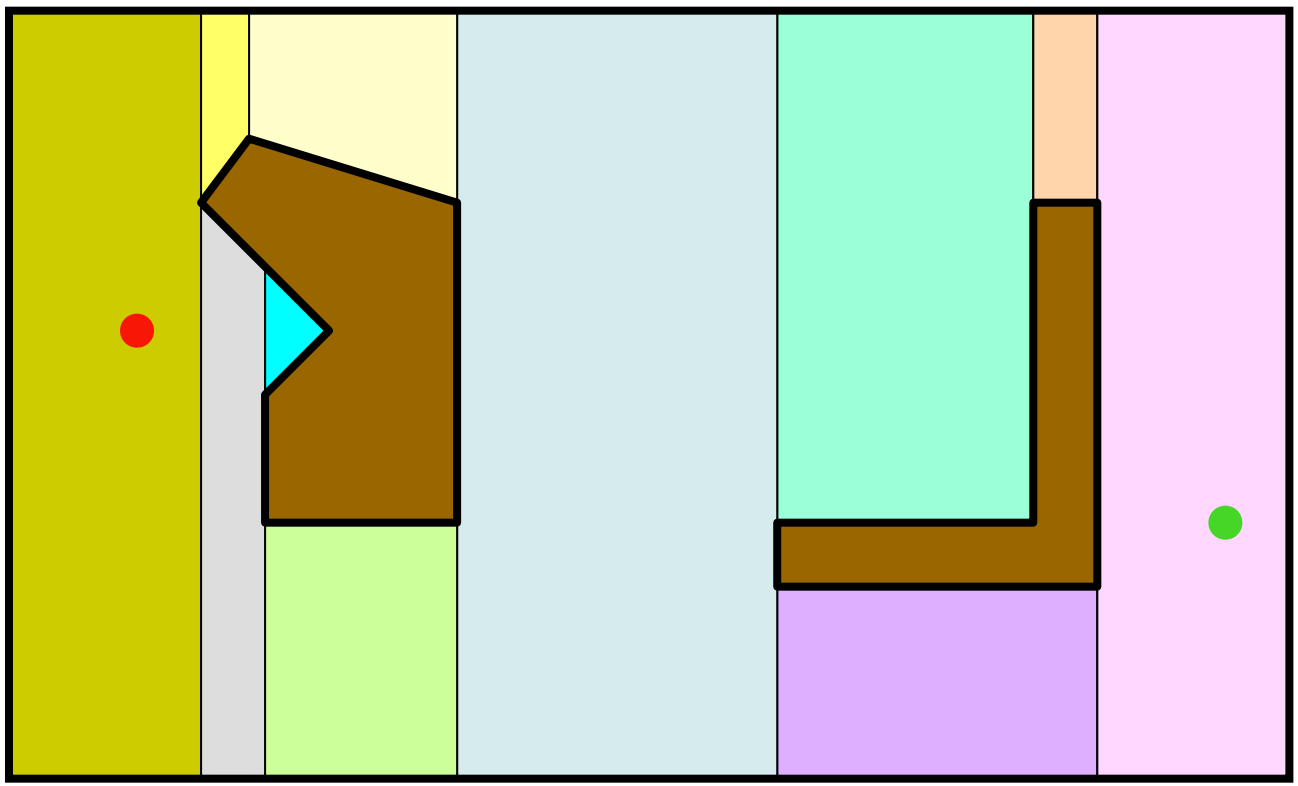
Pianificazione come ricerca (disegno di J.C. Latombe)

– Rappresentazione alternativa: regioni definite da *sweep line* e singolarità



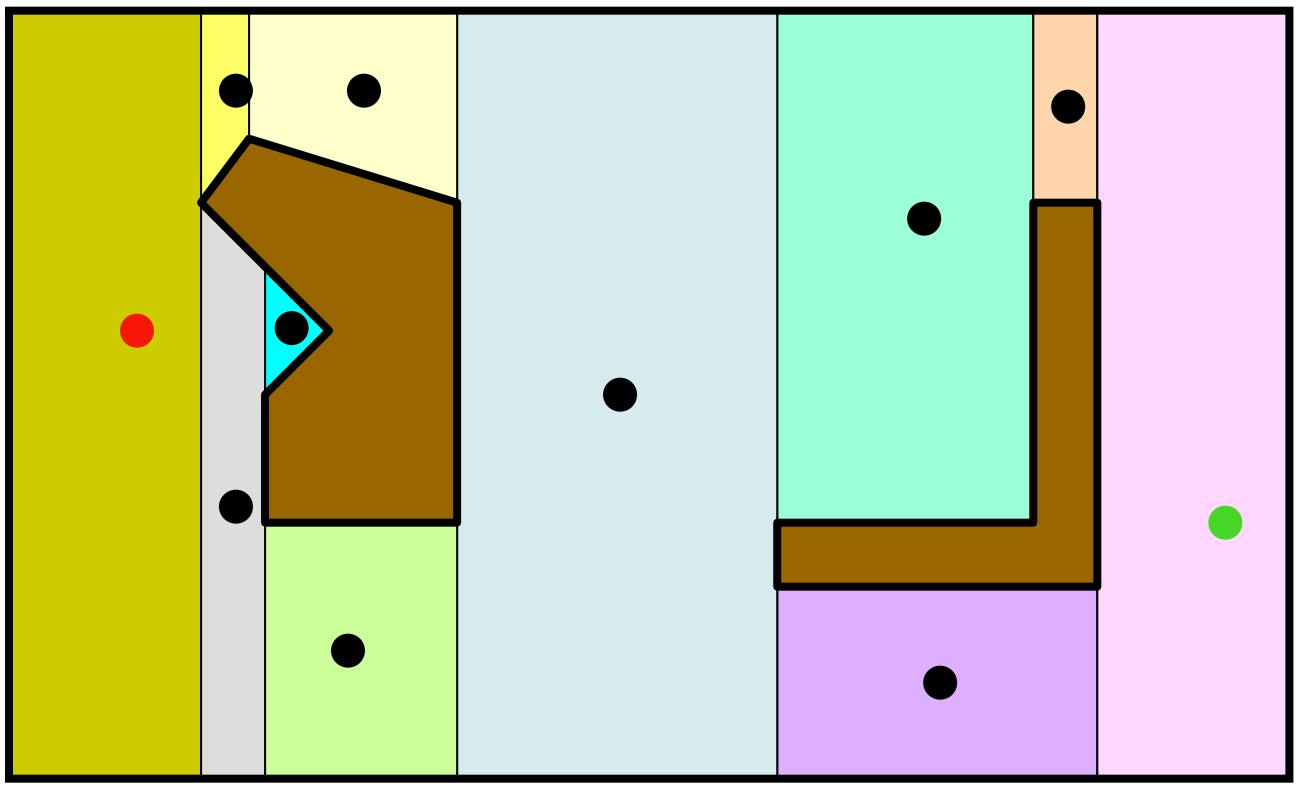
Pianificazione come ricerca (disegno di J.C. Latombe)

– Regioni



Pianificazione come ricerca (disegno di J.C. Latombe)

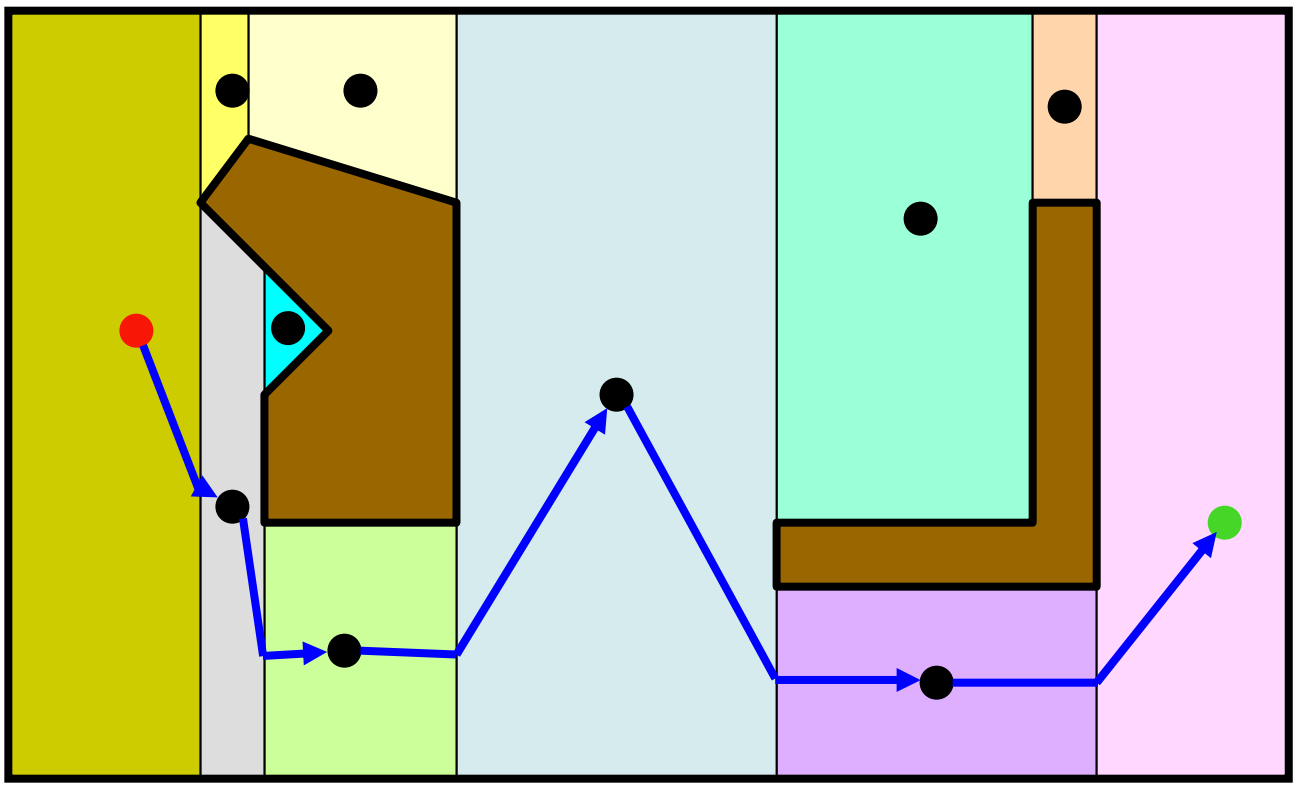
– Stati: il centroide di ogni regione (11 stati)



– Costi: distanza tra i centroidi

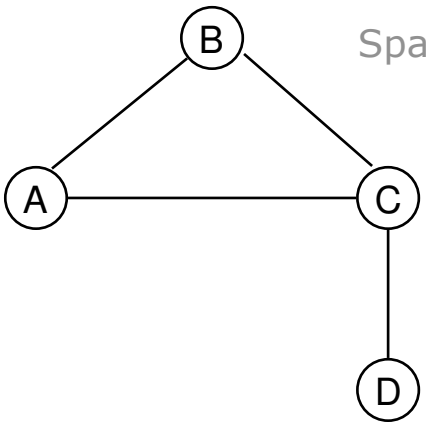
Pianificazione come ricerca (disegno di J.C. Latombe)

– Soluzione (approssimata)

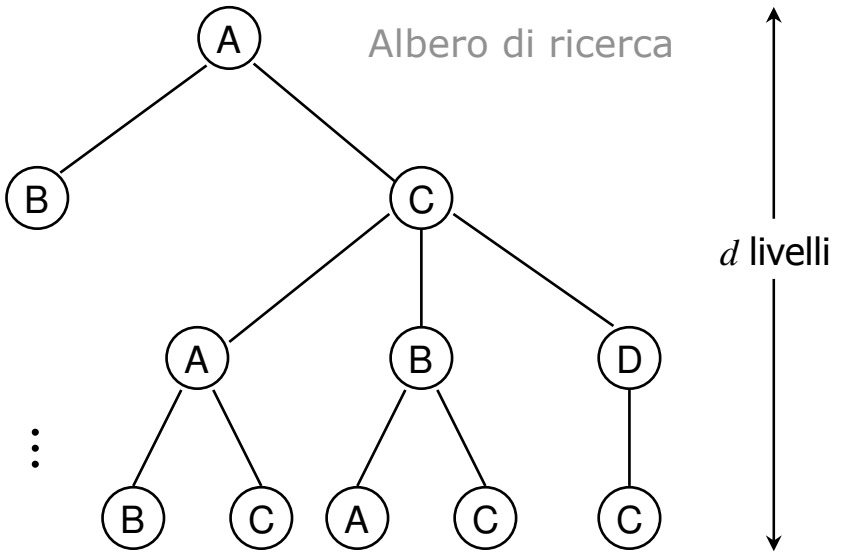


– Occorre un'ulteriore passo di affinamento, per accorciare il percorso

Definizioni

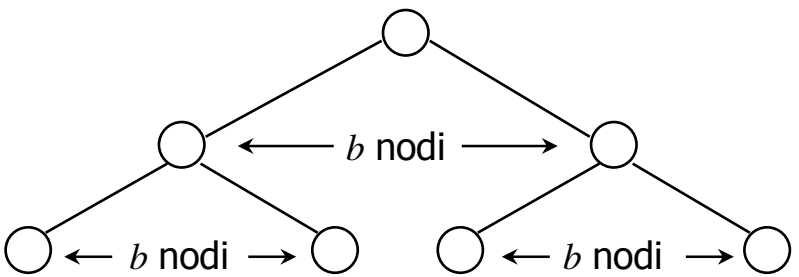


Spazio degli stati



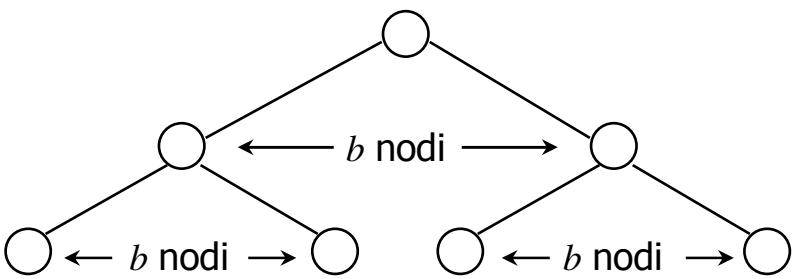
Albero di ricerca

- Nodo (dell'albero)
- Nodo radice
- Nodo genitore
- Profondità (d):
  numero di livelli
- Branching factor (b):
  fattore di biforcazione degli archi



Strategie di ricerca

- Parametri di valutazione
  - Completezza: è certo che si trovi una soluzione? (o tutte le soluzioni, se necessario)
  - Ottimalità: in presenza di diverse soluzioni, verrà trovata la soluzione migliore?
  - Time complexity: quanto tempo sarà necessario?
  - Space complexity: quanta memoria sarà necessaria?

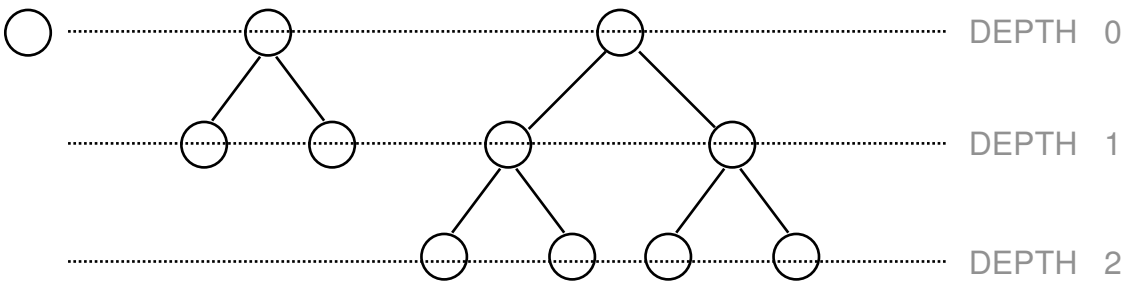


Numero di nodi di un albero con branching factor costante:

$$n = 1 + b + b^2 + b^3 + \ldots + b^d$$

Strategie di ricerca (2)

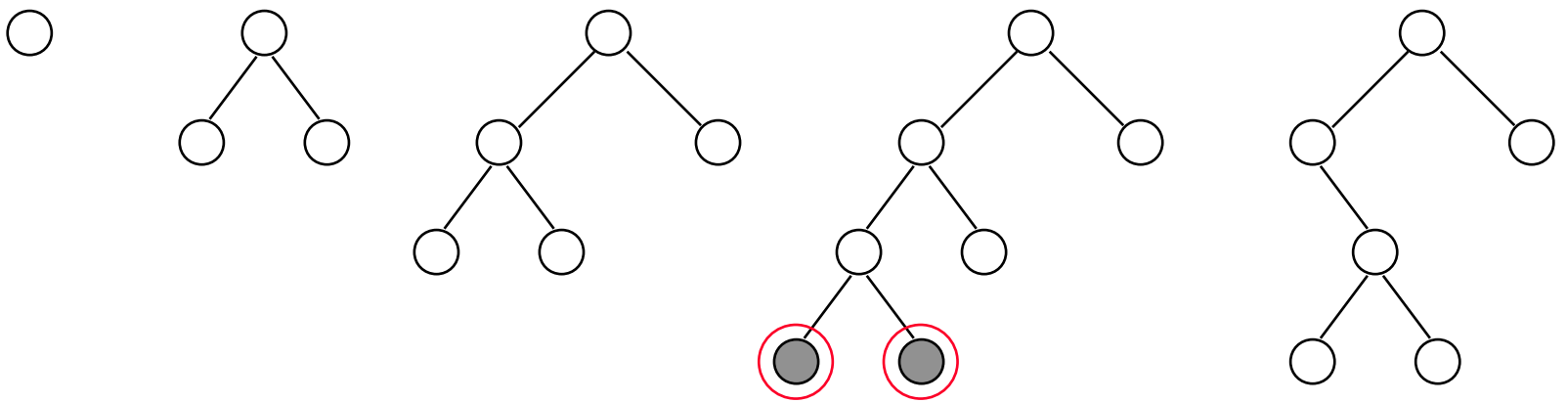
- **Strategia breadth-first** (ricerca in ampiezza)
  - Si espandono tutti i nodi di un livello prima di passare al successivo



  - Completa
  - Ottimale
  - Time complexity: $O(b^d)$
  - Space complexity: $O(b^d)$

Strategie di ricerca (3)

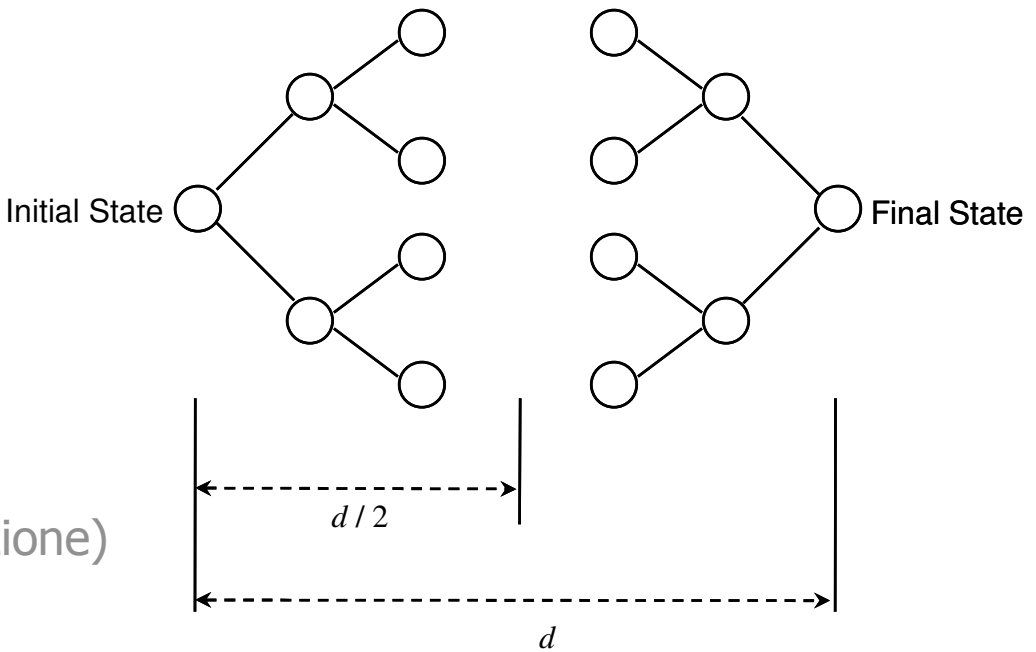
- **Strategia depth-first** (ricerca in profondità)
  - Si espandono i nodi privilegiando la profondità
  - Si ritrattano i rami che non hanno successo



  - Completa (se si cerca una sola soluzione)
  - Non ottimale
  - Time complexity: $O(b^d)$
  - Space complexity: $O(bd)$ (si espande un solo ramo alla volta)
  - Euristica: si trova prima una soluzione (qualsiasi)

Strategie di ricerca (4)

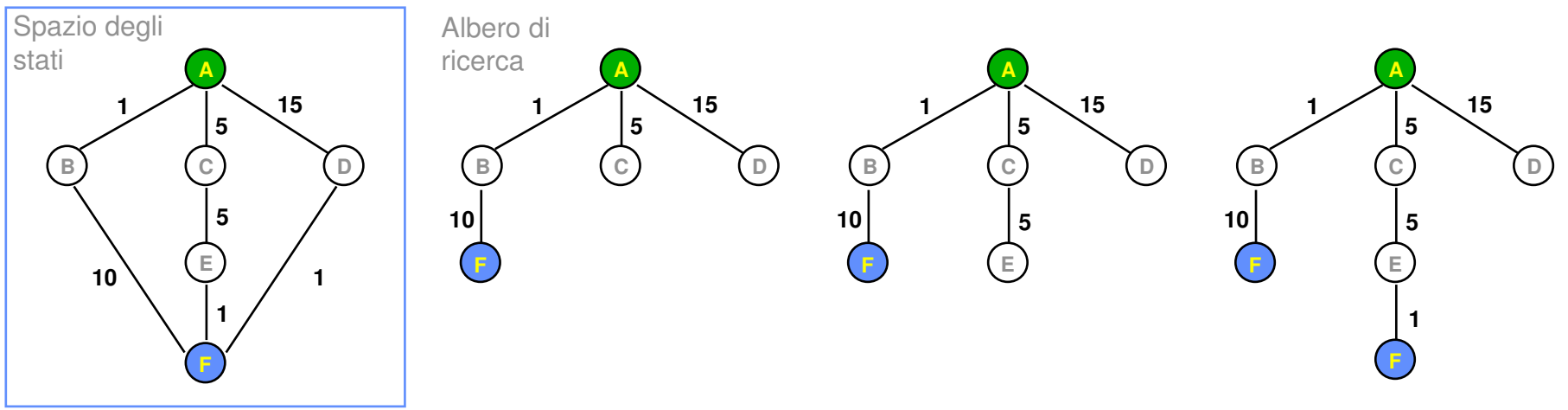
- **Strategia bidirezionale**
  - Si espandono i nodi nelle due direzioni, *top-down* e *bottom-up*



  - Completa
    (se si cerca una sola soluzione)
  - Non ottimale
  - Time complexity: $O(b^{d/2})$
  - Space complexity: $O(b^{d/2})$
  - Euristica: 'soffre meno' il *branching factor* …

Strategie di ricerca (5)

- **Strategia a costo uniforme** (simile a breadth-first)
  - Si privilegiano i cammini a costo complessivo più basso



  - Completa
  - Ottimale
  - Time complexity: $O(b^d)$
  - Space complexity: $O(b^d)$
  - Euristica: si trovano prima i cammini meno costosi