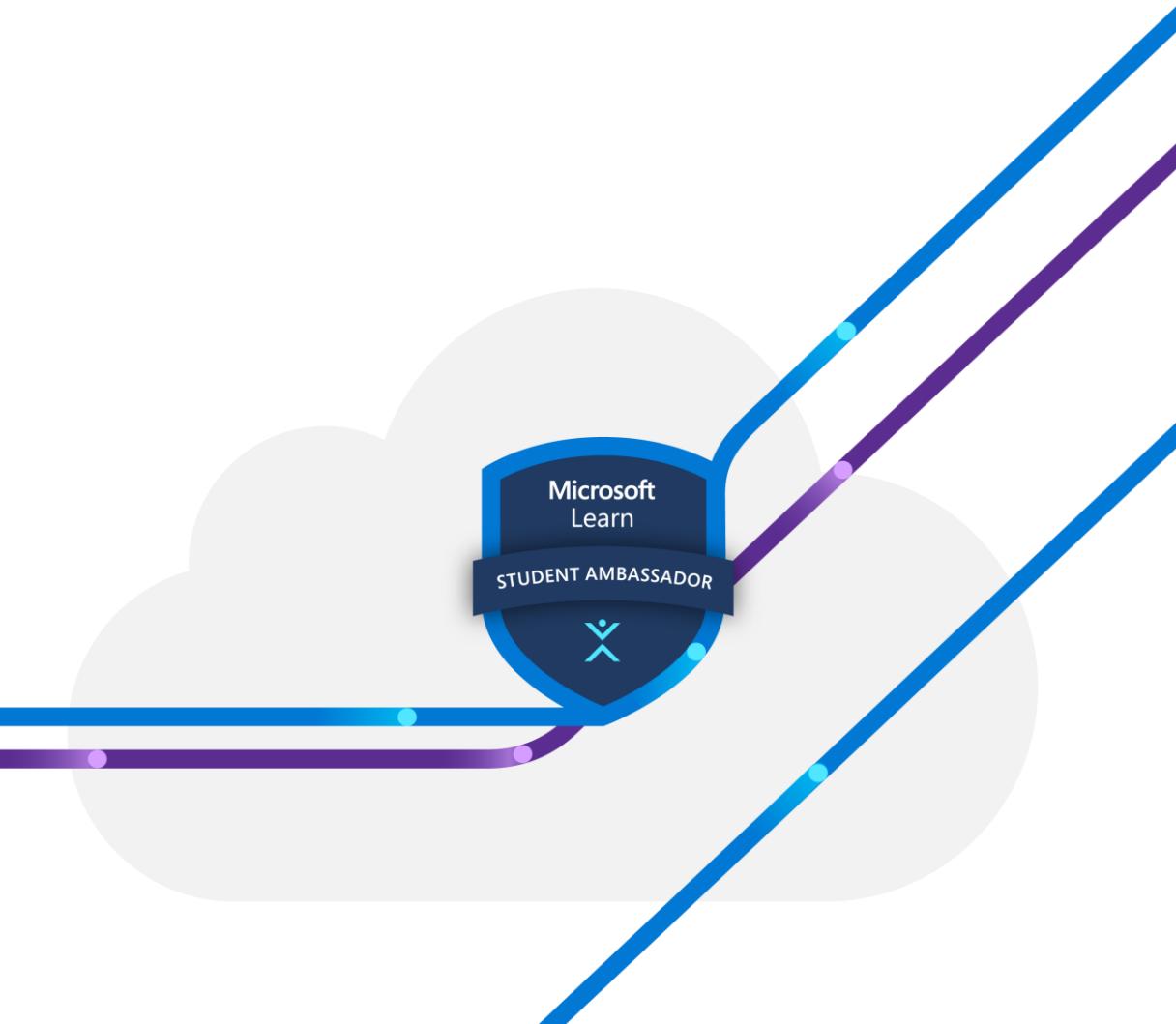


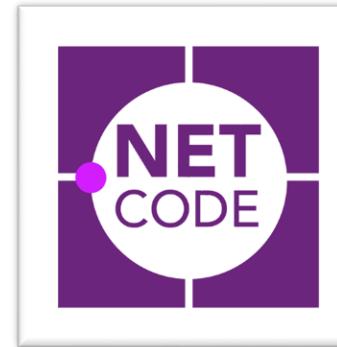
# Search Path & Dijkstra

Mario Cuomo  
27.10.2021



# MARIO CUOMO

-  [mariocuomo.github.io](https://mariocuomo.github.io)
-  [linkedin/in/mariocuomo](https://linkedin/in/mariocuomo)
-  [@mariocuomo.exe](https://instagram.com/@mariocuomo.exe)
-  [@mariocuomoEXE](https://twitter.com/@mariocuomoEXE)



# Intelligenza Artificiale



# Intelligenza Artificiale

L'Intelligenza Artificiale studia come far eseguire al computer compiti che per l'uomo richiederebbero intelligenza.



# Intelligenza Artificiale

L'Intelligenza Artificiale studia come far eseguire al computer compiti che per l'uomo richiederebbero intelligenza.

Nasce nel 1950.

Alan Turing - *Computing Machinery and Intelligence*





# Search Path

Ricerca di cammino da un punto A a un punto B



# Search Path

Ricerca di cammino da un punto A a un punto B

- Definizione dell'obiettivo



# Search Path

Ricerca di cammino da un punto A a un punto B

- Definizione dell'obiettivo
- Formulare il problema



# Search Path

Ricerca di cammino da un punto A a un punto B

- Definizione dell'obiettivo
- Formulare il problema
- Trovare una sequenza di passi



# Search Path

Ricerca di cammino da un punto A a un punto B

- Definizione dell'obiettivo
  - Identificare un insieme di stati
- Formulare il problema
- Trovare una sequenza di passi



# Search Path

Ricerca di cammino da un punto A a un punto B

- Definizione dell'obiettivo
  - Identificare un insieme di stati
- Formulare il problema
  - Identificare un insieme di azioni
- Trovare una sequenza di passi



# Esempio (classico in letteratura)

Mi trovo ad Aarad e devo raggiungere Bucarest.



# Esempio (classico in letteratura)

Mi trovo ad Aarad e devo raggiungere Bucarest.

Le città intermedie sono gli stati.



# Esempio (classico in letteratura)

Mi trovo ad Aarad e devo raggiungere Bucarest.

Le città intermedie sono gli **stati**.

Gli spostamenti tra le città sono le **azioni**.



# Esempio (classico in letteratura)

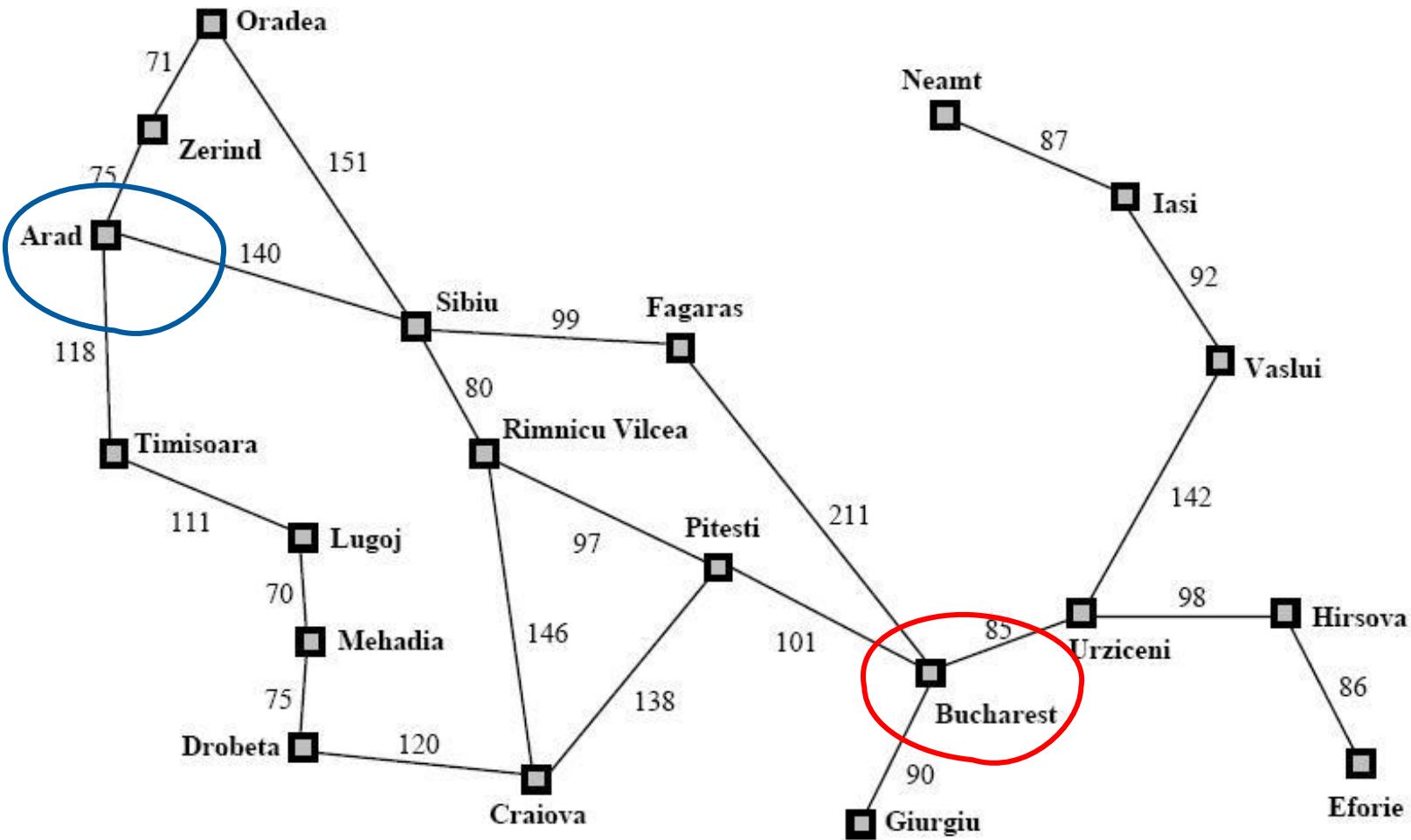
Mi trovo ad Aarad e devo raggiungere Bucarest.

Le città intermedie sono gli stati.

Gli spostamenti tra le città sono le azioni.

La sequenza di città visitate è la soluzione.





# Rappresentazione

Per rappresentare la sequenza delle operazioni da effettuare si utilizza un albero.



# Rappresentazione

Per rappresentare la sequenza delle operazioni da effettuare si utilizza un albero.

- Un nodo rappresenta uno stato *ammissibile* dello *spazio degli stati*

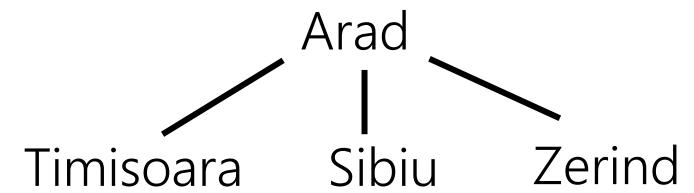
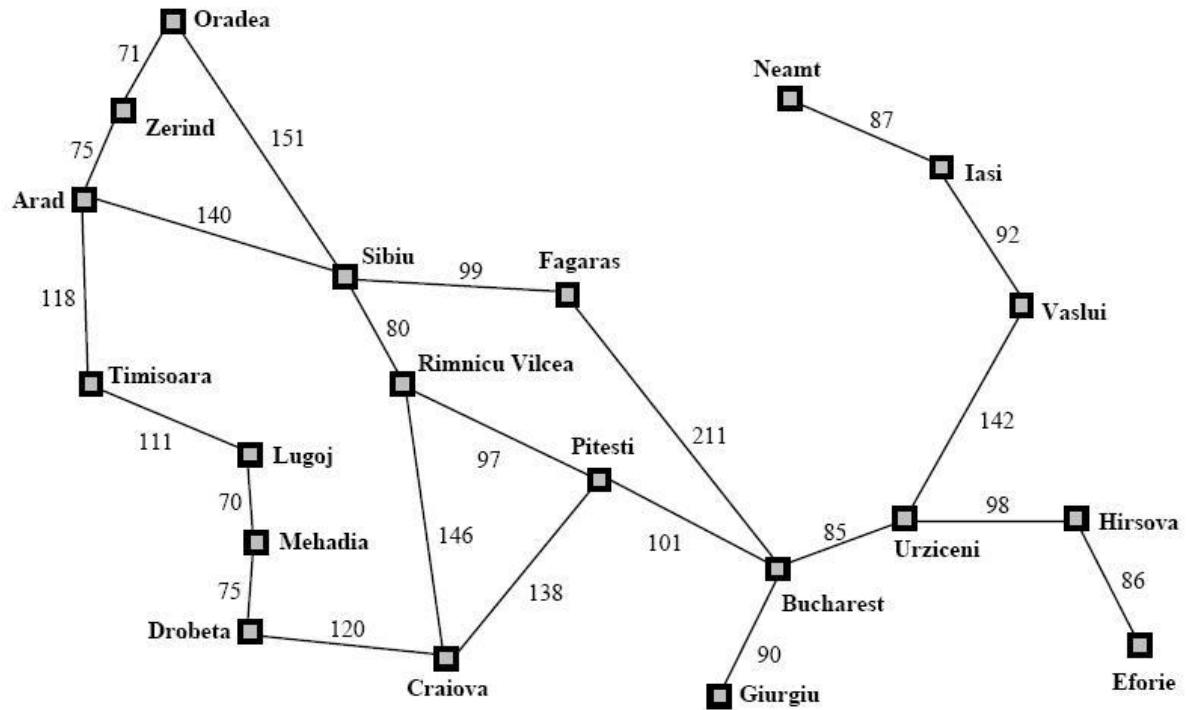


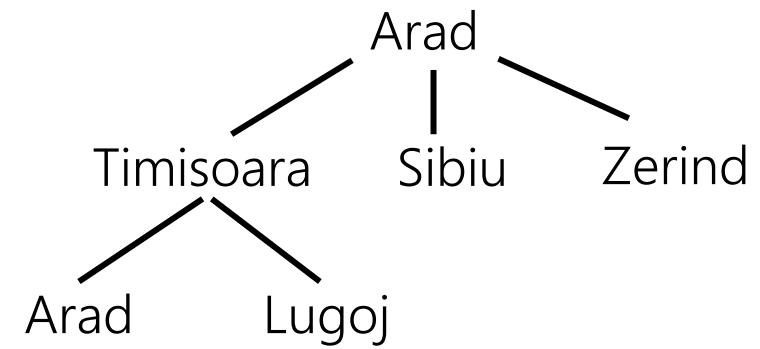
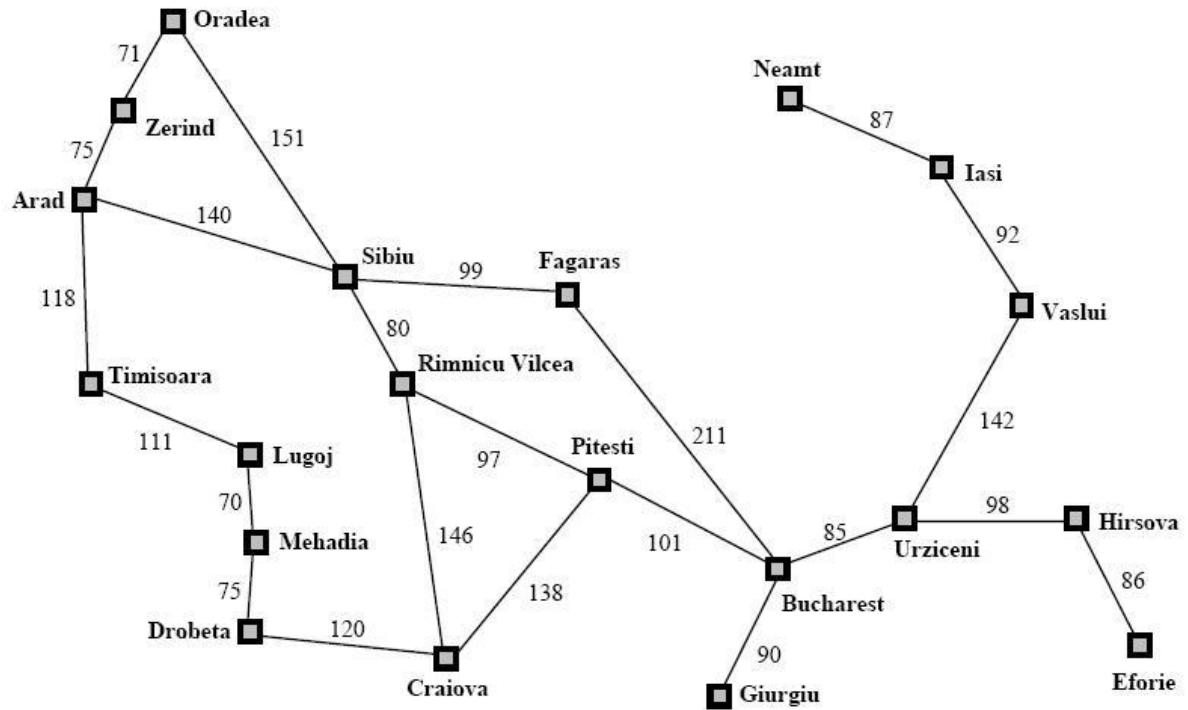
# Rappresentazione

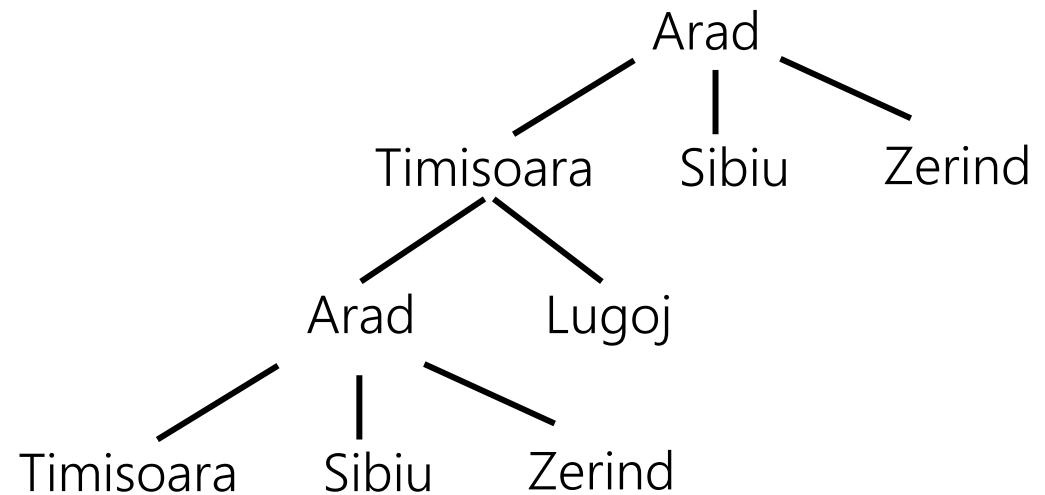
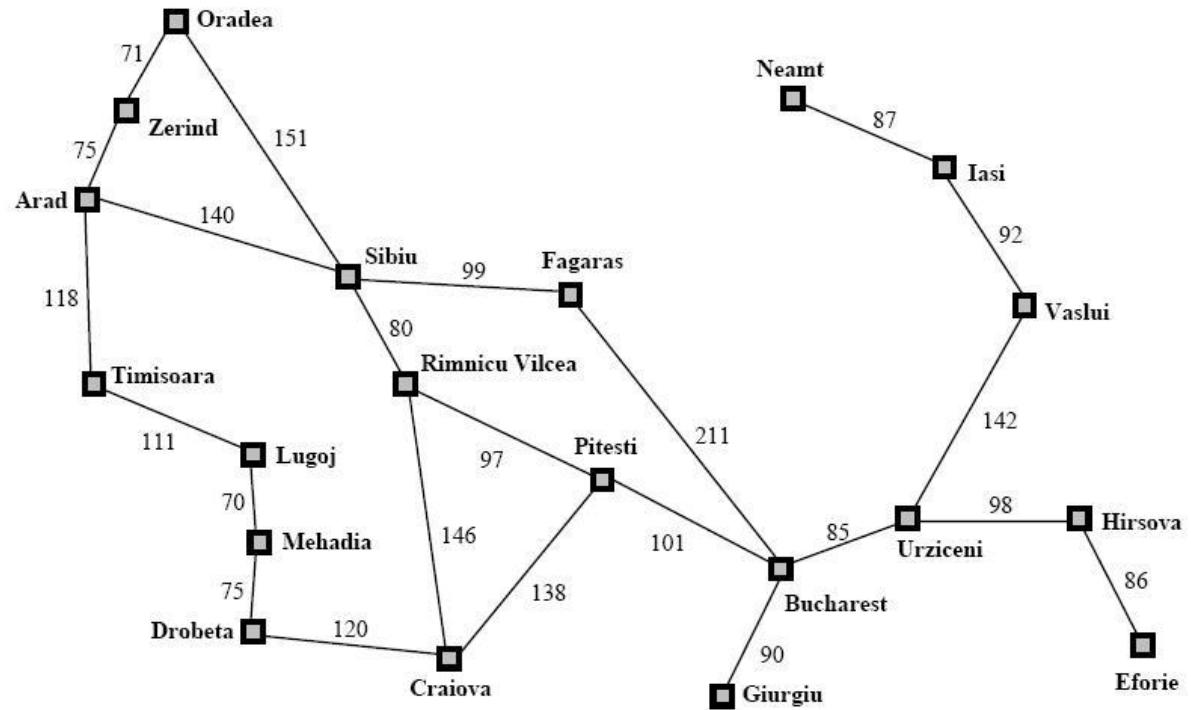
Per rappresentare la sequenza delle operazioni da effettuare si utilizza un albero.

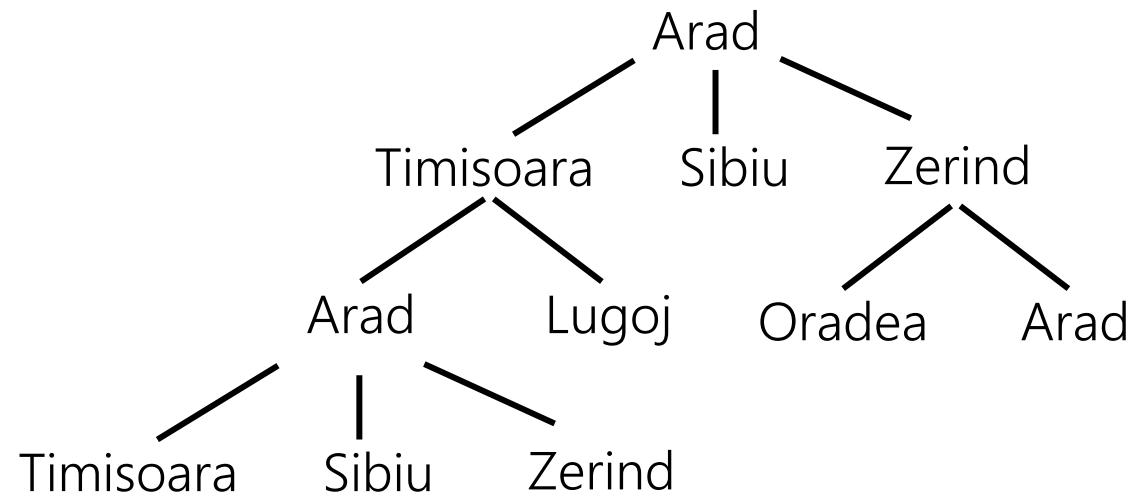
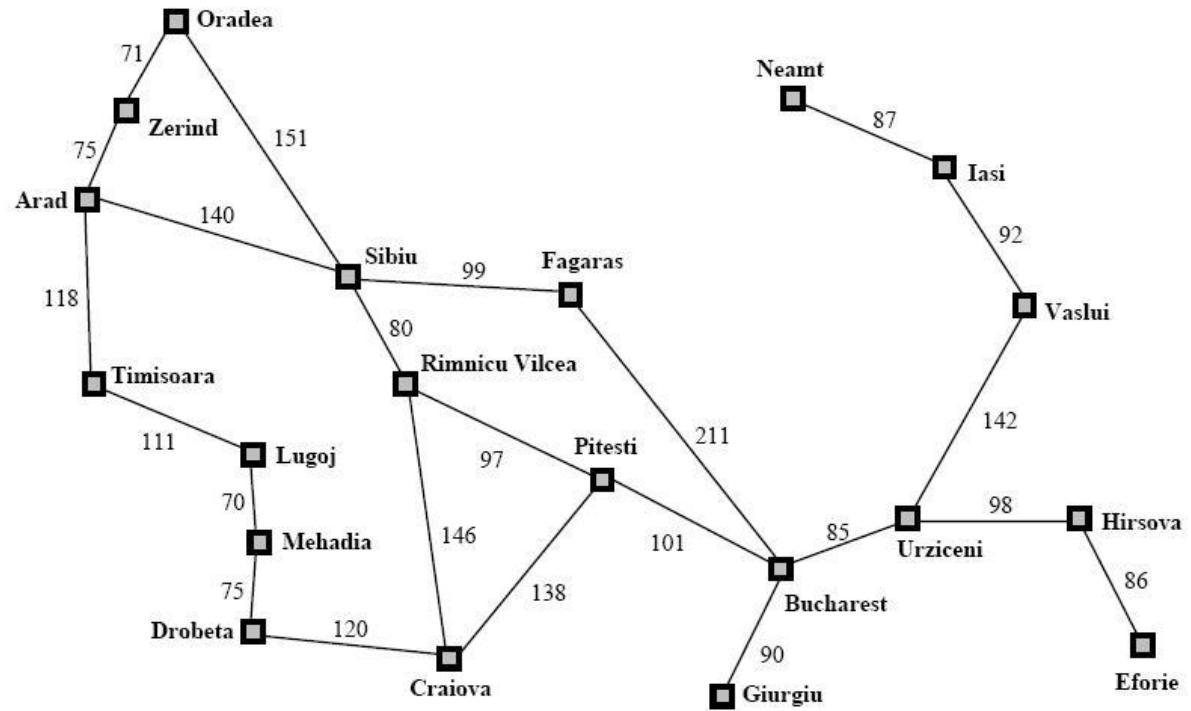
- Un nodo rappresenta uno stato *ammissibile* dello *spazio degli stati*
- Un arco rappresenta una delle *possibili azioni* che è possibile effettuare dallo stato corrente











# Obiettivo

Trovare la strada ‘migliore’.



# Obiettivo

Trovare la strada ‘migliore’.

Caratteristiche salvabili in un nodo

$\langle stato, genitore, operatore, profondità, costo parziale, \dots \rangle$



# Algoritmo Tree-search

```
function tree-search (problema, strategia) returns soluzione o fallimento
```



# Algoritmo Tree-search

```
function tree-search (problema, strategia) returns soluzione o fallimento
```

1. Verifica se la frontiera è vuota



# Algoritmo Tree-search

```
function tree-search (problema, strategia) returns soluzione o fallimento
```

1. Verifica se la frontiera è vuota

- se è vuota, solleva un fallimento.  
se c'è almeno un elemento, passa al punto 2



# Algoritmo Tree-search

```
function tree-search (problema, strategia) returns soluzione o fallimento
```

1. Verifica se la frontiera è vuota
  - se è vuota, solleva un fallimento.
  - se c'è almeno un elemento, passa al punto 2
2. Scegli uno dei nodi della frontiera



# Algoritmo Tree-search

```
function tree-search (problema, strategia) returns soluzione o fallimento
```

1. Verifica se la frontiera è vuota

- se è vuota, solleva un fallimento.  
se c'è almeno un elemento, passa al punto 2

2. Scegli uno dei nodi della frontiera

- il modo in cui è scelto dipende dalla strategia
- se il nodo scelto è il nodo obiettivo, ricostruisci il cammino. Altrimenti aggiungi i nodi foglia.



# Algoritmo Tree-search

```
function tree-search (problema, strategia) returns soluzione o fallimento
```

1. Verifica se la frontiera è vuota
  - se è vuota, solleva un fallimento.
  - se c'è almeno un elemento, passa al punto 2
2. Scegli uno dei nodi della frontiera
  - il modo in cui è scelto dipende dalla strategia
  - se il nodo scelto è il nodo obiettivo, ricostruisci il cammino. Altrimenti aggiungi i nodi foglia.
3. Ritorna al punto 1



# Strategia nella Tree-search

La strategia deve essere valutata in termini di



# Strategia nella Tree-search

La strategia deve essere valutata in termini di

- COMPLESSITÀ
  - spaziale e temporale



# Strategia nella Tree-search

La strategia deve essere valutata in termini di

- COMPLESSITÀ
  - spaziale e temporale
- COMPLETEZZA



# Strategia nella Tree-search

La strategia deve essere valutata in termini di

- COMPLESSITÀ
  - spaziale e temporale
- COMPLETEZZA
- OTTIMALITÀ



# Ricerca in ampiezza

Espande i nodi a livelli.



# Ricerca in ampiezza

Espande i nodi a livelli.

Prima tutti i figli della radice, poi tutti i figli del primo figlio.

Poi tutti i figli del secondo figlio della radice e così via



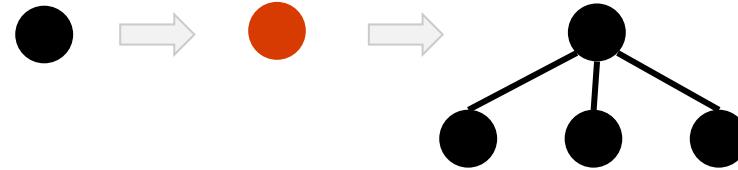
# Esempio Ricerca in ampiezza



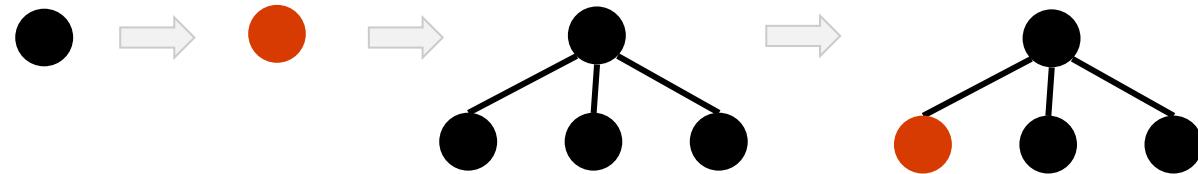
# Esempio Ricerca in ampiezza



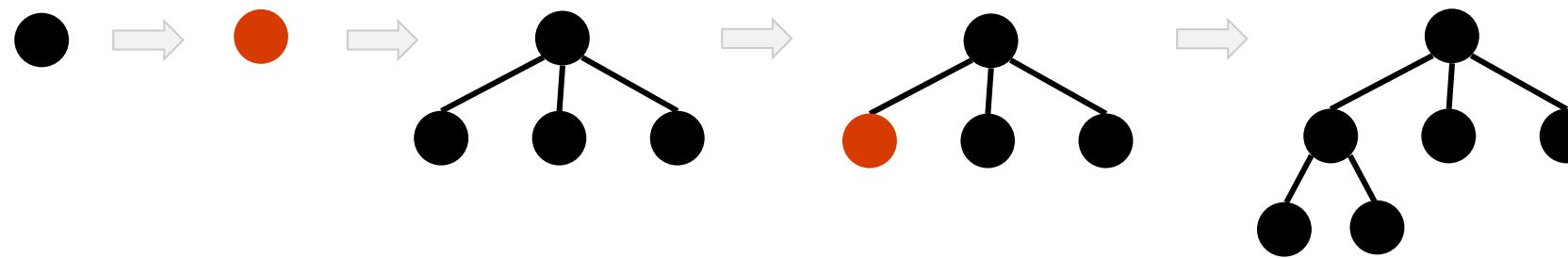
# Esempio Ricerca in ampiezza



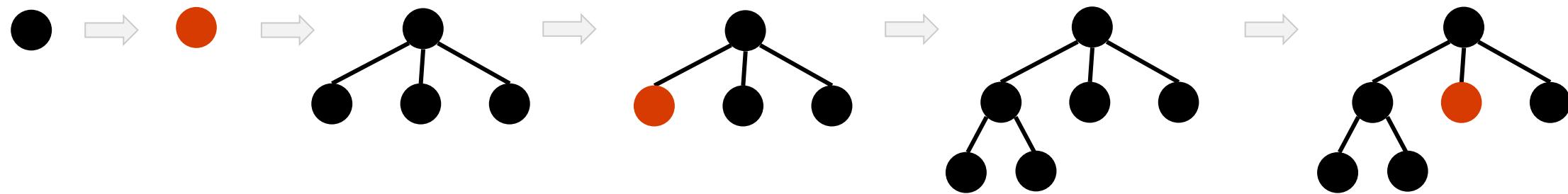
# Esempio Ricerca in ampiezza



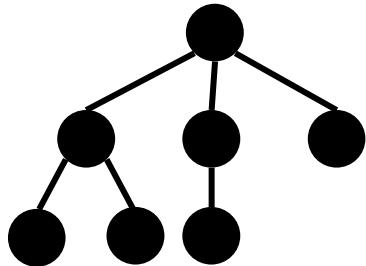
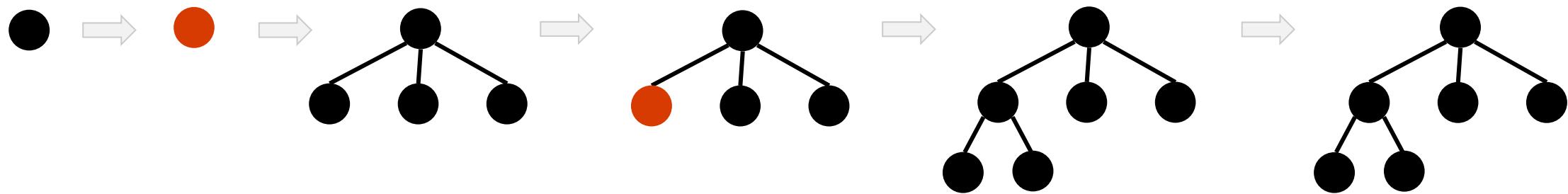
# Esempio Ricerca in ampiezza



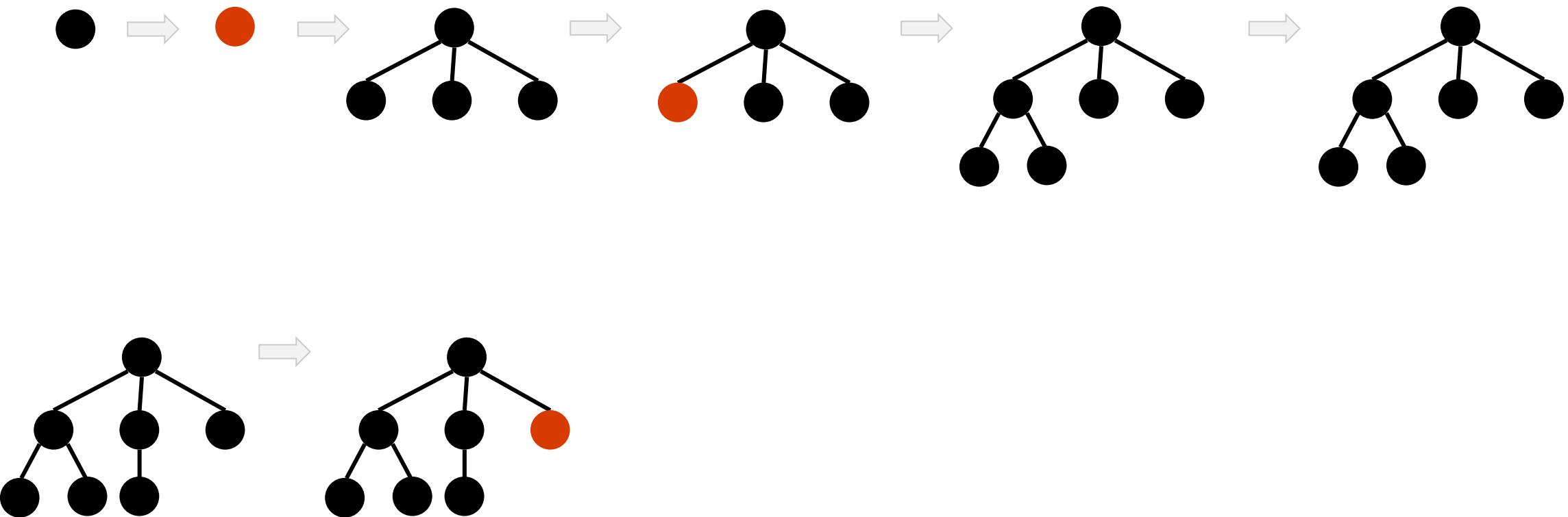
# Esempio Ricerca in ampiezza



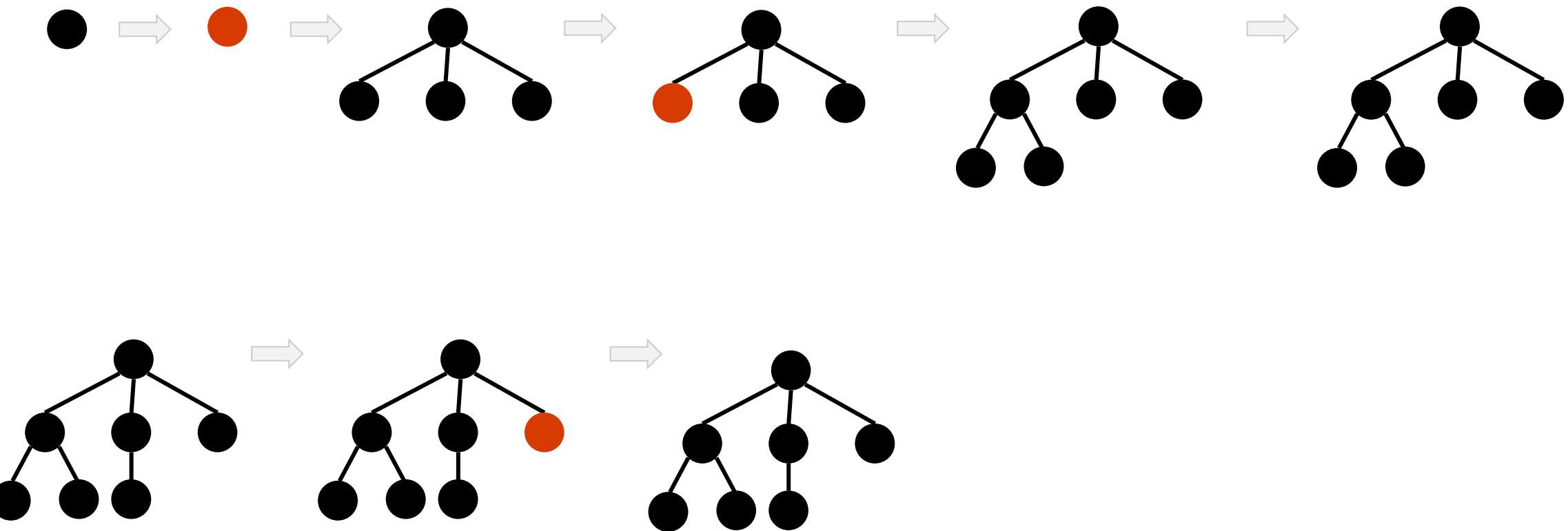
# Esempio Ricerca in ampiezza



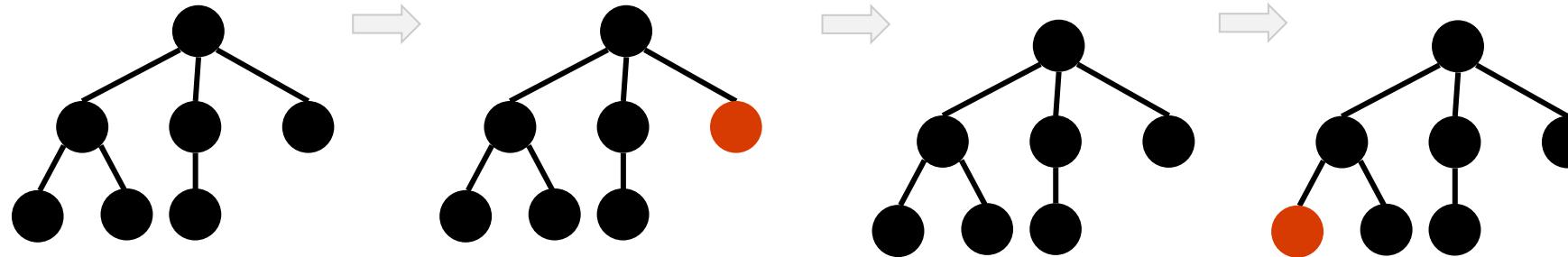
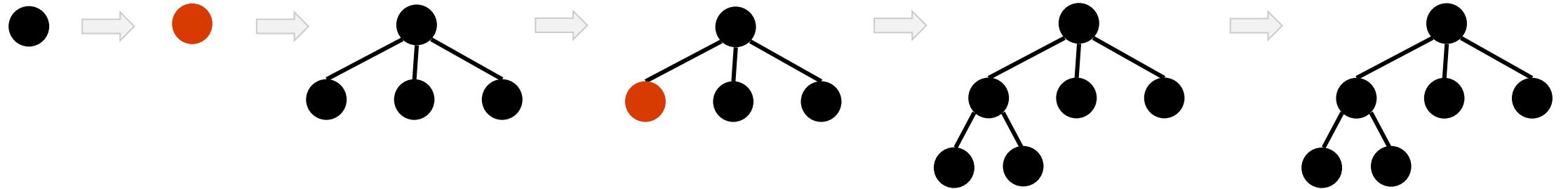
# Esempio Ricerca in ampiezza



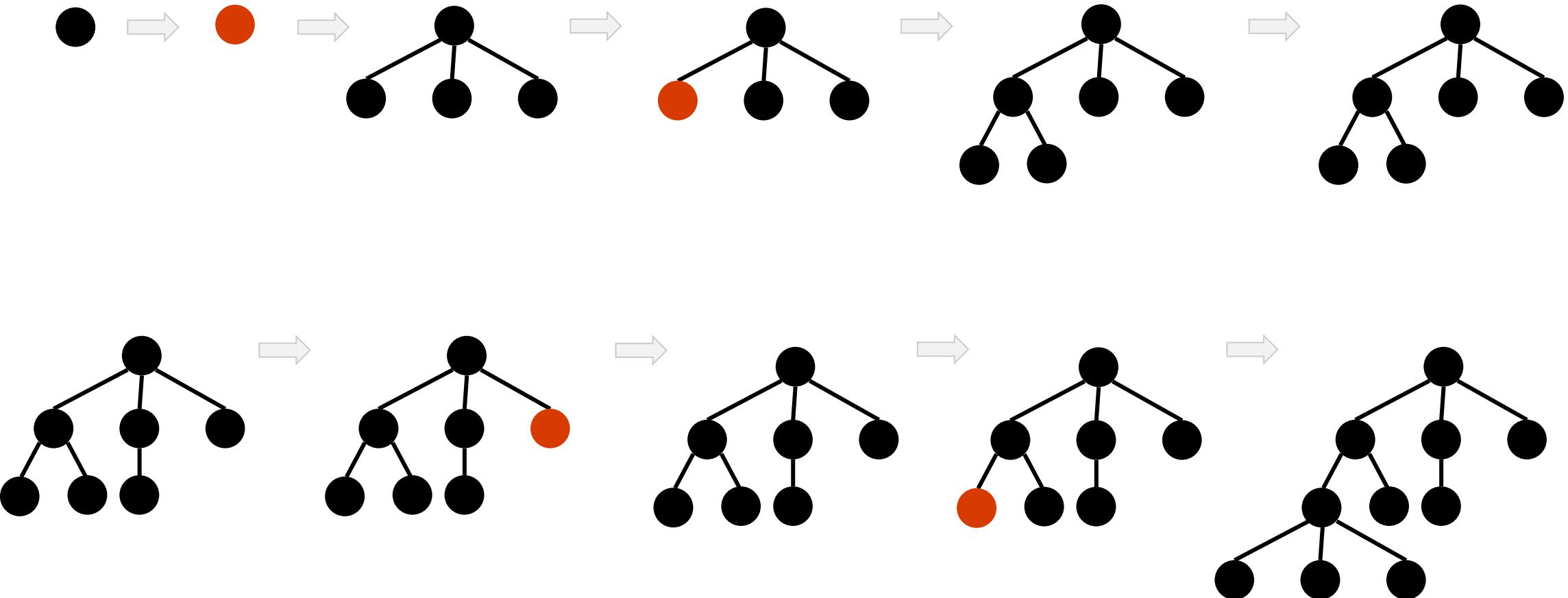
# Esempio Ricerca in ampiezza



# Esempio Ricerca in ampiezza



# Esempio Ricerca in ampiezza



# Caratteristiche Ricerca in ampiezza

COMPLETEZZA	Sì
OTTIMALITÀ	<p>Non necessariamente. È ottimo se il costo del cammino è una funzione <i>monotona</i> non decrescente della profondità nel nodo</p> $\text{depth}(n) < \text{depth}(m) \rightarrow \text{path. cost}(n) \leq \text{path. cost}(m)$
COMPLESSITÀ TEMPORALE	$O(b^d)$ $b$ è il fattore di ramificazione $d$ è la lunghezza minima di un cammino dal nodo iniziale alla soluzione  Ipotizziamo di trovarci a una profondità $d$ e la soluzione è l'ultimo nodo a destra. Per arrivare a visitare questo nodo devo aver visitato tutti i nodi in precedenza che sono $1 + b + b^2 + \dots + (b^{d+1} - b)$
COMPLESSITÀ IN SPAZIO	$O(b^d)$ Perché tutte le foglie sono salvate in memoria.





**DEMO**

# Ricerca guidata dal costo

Tra i nodi della frontiera espande prima quello che ha costo parziale  $g(n)$  minore.



# Ricerca guidata dal costo

Tra i nodi della frontiera espande prima quello che ha costo parziale  $g(n)$  minore.

NOTA: se  $g(n)=\text{depth}(n)$  si ha la ricerca in ampiezza



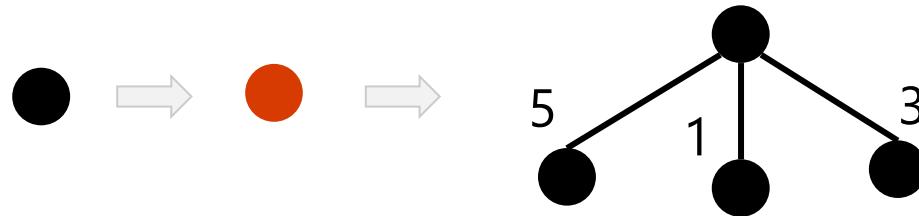
# Esempio Ricerca guidata dal costo



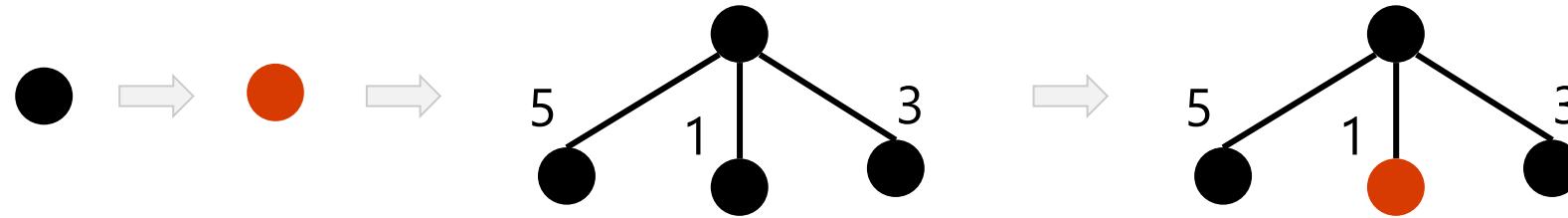
# Esempio Ricerca guidata dal costo



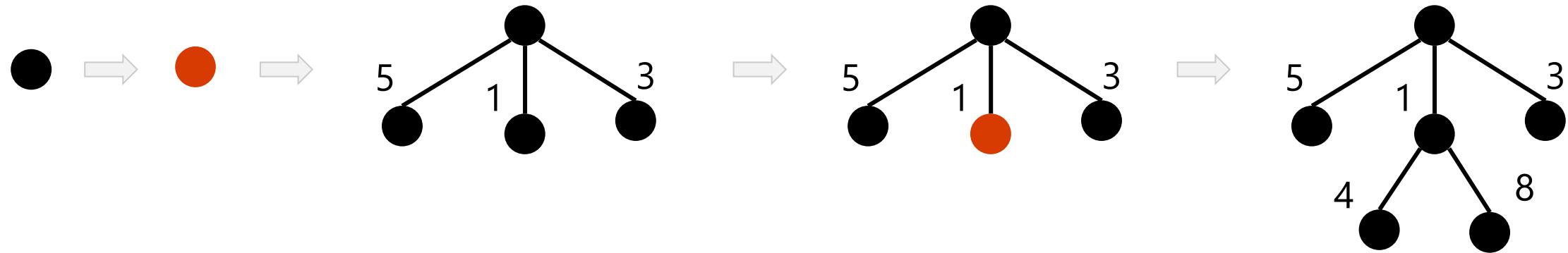
# Esempio Ricerca guidata dal costo



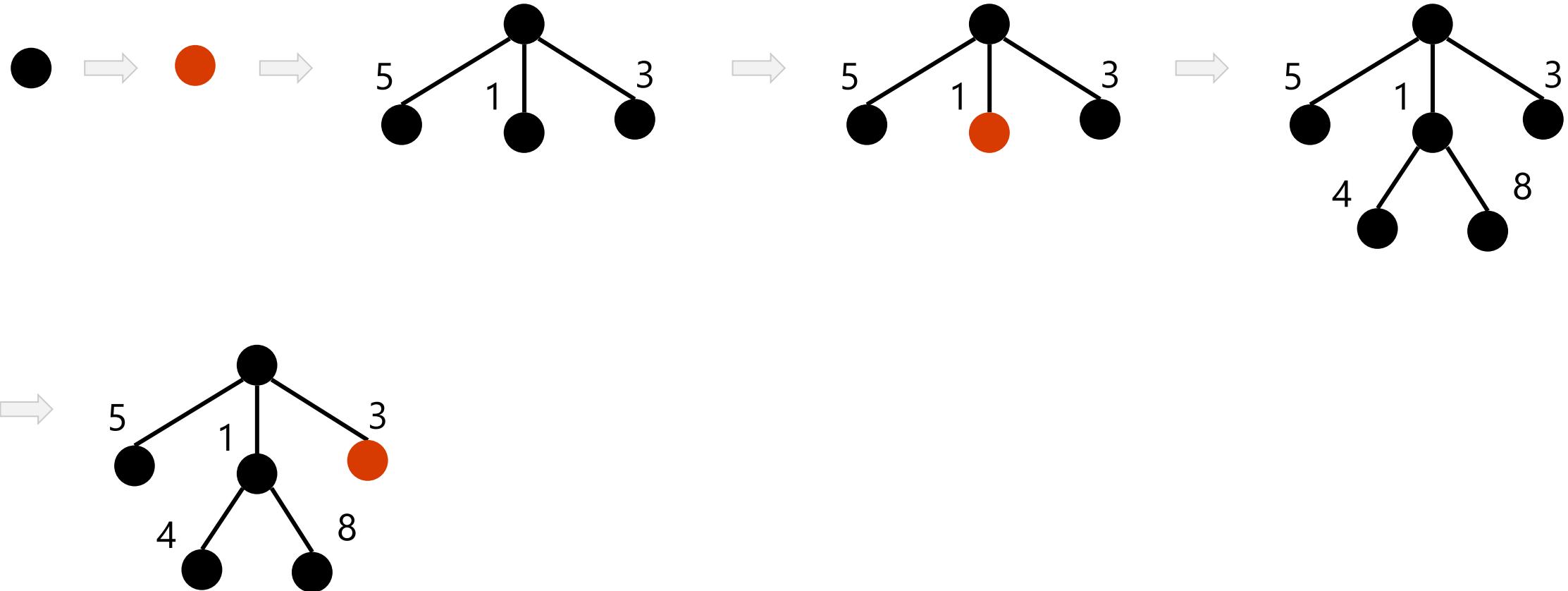
# Esempio Ricerca guidata dal costo



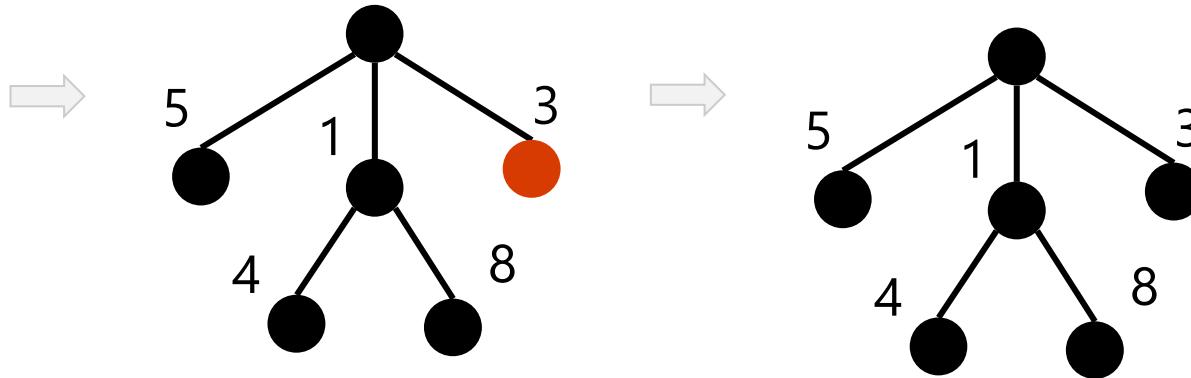
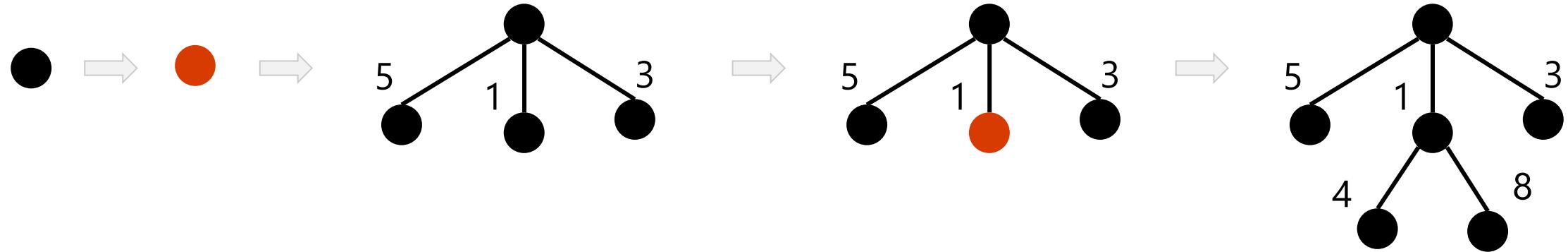
# Esempio Ricerca guidata dal costo



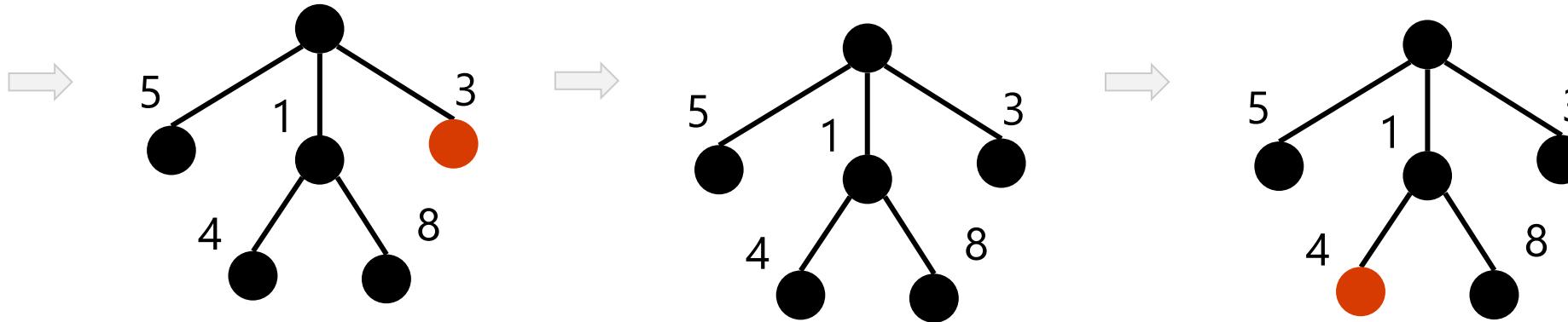
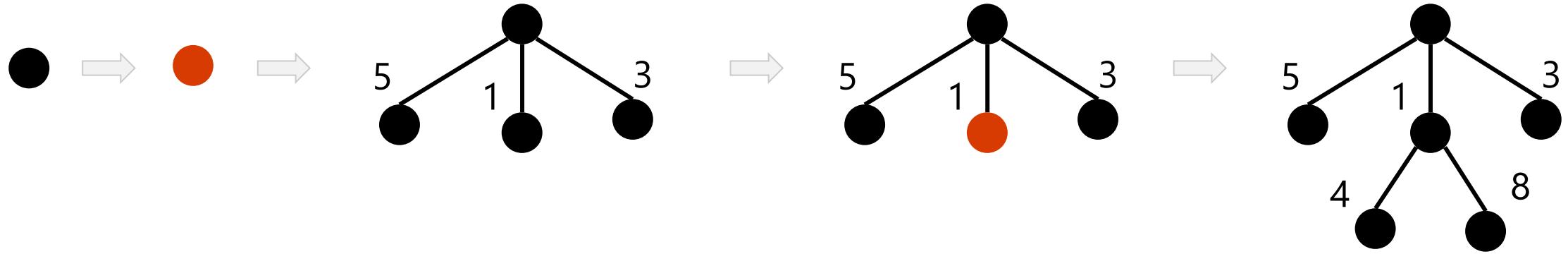
# Esempio Ricerca guidata dal costo



# Esempio Ricerca guidata dal costo



# Esempio Ricerca guidata dal costo



# Caratteristiche Ricerca guidata dal costo

<b>COMPLETEZZA</b>	Sì
<b>OTTIMALITÀ</b>	Non necessariamente. È ottimo se il costo di ogni step $g(\text{SUCCESSOR}(n)) - g(n)$ è sempre maggiore o uguale di una costante positiva piccola $\varepsilon$
<b>COMPLESSITÀ TEMPORALE</b>	$O(b^d)$ $b$ è il fattore di ramificazione $d$ è la lunghezza minima di un cammino dal nodo iniziale alla soluzione  Ipotizziamo di trovarci a una profondità $d$ e la soluzione è l'ultimo nodo a destra. Per arrivare a visitare questo nodo devo aver visitato tutti i nodi in precedenza che sono $1 + b + b^2 + \dots + (b^{d+1} - b)$
<b>COMPLESSITÀ IN SPAZIO</b>	$O(b^d)$ Perché tutte le foglie sono salvate in memoria.



# Ricerca in profondità

Una volta 'scelta' una strada la percorre fino a che è possibile.



# Ricerca in profondità

Una volta 'scelta' una strada la percorre fino a che è possibile.

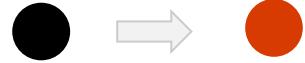
Solo dopo aver raggiunto un fallimento 'torna indietro' a provare una seconda strada.



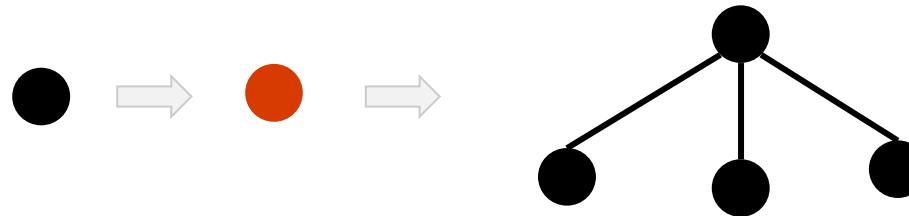
# Esempio Ricerca in profondità



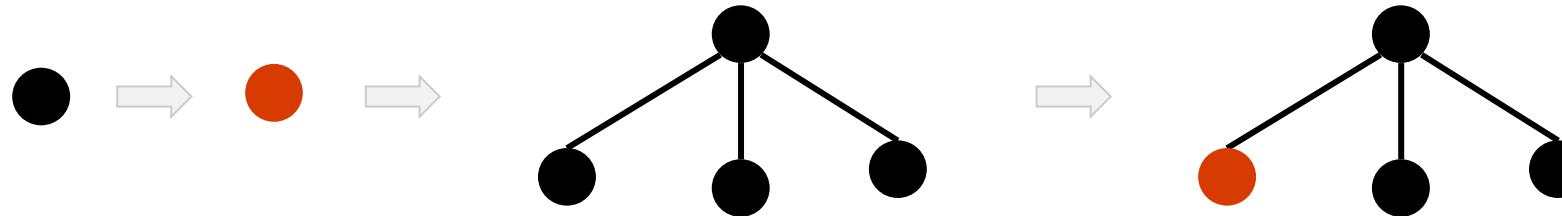
# Esempio Ricerca in profondità



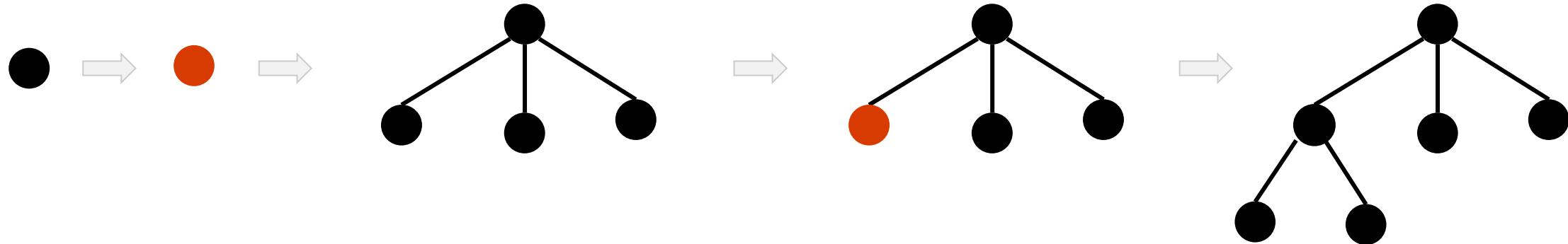
# Esempio Ricerca in profondità



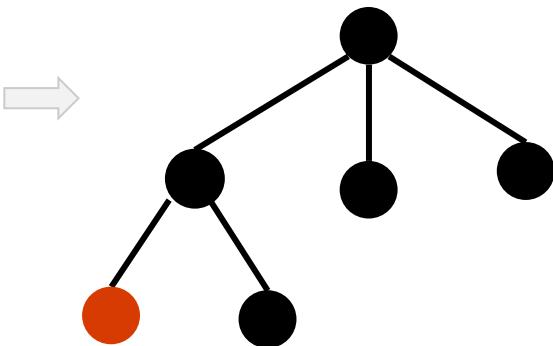
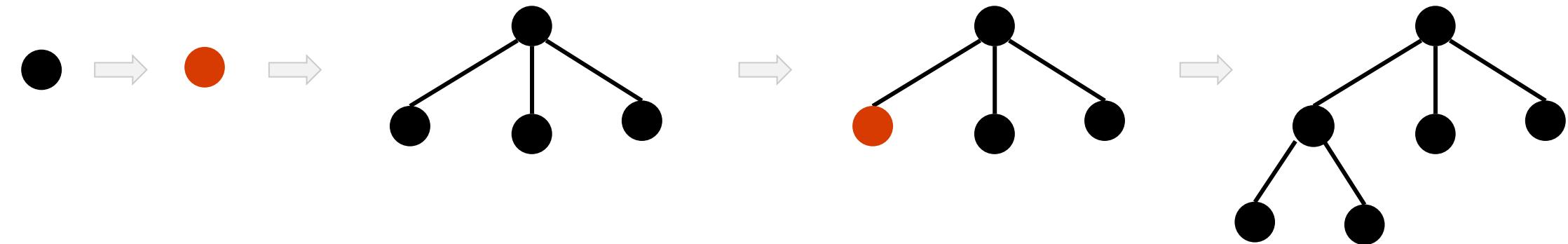
# Esempio Ricerca in profondità



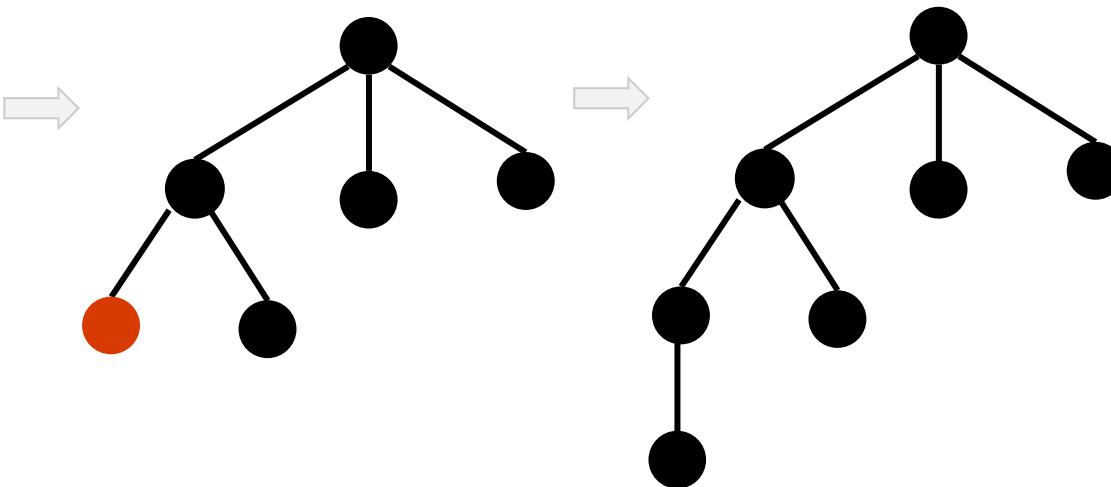
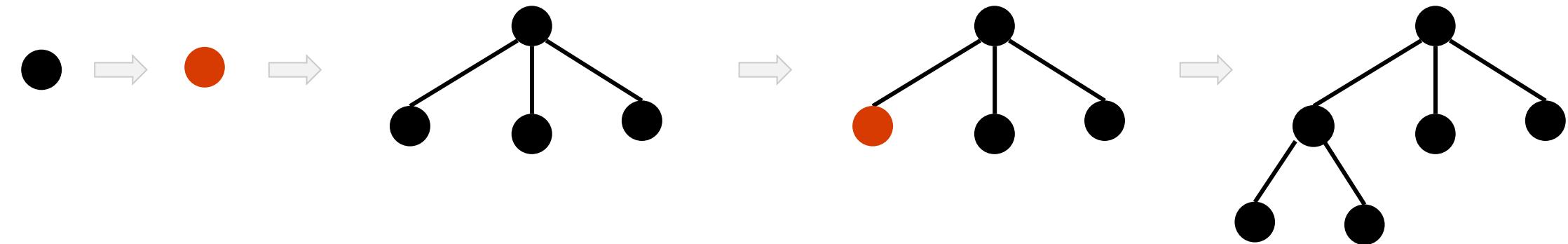
# Esempio Ricerca in profondità



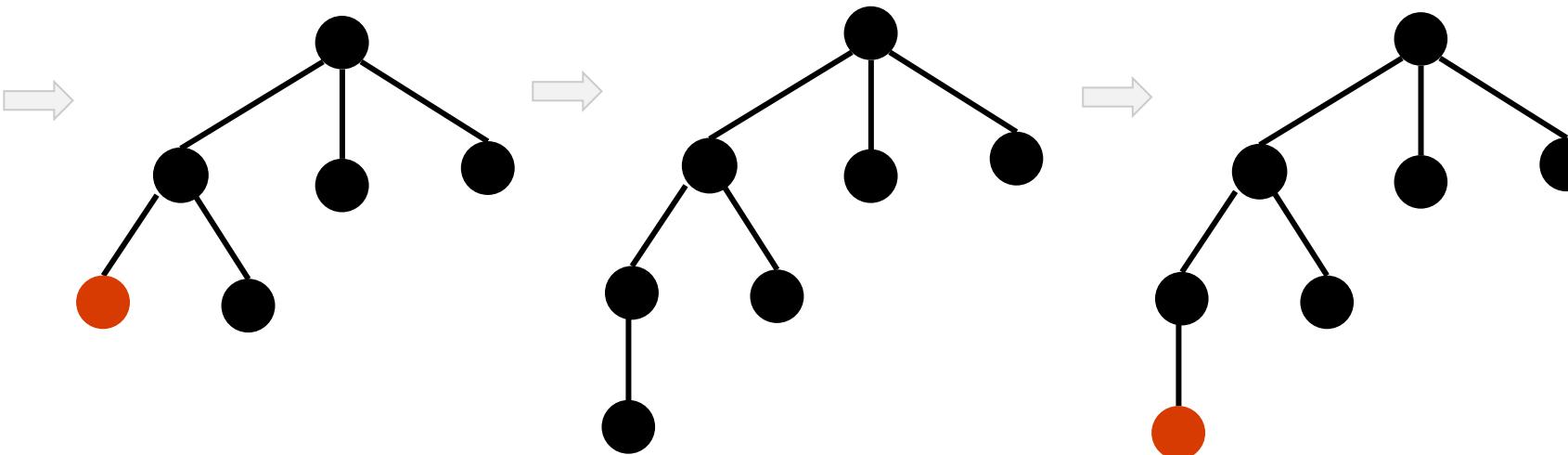
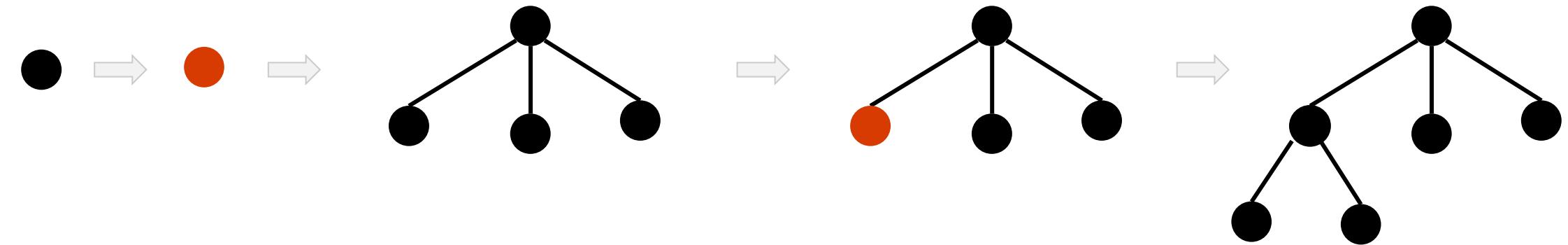
# Esempio Ricerca in profondità



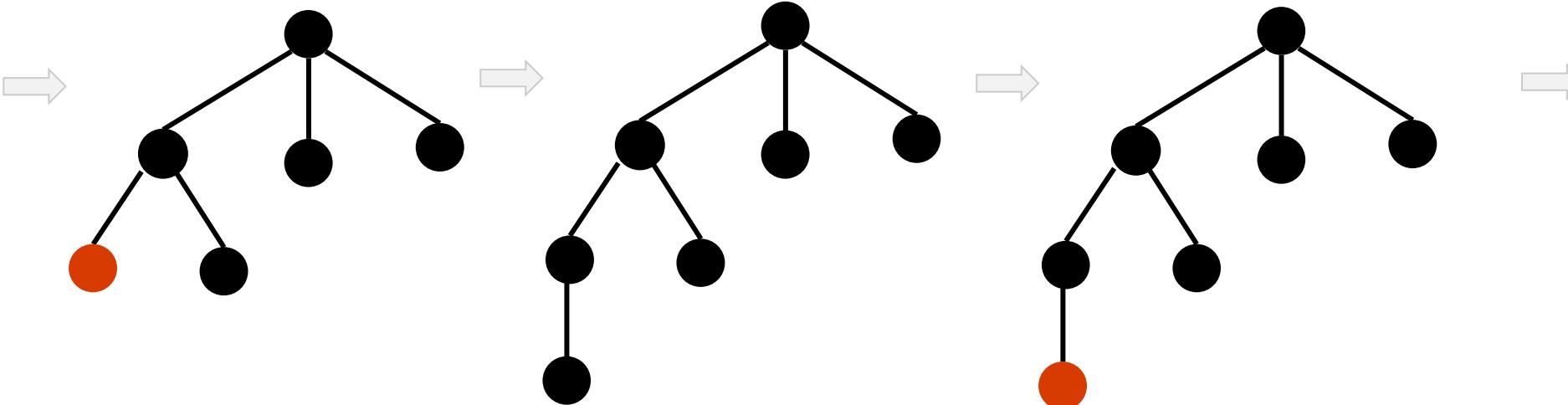
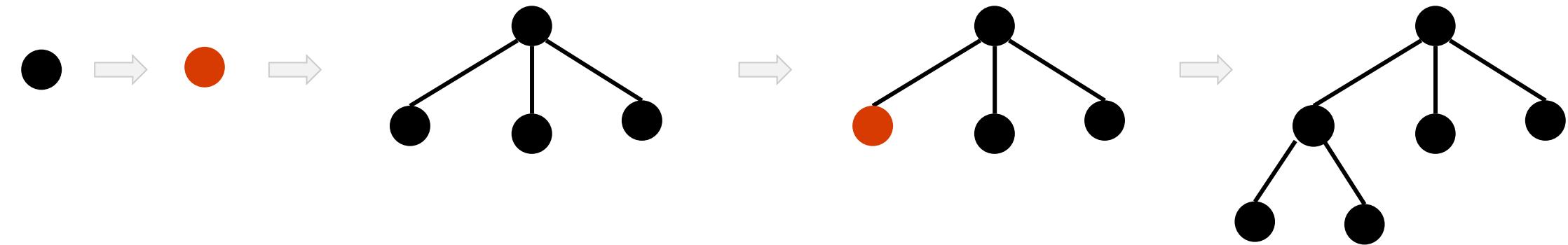
# Esempio Ricerca in profondità



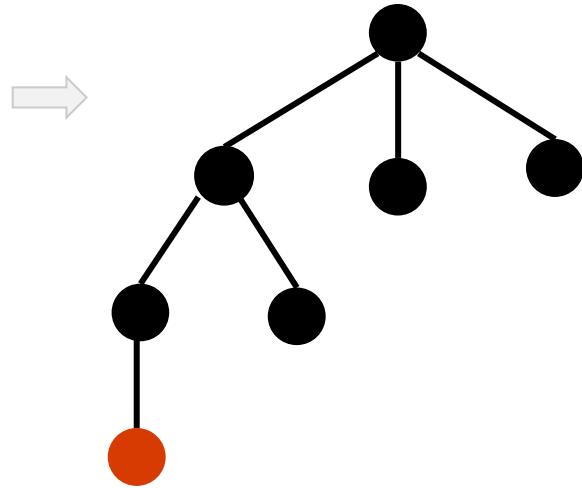
# Esempio Ricerca in profondità



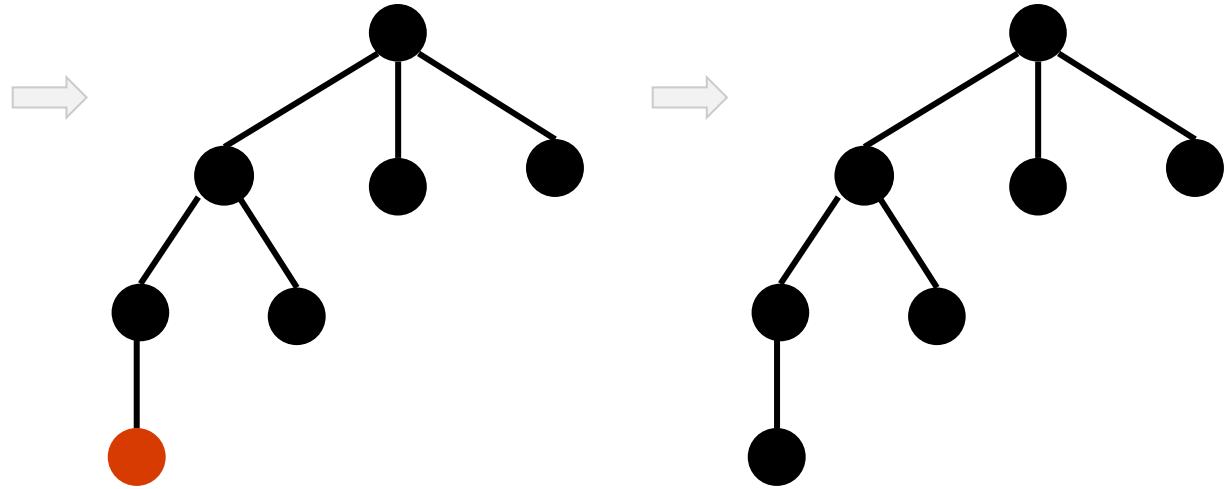
# Esempio Ricerca in profondità



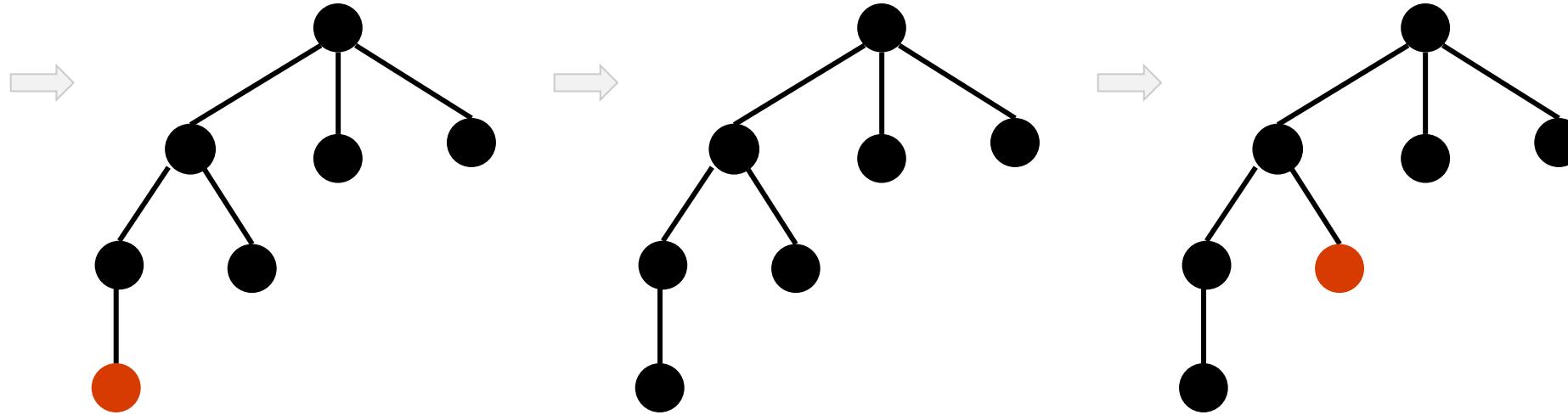
# Esempio Ricerca in profondità



# Esempio Ricerca in profondità



# Esempio Ricerca in profondità



# Caratteristiche Ricerca in profondità

<b>COMPLETEZZA</b>	No. Potrei avere un sottoalbero destro di dimensione infinita e la soluzione non si trova su questo
<b>OTTIMALITÀ</b>	Non necessariamente. Potrei avere due soluzioni: una nel primo sottoalbero e una nel secondo. Quest'ultima è ottima. L'algoritmo individua prima quella non ottima.
<b>COMPLESSITÀ TEMPORALE</b>	$O(b^m)$ $b$ è il fattore di ramificazione $m$ è la profondità massima dell'albero di ricerca  La prima soluzione potrebbe trovarsi nel sottoalbero più profondo.
<b>COMPLESSITÀ IN SPAZIO</b>	$O(b \cdot m)$ $b$ è il fattore di ramificazione $m$ è la profondità massima dell'albero di ricerca  La soluzione si trova a profondità $m$ e di ogni livello ho al più $b$ figli. In totale ho al più $b \cdot m$ nodi in memoria.





**DEMO**

# Varianti della Ricerca in profondità

- PROFONDITÀ LIMITATA
- PROFONDITÀ LIMITATA ITERATIVA



# Ricerca in profondità LIMITATA

Si decide di non espandere più i nodi a una profondità maggiore di  $k$ .



# Ricerca in profondità LIMITATA

Si decide di non espandere più i nodi a una profondità maggiore di  $k$ .

GUADAGNO: l'algoritmo potrebbe essere completo se la soluzione ha una lunghezza minore o uguale a  $k$



# Ricerca in profondità LIMITATA ITERATIVA

Non conoscendo  $k$  a priori si effettuano più cicli.

A ogni ciclo si incrementa  $k$ .



# Ricerca in profondità LIMITATA ITERATIVA

Non conoscendo  $k$  a priori si effettuano più cicli.  
A ogni ciclo si incrementa  $k$ .

GUADAGNO: l'algoritmo è completo.



# Ricerca in profondità LIMITATA ITERATIVA

Non conoscendo  $k$  a priori si effettuano più cicli.  
A ogni ciclo si incrementa  $k$ .

GUADAGNO: l'algoritmo è completo.

GUADAGNO: è ottimo se il costo del cammino è funzione *monotona non decrescente* della profondità del nodo.



# Tree search vs Graph search

L'approccio appena visto soffre di un grave problema: non effettua nessuno controllo sugli stati già visitati.



# Tree search vs Graph search

L'approccio appena visto soffre di un grave problema: non effettua nessuno controllo sugli stati già visitati.

Potrei trasformare un problema *lineare* in uno *esponenziale*.



# Tree search vs Graph search

L'approccio appena visto soffre di un grave problema: non effettua nessuno controllo sugli stati già visitati.

Potrei trasformare un problema *lineare* in uno *esponenziale*.

Devo tener conto degli stati già visitati e non rivisitarli.



# Tree search vs Graph search

ATTENZIONE.

Ragionamento, in generale, errato.



# Tree search vs Graph search

ATTENZIONE.

Ragionamento, in generale, errato.

Visito per la seconda volta un nodo.

Scarto il cammino che mi ci ha portato.

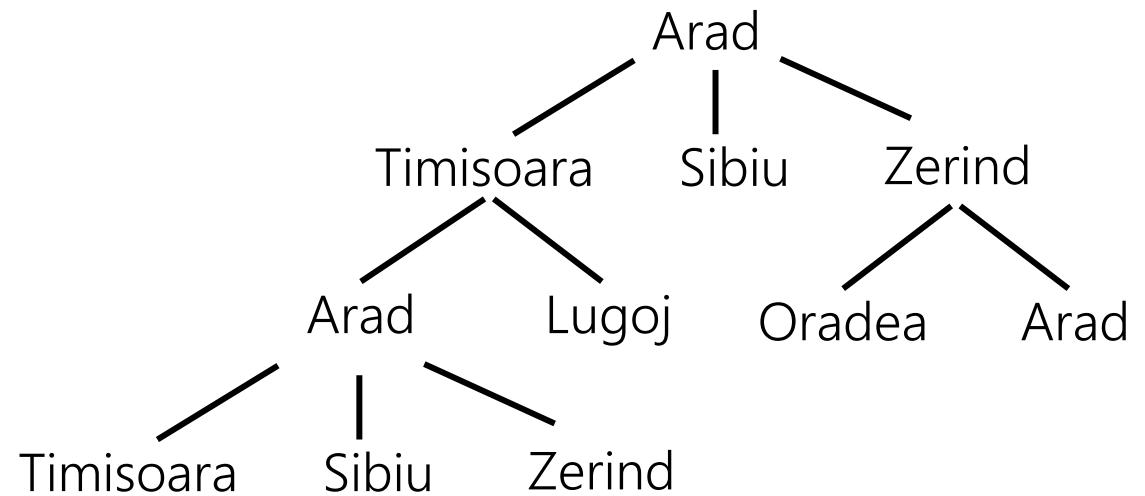
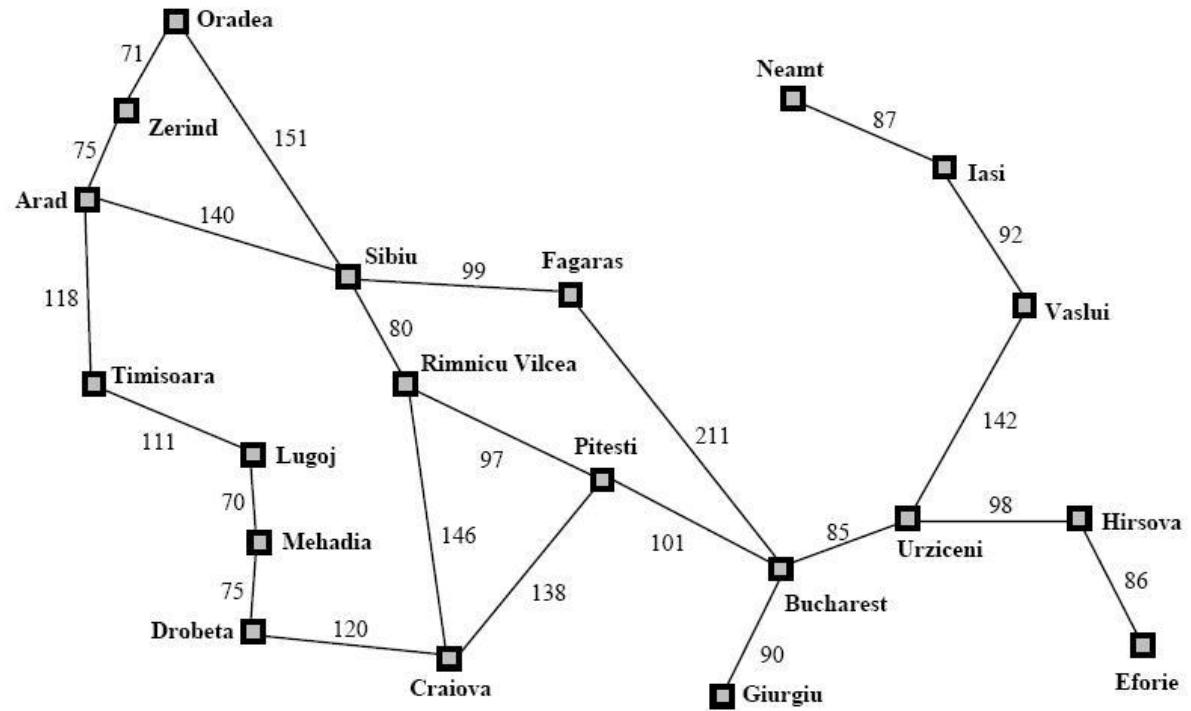
NON ho informazioni per dire che il secondo cammino sia peggiore del primo.

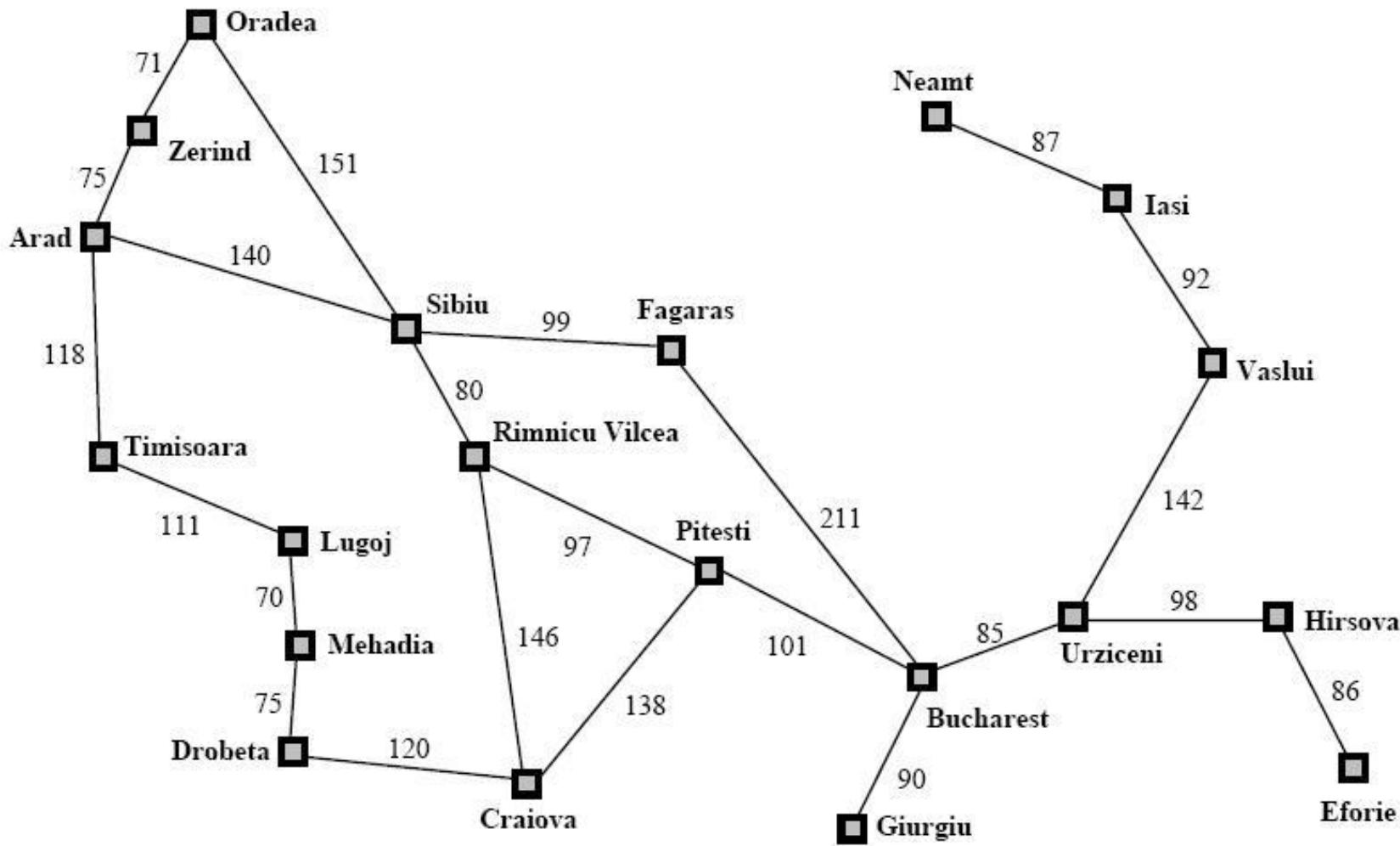


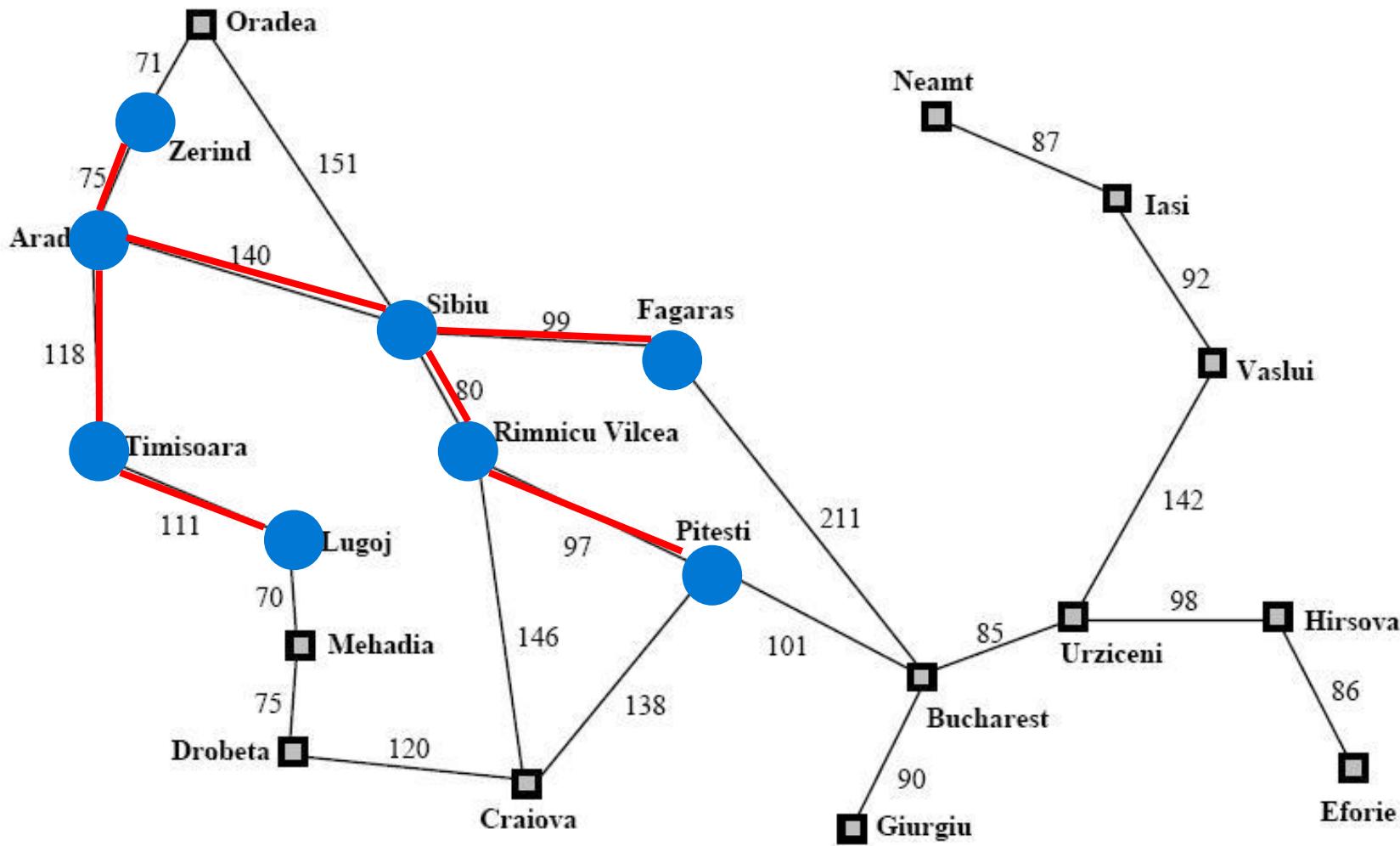
# Graph Search

L'idea è quella di costruire un albero di ricerca direttamente sullo *spazio* degli stati









# Ricerca informata

Ciò che abbiamo visto si definisce *ricerca non informata* (o *cieca*).



# Ricerca informata

Ciò che abbiamo visto si definisce *ricerca non informata* (o *cieca*).

Analizziamo ora la *ricerca informata*.



# Funzione di valutazione

Utilizziamo una *funzione di valutazione*  $f$  che esprime la desiderabilità di espandere il nodo.



# Funzione di valutazione

Utilizziamo una *funzione di valutazione*  $f$  che esprime la desiderabilità di espandere il nodo.

Espandere il nodo che ha desiderabilità più alta significa effettuare una *ricerca best-first*.



# Esempio

Da Roma a Milano.



# Ricerca golosa

Utilizza come funzione di valutazione  $f(n) = h(n)$  dove  $h(n)$  è la stima di  $h^*(n)$ .



# Ricerca golosa

Utilizza come funzione di valutazione  $f(n) = h(n)$  dove  $h(n)$  è la stima di  $h^*(n)$ .

$h^*(n)$  è il costo effettivo del cammino minimo da  $n$  fino al nodo obiettivo.



# Caratteristiche Ricerca golosa

<b>COMPLETEZZA</b>	No.  Potrebbe seguire un cammino infinito senza trovare mai la soluzione. (stessi motivi della ricerca in profondità)
<b>OTTIMALITÀ</b>	No.  (stessi motivi ricerca in profondità)
<b>COMPLESSITÀ TEMPORALE</b>	$O(b^m)$  $b$ è il fattore di ramificazione $m$ è la profondità massima dell'albero di ricerca
<b>COMPLESSITÀ IN SPAZIO</b>	$O(b^m)$  $b$ è il fattore di ramificazione $m$ è la profondità massima dell'albero di ricerca  La soluzione si trova a profondità $m$ e di ogni livello ho al più $b$ figli. In totale ho al più $b \cdot m$ nodi in memoria.

# Algoritmo $A^*$

Utilizza come funzione di valutazione  $f(n) = h(n) + g(n)$  dove  $h(n)$  è la stima di  $h^*(n)$



# Algoritmo $A^*$

Utilizza come funzione di valutazione  $f(n) = h(n) + g(n)$  dove  $h(n)$  è la stima di  $h^*(n)$

$h^*(n)$  è il costo effettivo del cammino minimo da  $n$  fino al nodo obiettivo.

$g(n)$  è il costo del cammino dal nodo iniziale al nodo  $n$ .



# Algoritmo $A^*$

Utilizza come funzione di valutazione  $f(n) = h(n) + g(n)$  dove  $h(n)$  è la stima di  $h^*(n)$

$h^*(n)$  è il costo effettivo del cammino minimo da  $n$  fino al nodo obiettivo.

$g(n)$  è il costo del cammino dal nodo iniziale al nodo  $n$ .

È completo e ottimo se l'euristica è *ammissibile*



# Algoritmo $A^*$

Euristica ammissibile se

$$\forall n \ h(n) \leq h^*(n)$$



# Algoritmo di Dijkstra



# TAKE A BREAK!



# Algoritmo di Dijkstra

Permette di trovare il cammino di *costo minimo* in un grafo.



# Algoritmo di Dijkstra

Permette di trovare il cammino di *costo minimo* in un grafo.

NOTA

il grafo deve avere pesi non negati.

Nel caso, utilizzare algoritmo di Bellman-Ford



# Algoritmo di Dijkstra

L'insieme dei nodi è partizionato in 3 parti



# Algoritmo di Dijkstra

L'insieme dei nodi è partizionato in 3 parti

- $V \rightarrow$  nodi visitati
- $F \rightarrow$  nodi di frontiera (nodi successori di quelli visitati raggiungibili con un solo arco)
- $K \rightarrow$  nodi da visitare



# Algoritmo di Dijkstra

L'insieme dei nodi è partizionato in 3 parti

- $V \rightarrow$  nodi visitati
- $F \rightarrow$  nodi di frontiera (nodi successori di quelli visitati raggiungibili con un solo arco e non ancora visitati)
- $K \rightarrow$  nodi da visitare

2 valori di supporto per il nodo  $i$  :

- $d_i$ : distanza minima attuale dal nodo di partenza fino a  $i$
- $p_i$ : nodo precedente di  $i$  nel cammino minimo attuale



# Algoritmo di Dijkstra

Situazione iniziale



# Algoritmo di Dijkstra

Situazione iniziale

- $V \rightarrow$  insieme vuoto
- $F \rightarrow$  nodo di partenza
- $K \rightarrow$  tutti gli altri nodi



# Algoritmo di Dijkstra

Situazione iniziale

- $V \rightarrow$  insieme vuoto
- $F \rightarrow$  nodo di partenza
- $K \rightarrow$  tutti gli altri nodi

$d_i: \infty$ , per tutti i nodi  $i$  tranne che per la sorgente  $d_{sorgente} = 0$

$p_i: \emptyset$



# Algoritmo di Dijkstra

Fino a che  $F$  non è vuoto



# Algoritmo di Dijkstra

Fino a che  $F$  non è vuoto

1. prendi da  $F$  il nodo  $j$  che ha la distanza più piccola dalla sorgente



# Algoritmo di Dijkstra

Fino a che  $F$  non è vuoto

1. prendi da  $F$  il nodo  $j$  che ha la distanza più piccola dalla sorgente
2. sposta  $j$  in  $V$



# Algoritmo di Dijkstra

Fino a che  $F$  non è vuoto

1. prendi da  $F$  il nodo  $j$  che ha la distanza più piccola dalla sorgente
2. sposta  $j$  in  $V$
3. considera i vicini di  $j$  non ancora visitati e per ognuno di essi (indichiamolo con  $w$ ) ricalcola  $d_w$  scegliendo il minimo tra  $d_w$  e  $d_j + a_{j,w}$  ( $a_{j,w}$  è il costo dell'arco da  $j$  a  $w$ ). Inserisci  $w$  in  $F$ .



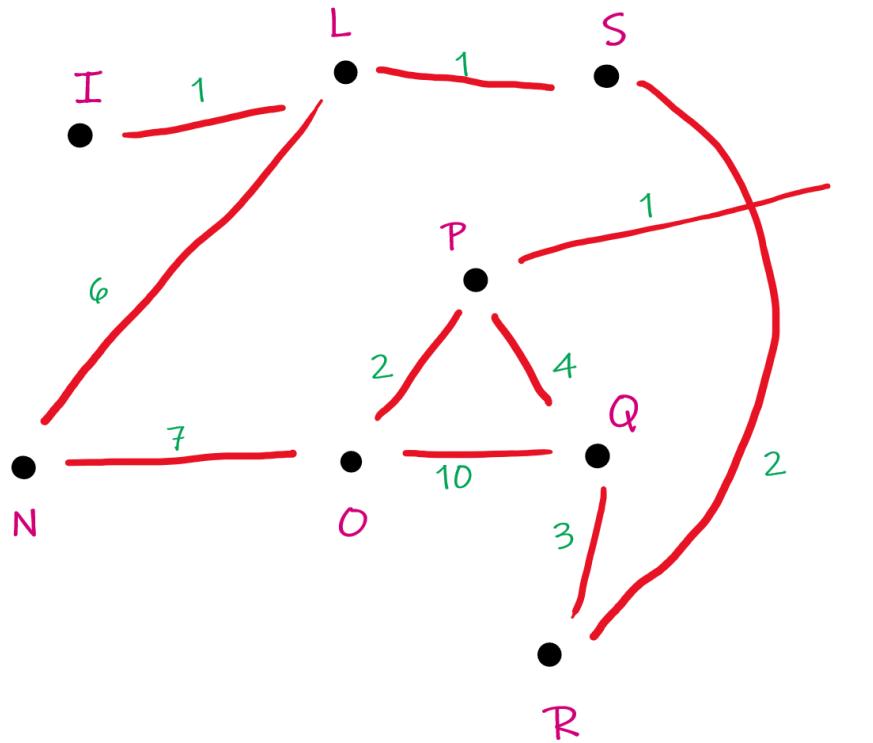
# Algoritmo di Dijkstra

Fino a che  $F$  non è vuoto

1. prendi da  $F$  il nodo  $j$  che ha la distanza più piccola dalla sorgente
2. sposta  $j$  in  $V$
3. considera i vicini di  $j$  non ancora visitati e per ognuno di essi (indichiamolo con  $w$ ) ricalcola  $d_w$  scegliendo il minimo tra  $d_w$  e  $d_j + a_{j,w}$  ( $a_{j,w}$  è il costo dell'arco da  $j$  a  $w$ ). Inserisci  $w$  in  $F$ .
4. se si è aggiornato  $d_w$ , è necessario aggiornare  $p_w \leftarrow j$



# Esempio



Cammino ottimo da *I* a *Z*

# Algoritmo di Dijkstra

Situazione iniziale

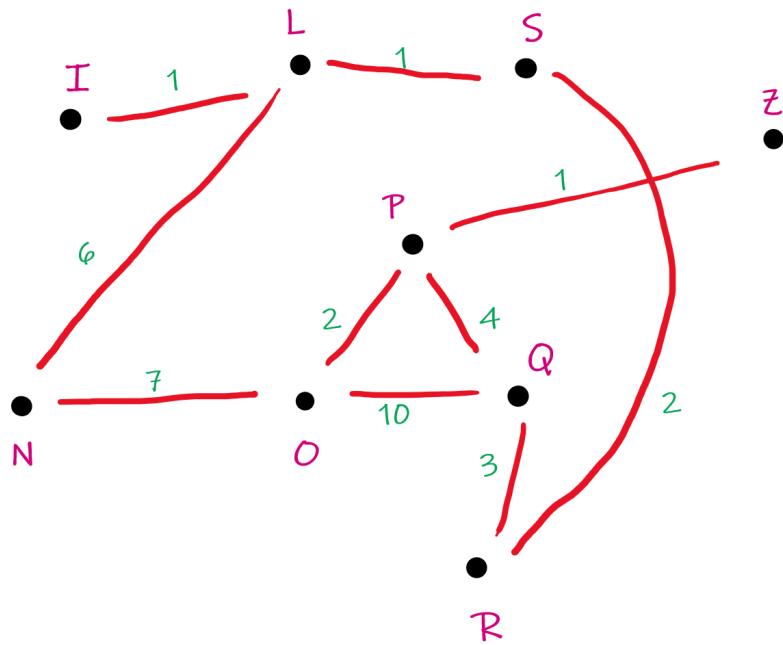
- $V \rightarrow$  insieme vuoto
- $F \rightarrow$  nodo di partenza
- $K \rightarrow$  tutti gli altri nodi

$d_i: \infty$ , per tutti i nodi  $i$  tranne che per la sorgente  $d_{sorgente} = 0$

$p_i: \emptyset$



# Esempio



$d_I$	0
$d_L$	$\infty$
$d_S$	$\infty$
$d_R$	$\infty$
$d_Q$	$\infty$
$d_P$	$\infty$
$d_O$	$\infty$
$d_N$	$\infty$
$d_Z$	$\infty$

$p_I$	
$p_L$	
$p_S$	
$p_R$	
$p_Q$	
$p_P$	
$p_O$	
$p_N$	
$p_Z$	

$$V = \{\}$$

$$F = \{I\}$$

$$K = \{I, L, S, R, Q, P, O, N, Z\}$$

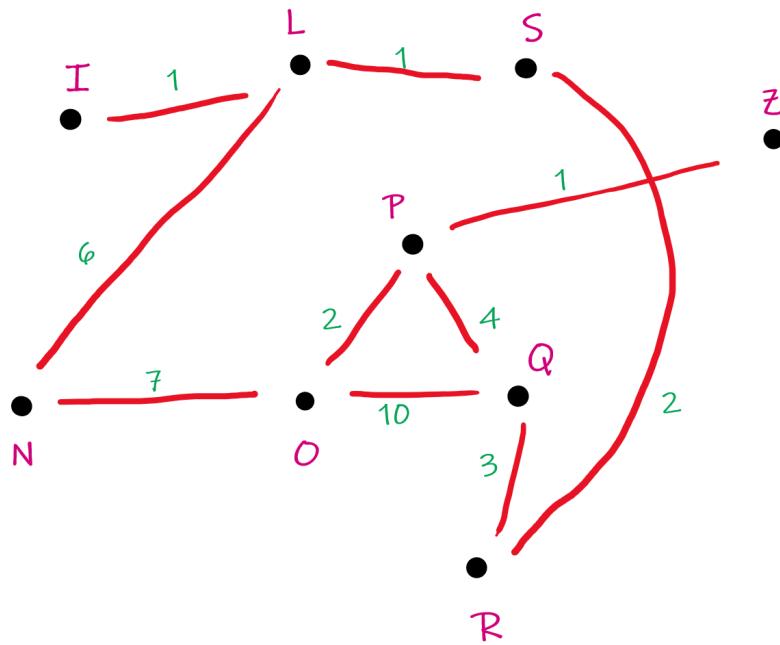
# Algoritmo di Dijkstra

Fino a che  $F$  non è vuoto

1. prendi da  $F$  il nodo  $j$  che ha la distanza più piccola dalla sorgente
2. sposta  $j$  in  $V$
3. considera i vicini di  $j$  non ancora visitati e per ognuno di essi (indichiamolo con  $w$ ) ricalcola  $d_w$  scegliendo il minimo tra  $d_w$  e  $d_j + a_{j,w}$  ( $a_{j,w}$  è il costo dell'arco da  $j$  a  $w$ ). Inserisci  $w$  in  $F$ .
4. se si è aggiornato  $d_w$ , è necessario aggiornare  $p_w \leftarrow j$



# Esempio



$d_I$	0
$d_L$	$\infty$
$d_S$	$\infty$
$d_R$	$\infty$
$d_Q$	$\infty$
$d_P$	$\infty$
$d_O$	$\infty$
$d_N$	$\infty$
$d_Z$	$\infty$

$p_I$	
$p_L$	
$p_S$	
$p_R$	
$p_Q$	
$p_P$	
$p_O$	
$p_N$	
$p_Z$	

$$V = \{\}$$

$$F = \{I\}$$

$$K = \{I, L, S, R, Q, P, O, N, Z\}$$

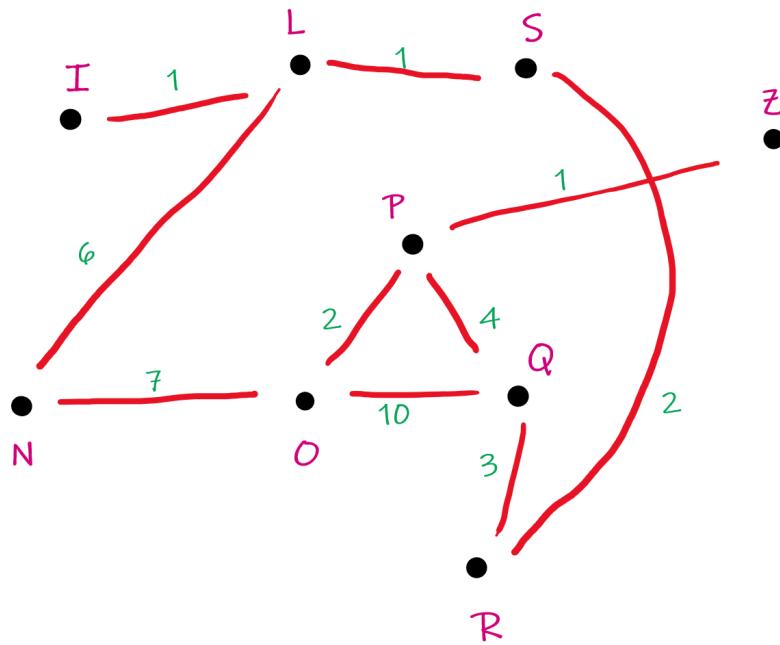
# Algoritmo di Dijkstra

Fino a che  $F$  non è vuoto

1. prendi da  $F$  il nodo  $j$  che ha la distanza più piccola dalla sorgente
2. sposta  $j$  in  $V$
3. considera i vicini di  $j$  non ancora visitati e per ognuno di essi (indichiamolo con  $w$ ) ricalcola  $d_w$  scegliendo il minimo tra  $d_w$  e  $d_j + a_{j,w}$  ( $a_{j,w}$  è il costo dell'arco da  $j$  a  $w$ ). Inserisci  $w$  in  $F$ .
4. se si è aggiornato  $d_w$ , è necessario aggiornare  $p_w \leftarrow j$



# Esempio



$d_I$	0
$d_L$	$\infty$
$d_S$	$\infty$
$d_R$	$\infty$
$d_Q$	$\infty$
$d_P$	$\infty$
$d_O$	$\infty$
$d_N$	$\infty$
$d_Z$	$\infty$

$p_I$	
$p_L$	
$p_S$	
$p_R$	
$p_Q$	
$p_P$	
$p_O$	
$p_N$	
$p_Z$	

$$V = \{I\}$$

$$F = \{\}$$

$$K = \{L, S, R, Q, P, O, N, Z\}$$

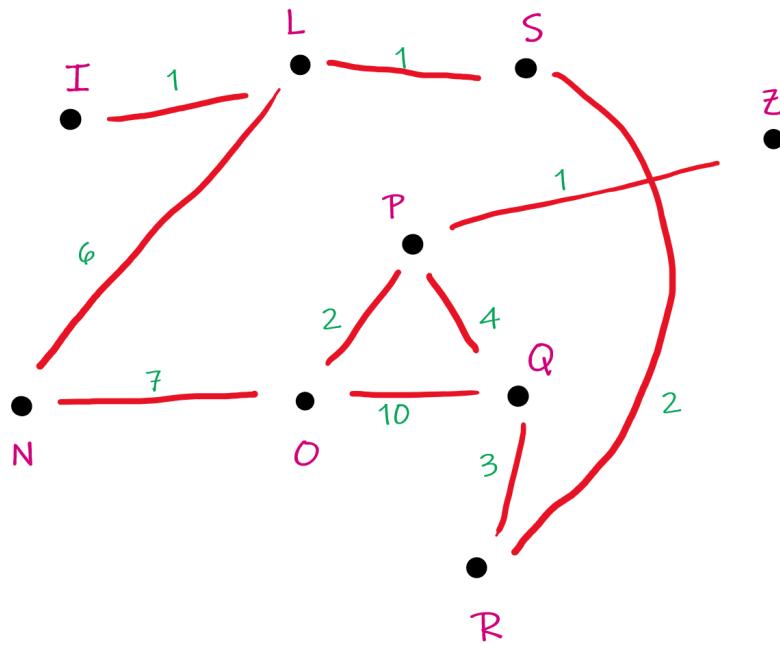
# Algoritmo di Dijkstra

Fino a che  $F$  non è vuoto

1. prendi da  $F$  il nodo  $j$  che ha la distanza più piccola dalla sorgente
2. sposta  $j$  in  $V$
3. considera i vicini di  $j$  non ancora visitati e per ognuno di essi (indichiamolo con  $w$ ) ricalcola  $d_w$  scegliendo il minimo tra  $d_w$  e  $d_j + a_{j,w}$  ( $a_{j,w}$  è il costo dell'arco da  $j$  a  $w$ ). Inserisci  $w$  in  $F$ .
4. se si è aggiornato  $d_w$ , è necessario aggiornare  $p_w \leftarrow j$



# Esempio



$d_I$	0
$d_L$	$\infty$
$d_S$	$\infty$
$d_R$	$\infty$
$d_Q$	$\infty$
$d_P$	$\infty$
$d_O$	$\infty$
$d_N$	$\infty$
$d_Z$	$\infty$

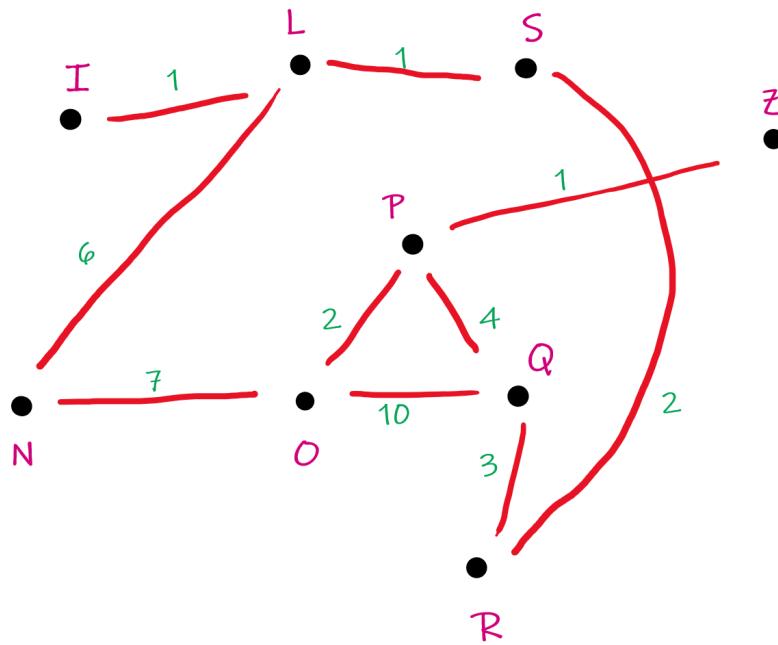
$p_I$	
$p_L$	
$p_S$	
$p_R$	
$p_Q$	
$p_P$	
$p_O$	
$p_N$	
$p_Z$	

$$V = \{I\}$$

$$F = \{L\}$$

$$K = \{L, S, R, Q, P, O, N, Z\}$$

# Esempio



$d_I$	0
$d_L$	1
$d_S$	$\infty$
$d_R$	$\infty$
$d_Q$	$\infty$
$d_P$	$\infty$
$d_O$	$\infty$
$d_N$	$\infty$
$d_Z$	$\infty$

$p_I$	
$p_L$	
$p_S$	
$p_R$	
$p_Q$	
$p_P$	
$p_O$	
$p_N$	
$p_Z$	

$$V = \{I\}$$

$$F = \{L\}$$

$$K = \{L, S, R, Q, P, O, N, Z\}$$

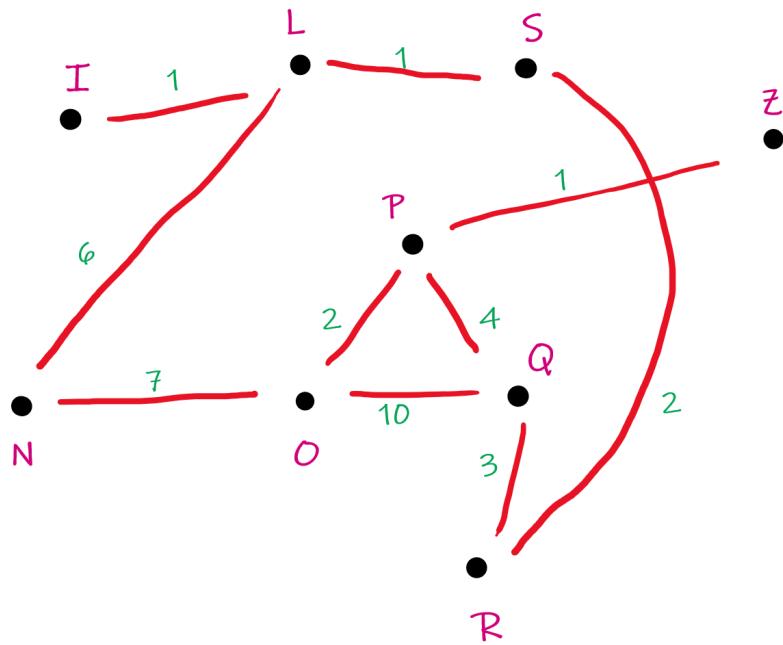
# Algoritmo di Dijkstra

Fino a che  $F$  non è vuoto

1. prendi da  $F$  il nodo  $j$  che ha la distanza più piccola dalla sorgente
2. sposta  $j$  in  $V$
3. considera i vicini di  $j$  non ancora visitati e per ognuno di essi (indichiamolo con  $w$ ) ricalcola  $d_w$  scegliendo il minimo tra  $d_w$  e  $d_j + a_{j,w}$  ( $a_{j,w}$  è il costo dell'arco da  $j$  a  $w$ ). Inserisci  $w$  in  $F$ .
4. se si è aggiornato  $d_w$ , è necessario aggiornare  $p_w \leftarrow j$



# Esempio



$d_I$	0
$d_L$	1
$d_S$	$\infty$
$d_R$	$\infty$
$d_Q$	$\infty$
$d_P$	$\infty$
$d_O$	$\infty$
$d_N$	$\infty$
$d_Z$	$\infty$

$p_I$	
$p_L$	$p_I$
$p_S$	
$p_R$	
$p_Q$	
$p_P$	
$p_O$	
$p_N$	
$p_Z$	

$$V = \{I\}$$

$$F = \{L\}$$

$$K = \{L, S, R, Q, P, O, N, Z\}$$

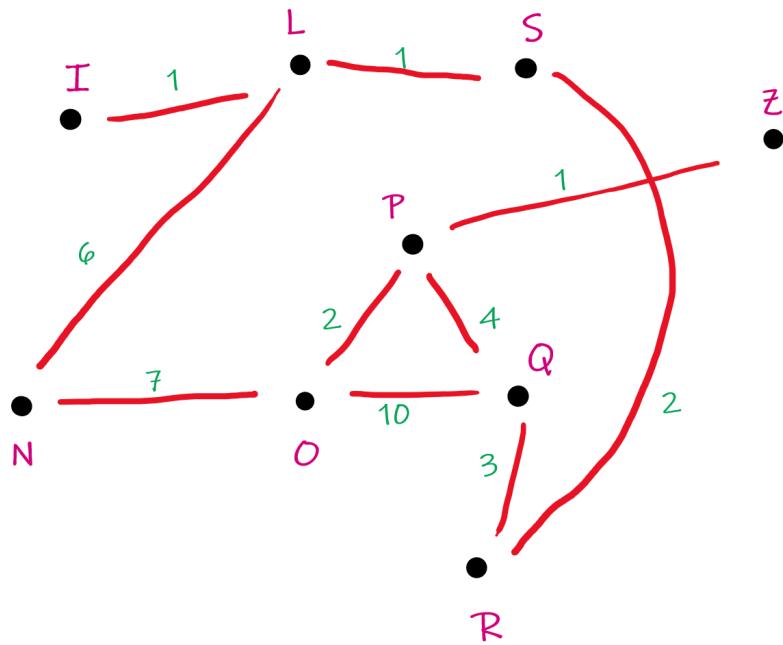
# Algoritmo di Dijkstra

Fino a che  $F$  non è vuoto

1. prendi da  $F$  il nodo  $j$  che ha la distanza più piccola dalla sorgente
2. sposta  $j$  in  $V$
3. considera i vicini di  $j$  non ancora visitati e per ognuno di essi (indichiamolo con  $w$ ) ricalcola  $d_w$  scegliendo il minimo tra  $d_w$  e  $d_j + a_{j,w}$  ( $a_{j,w}$  è il costo dell'arco da  $j$  a  $w$ ). Inserisci  $w$  in  $F$ .
4. se si è aggiornato  $d_w$ , è necessario aggiornare  $p_w \leftarrow j$



# Esempio



$d_I$	0
$d_L$	1
$d_S$	$\infty$
$d_R$	$\infty$
$d_Q$	$\infty$
$d_P$	$\infty$
$d_O$	$\infty$
$d_N$	$\infty$
$d_Z$	$\infty$

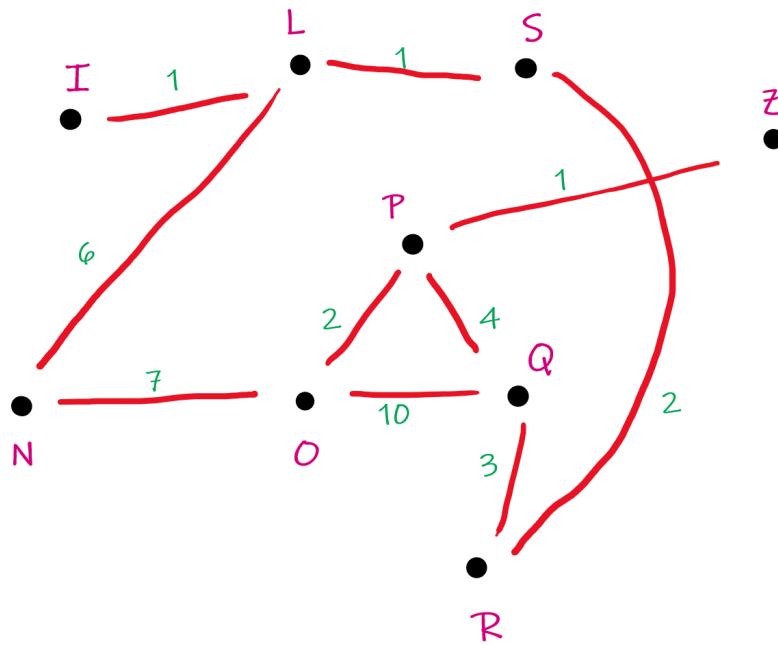
$p_I$	
$p_L$	$p_I$
$p_S$	
$p_R$	
$p_Q$	
$p_P$	
$p_O$	
$p_N$	
$p_Z$	

$$V = \{I\}$$

$$F = \{L\}$$

$$K = \{L, S, R, Q, P, O, N, Z\}$$

# Esempio



$d_I$	0
$d_L$	1
$d_S$	$\infty$
$d_R$	$\infty$
$d_Q$	$\infty$
$d_P$	$\infty$
$d_O$	$\infty$
$d_N$	$\infty$
$d_Z$	$\infty$

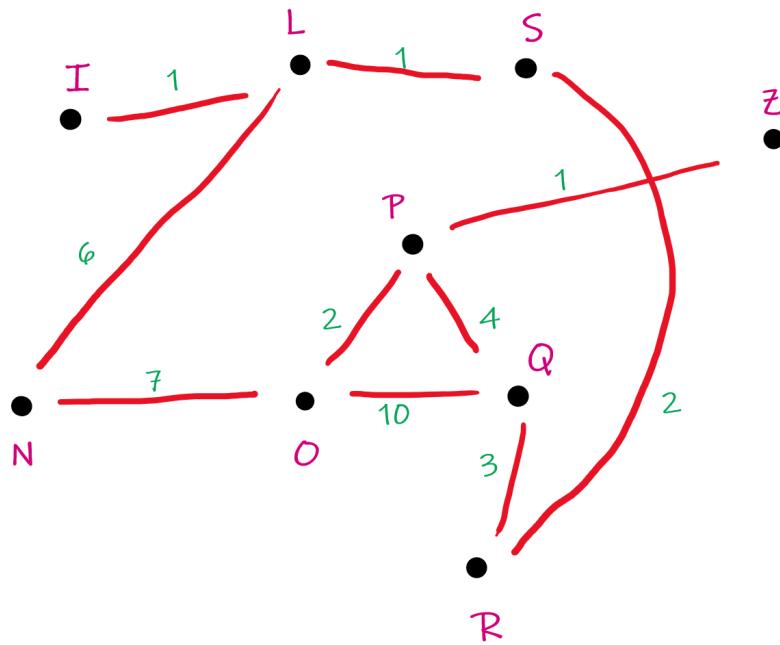
$p_I$	
$p_L$	$p_I$
$p_S$	
$p_R$	
$p_Q$	
$p_P$	
$p_O$	
$p_N$	
$p_Z$	

$$V = \{I, L\}$$

$$F = \{\}$$

$$K = \{S, R, Q, P, O, N, Z\}$$

# Esempio



$d_I$	0
$d_L$	1
$d_S$	$\infty$
$d_R$	$\infty$
$d_Q$	$\infty$
$d_P$	$\infty$
$d_O$	$\infty$
$d_N$	$\infty$
$d_Z$	$\infty$

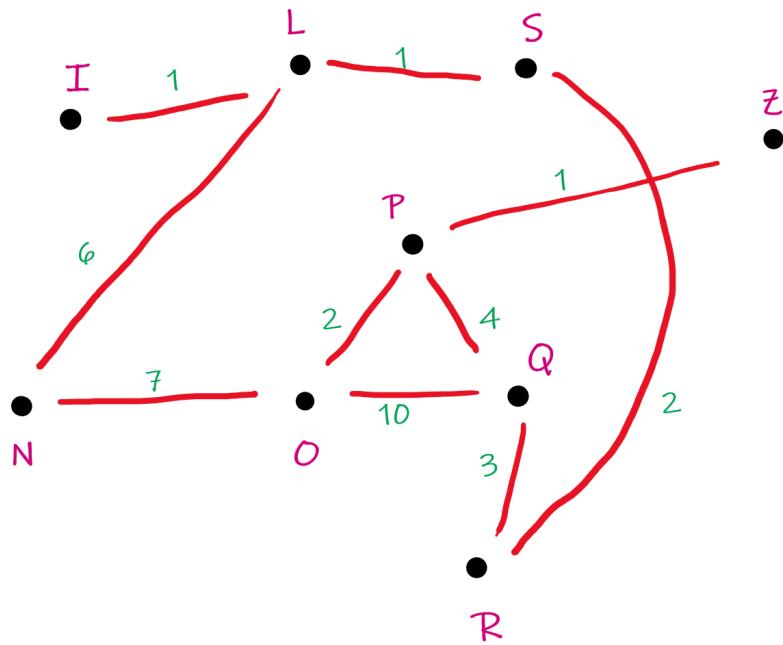
$p_I$	
$p_L$	$p_I$
$p_S$	
$p_R$	
$p_Q$	
$p_P$	
$p_O$	
$p_N$	
$p_Z$	

$$V = \{I, L\}$$

$$F = \{S, N\}$$

$$K = \{S, R, Q, P, O, N, Z\}$$

# Esempio



$d_I$	0
$d_L$	1
$d_S$	$\infty$
$d_R$	$\infty$
$d_Q$	$\infty$
$d_P$	$\infty$
$d_O$	$\infty$
$d_N$	$\infty$
$d_Z$	$\infty$

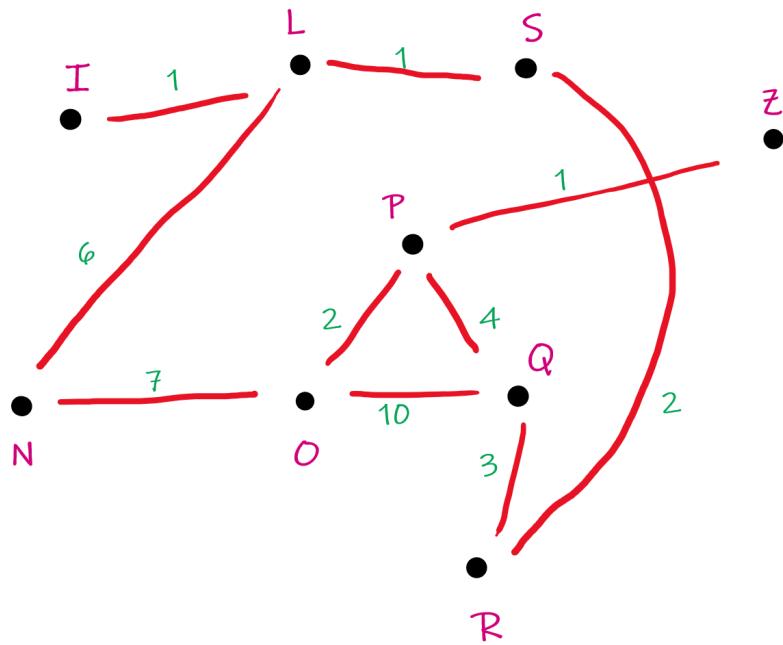
$p_I$	
$p_L$	$p_I$
$p_S$	
$p_R$	
$p_Q$	
$p_P$	
$p_O$	
$p_N$	
$p_Z$	

$$V = \{I, L\}$$

$$F = \{\textcolor{red}{S}, N\}$$

$$K = \{S, R, Q, P, O, N, Z\}$$

# Esempio



$d_I$	0
$d_L$	1
$d_S$	2
$d_R$	$\infty$
$d_Q$	$\infty$
$d_P$	$\infty$
$d_O$	$\infty$
$d_N$	$\infty$
$d_Z$	$\infty$

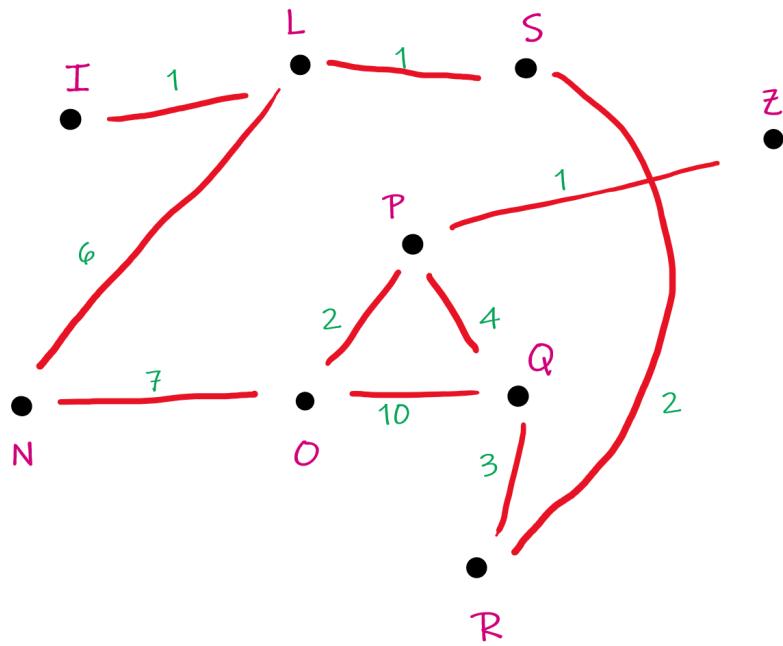
$p_I$	
$p_L$	$p_I$
$p_S$	
$p_R$	
$p_Q$	
$p_P$	
$p_O$	
$p_N$	
$p_Z$	

$$V = \{I, L\}$$

$$F = \{\textcolor{red}{S}, N\}$$

$$K = \{S, R, Q, P, O, N, Z\}$$

# Esempio



$d_I$	0
$d_L$	1
$d_S$	2
$d_R$	$\infty$
$d_Q$	$\infty$
$d_P$	$\infty$
$d_O$	$\infty$
$d_N$	$\infty$
$d_Z$	$\infty$

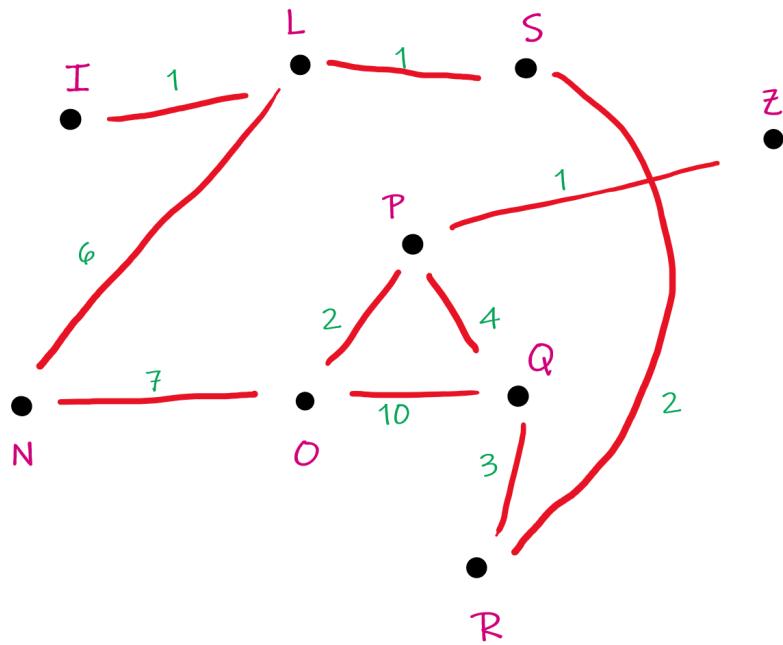
$p_I$	
$p_L$	$p_I$
$p_S$	$p_L$
$p_R$	
$p_Q$	
$p_P$	
$p_O$	
$p_N$	
$p_Z$	

$$V = \{I, L\}$$

$$F = \{S, N\}$$

$$K = \{S, R, Q, P, O, N, Z\}$$

# Esempio



$d_I$	0
$d_L$	1
$d_S$	2
$d_R$	$\infty$
$d_Q$	$\infty$
$d_P$	$\infty$
$d_O$	$\infty$
$d_N$	$\infty$
$d_Z$	$\infty$

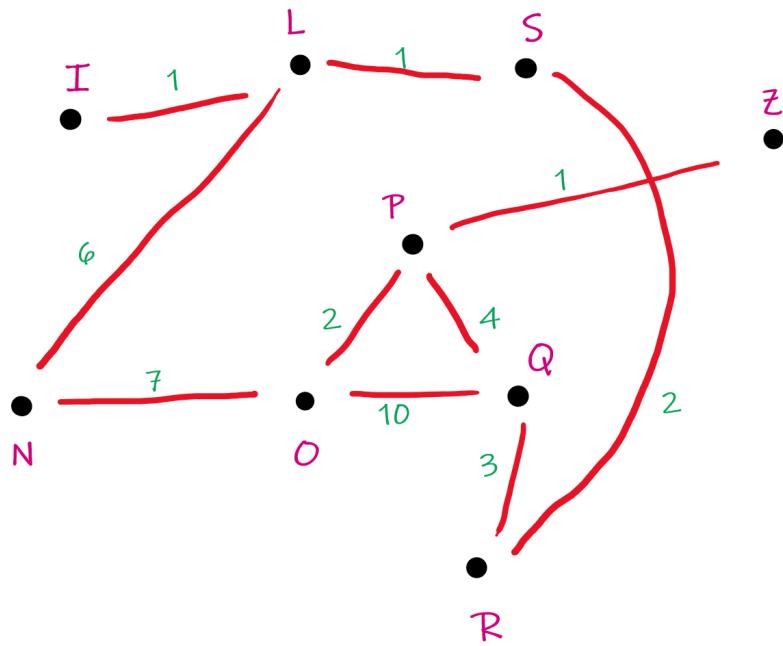
$p_I$	
$p_L$	$p_I$
$p_S$	$p_L$
$p_R$	
$p_Q$	
$p_P$	
$p_O$	
$p_N$	
$p_Z$	

$$V = \{I, L\}$$

$$F = \{S, \textcolor{red}{N}\}$$

$$K = \{S, R, Q, P, O, N, Z\}$$

# Esempio



$d_I$	0
$d_L$	1
$d_S$	2
$d_R$	$\infty$
$d_Q$	$\infty$
$d_P$	$\infty$
$d_O$	$\infty$
$d_N$	7
$d_Z$	$\infty$

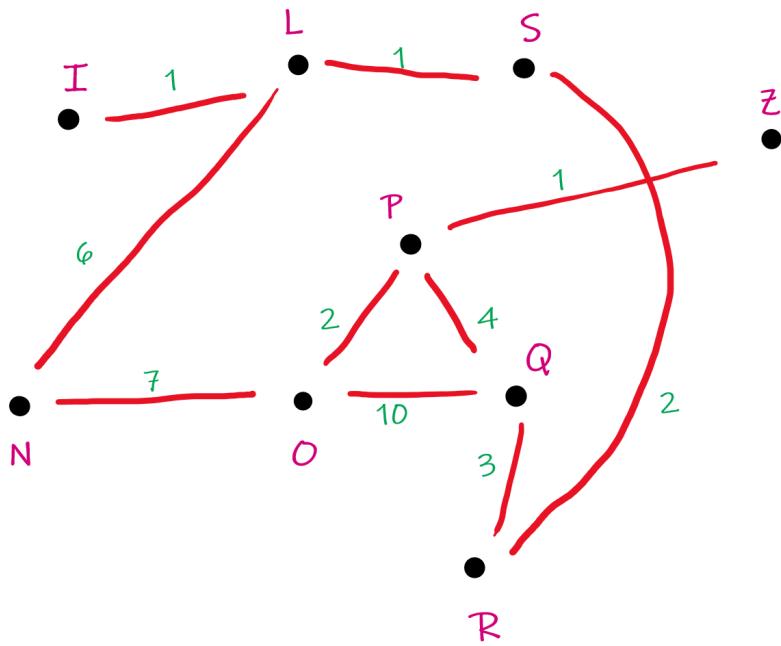
$p_I$	
$p_L$	$p_I$
$p_S$	$p_L$
$p_R$	
$p_Q$	
$p_P$	
$p_O$	
$p_N$	
$p_Z$	

$$V = \{I, L\}$$

$$F = \{S, \textcolor{red}{N}\}$$

$$K = \{S, R, Q, P, O, N, Z\}$$

# Esempio



$d_I$	0
$d_L$	1
$d_S$	2
$d_R$	$\infty$
$d_Q$	$\infty$
$d_P$	$\infty$
$d_O$	$\infty$
$d_N$	7
$d_Z$	$\infty$

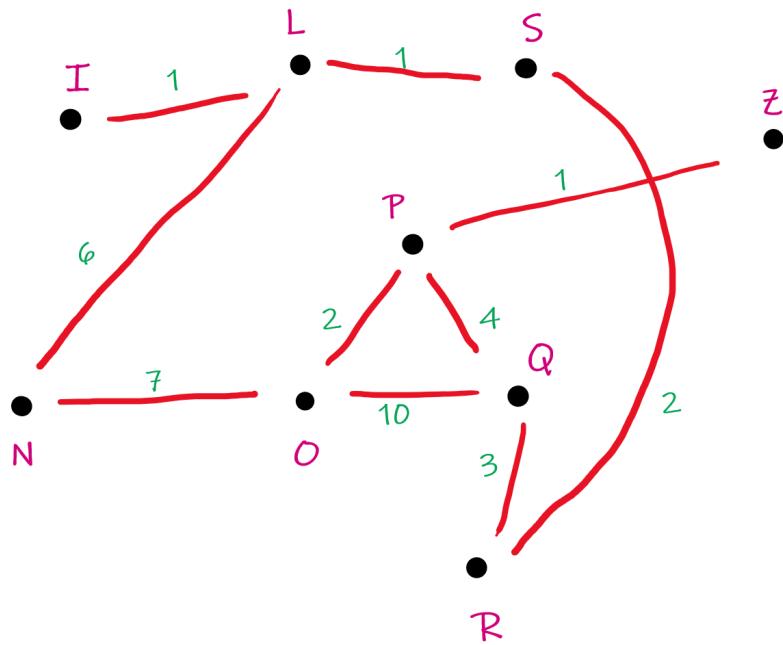
$p_I$	
$p_L$	$p_I$
$p_S$	$p_L$
$p_R$	
$p_Q$	
$p_P$	
$p_O$	
$p_N$	$p_L$
$p_Z$	

$$V = \{I, L\}$$

$$F = \{S, \textcolor{red}{N}\}$$

$$K = \{S, R, Q, P, O, N, Z\}$$

# Esempio



$d_I$	0
$d_L$	1
$d_S$	2
$d_R$	$\infty$
$d_Q$	$\infty$
$d_P$	$\infty$
$d_O$	$\infty$
$d_N$	7
$d_Z$	$\infty$

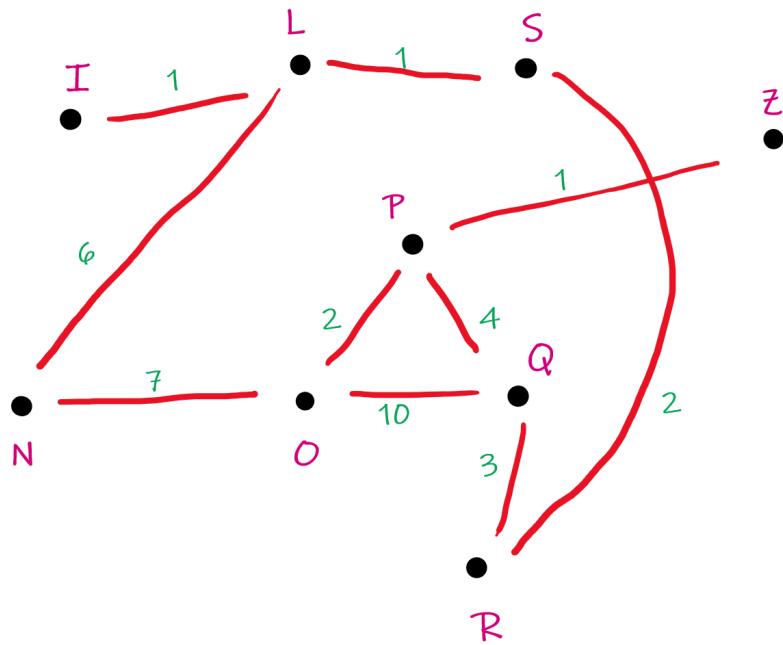
$p_I$	
$p_L$	$p_I$
$p_S$	$p_L$
$p_R$	
$p_Q$	
$p_P$	
$p_O$	
$p_N$	$p_L$
$p_Z$	

$$V = \{I, L\}$$

$$F = \{S, N\}$$

$$K = \{S, R, Q, P, O, N, Z\}$$

# Esempio



$d_I$	0
$d_L$	1
$d_S$	2
$d_R$	$\infty$
$d_Q$	$\infty$
$d_P$	$\infty$
$d_O$	$\infty$
$d_N$	7
$d_Z$	$\infty$

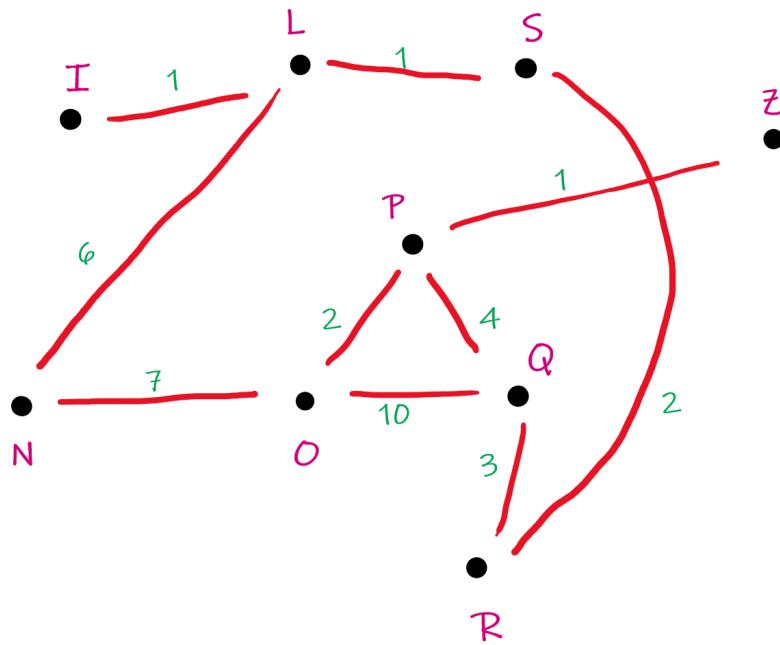
$p_I$	
$p_L$	$p_I$
$p_S$	$p_L$
$p_R$	
$p_Q$	
$p_P$	
$p_O$	
$p_N$	$p_L$
$p_Z$	

$$V = \{I, L\}$$

$$F = \{\textcolor{red}{S}, N\}$$

$$K = \{S, R, Q, P, O, N, Z\}$$

# Esempio



$d_I$	0
$d_L$	1
$d_S$	2
$d_R$	$\infty$
$d_Q$	$\infty$
$d_P$	$\infty$
$d_O$	$\infty$
$d_N$	7
$d_Z$	$\infty$

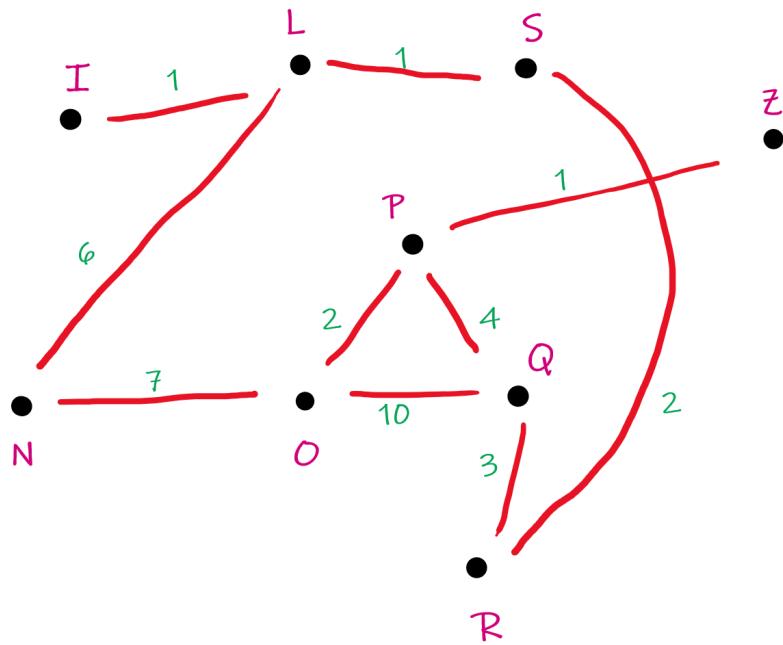
$p_I$	
$p_L$	$p_I$
$p_S$	$p_L$
$p_R$	
$p_Q$	
$p_P$	
$p_O$	
$p_N$	$p_L$
$p_Z$	

$$V = \{I, L, S\}$$

$$F = \{N\}$$

$$K = \{R, Q, P, O, N, Z\}$$

# Esempio



$d_I$	0
$d_L$	1
$d_S$	2
$d_R$	$\infty$
$d_Q$	$\infty$
$d_P$	$\infty$
$d_O$	$\infty$
$d_N$	7
$d_Z$	$\infty$

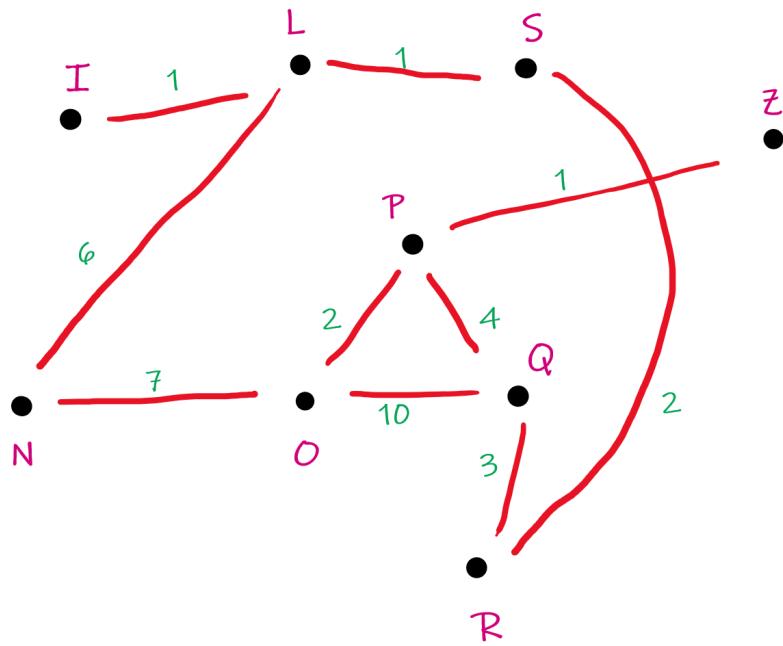
$p_I$	
$p_L$	$p_I$
$p_S$	$p_L$
$p_R$	
$p_Q$	
$p_P$	
$p_O$	
$p_N$	$p_L$
$p_Z$	

$$V = \{I, L, S\}$$

$$F = \{N, \textcolor{red}{R}\}$$

$$K = \{R, Q, P, O, N, Z\}$$

# Esempio



$d_I$	0
$d_L$	1
$d_S$	2
$d_R$	4
$d_Q$	$\infty$
$d_P$	$\infty$
$d_O$	$\infty$
$d_N$	7
$d_Z$	$\infty$

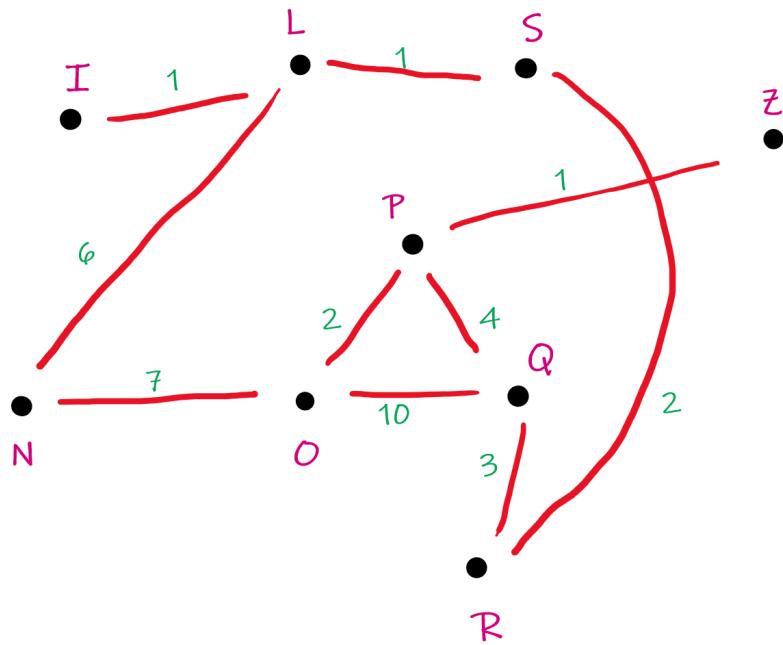
$p_I$	
$p_L$	$p_I$
$p_S$	$p_L$
$p_R$	$p_S$
$p_Q$	
$p_P$	
$p_O$	
$p_N$	$p_L$
$p_Z$	

$$V = \{I, L, S\}$$

$$F = \{N, \textcolor{red}{R}\}$$

$$K = \{R, Q, P, O, N, Z\}$$

# Esempio



$d_I$	0
$d_L$	1
$d_S$	2
$d_R$	4
$d_Q$	$\infty$
$d_P$	$\infty$
$d_O$	$\infty$
$d_N$	7
$d_Z$	$\infty$

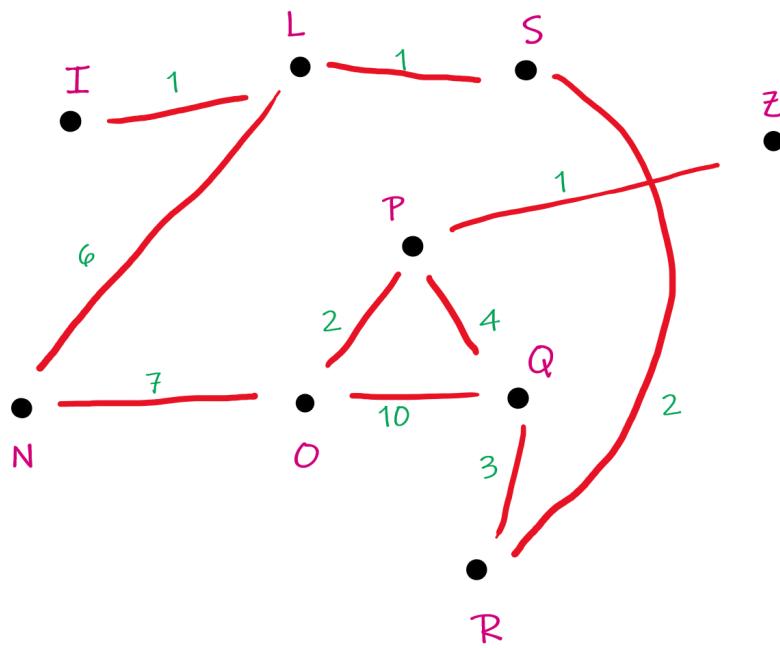
$p_I$	
$p_L$	$p_I$
$p_S$	$p_L$
$p_R$	$p_S$
$p_Q$	
$p_P$	
$p_O$	
$p_N$	$p_L$
$p_Z$	

$$V = \{I, L, S\}$$

$$F = \{N, R\}$$

$$K = \{R, Q, P, O, N, Z\}$$

# Esempio



$d_I$	0
$d_L$	1
$d_S$	2
$d_R$	4
$d_Q$	$\infty$
$d_P$	$\infty$
$d_O$	$\infty$
$d_N$	7
$d_Z$	$\infty$

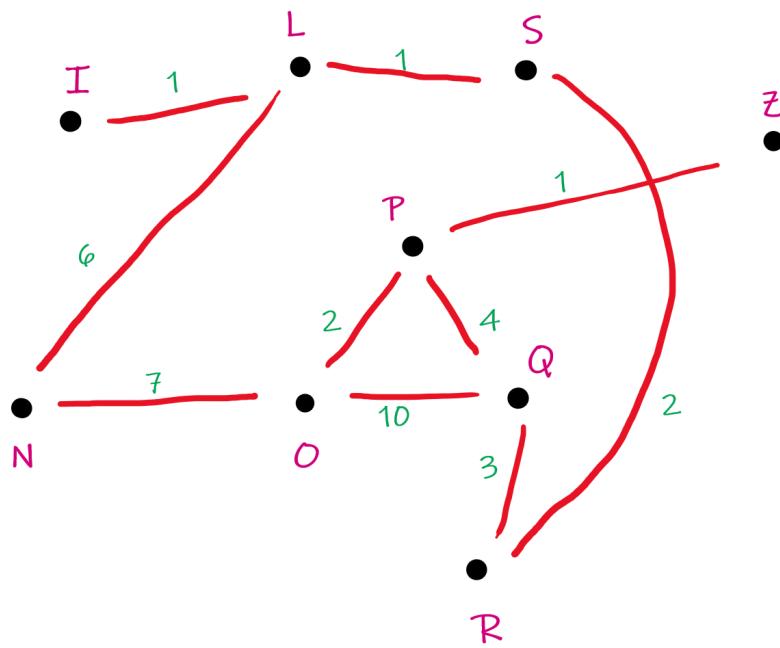
$p_I$	
$p_L$	$p_I$
$p_S$	$p_L$
$p_R$	$p_S$
$p_Q$	
$p_P$	
$p_O$	
$p_N$	$p_L$
$p_Z$	

$$V = \{I, L, S\}$$

$$F = \{N, \textcolor{red}{R}\}$$

$$K = \{R, Q, P, O, N, Z\}$$

# Esempio



$d_I$	0
$d_L$	1
$d_S$	2
$d_R$	4
$d_Q$	$\infty$
$d_P$	$\infty$
$d_O$	$\infty$
$d_N$	7
$d_Z$	$\infty$

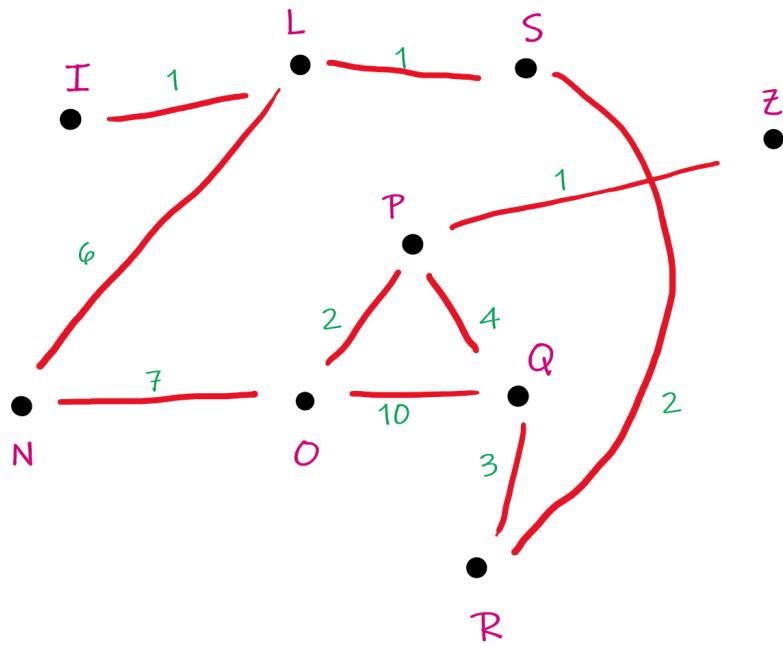
$p_I$	
$p_L$	$p_I$
$p_S$	$p_L$
$p_R$	$p_S$
$p_Q$	
$p_P$	
$p_O$	
$p_N$	$p_L$
$p_Z$	

$$V = \{I, L, S, \textcolor{red}{R}\}$$

$$F = \{N\}$$

$$K = \{Q, P, O, N, Z\}$$

# Esempio



$d_I$	0
$d_L$	1
$d_S$	2
$d_R$	4
$d_Q$	$\infty$
$d_P$	$\infty$
$d_O$	$\infty$
$d_N$	7
$d_Z$	$\infty$

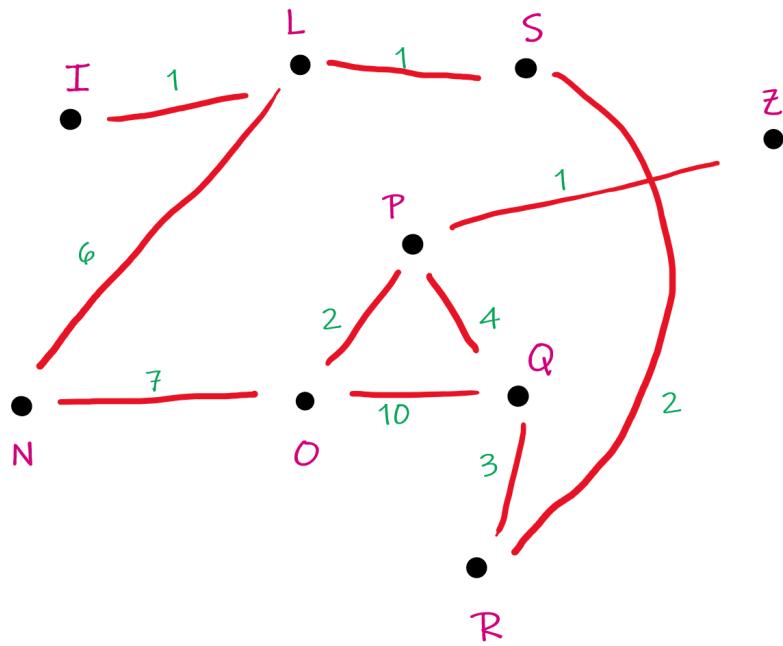
$p_I$	
$p_L$	$p_I$
$p_S$	$p_L$
$p_R$	$p_S$
$p_Q$	
$p_P$	
$p_O$	
$p_N$	$p_L$
$p_Z$	

$$V = \{I, L, S, \textcolor{red}{R}\}$$

$$F = \{N, \textcolor{red}{Q}\}$$

$$K = \{Q, P, O, N, Z\}$$

# Esempio



$d_I$	0
$d_L$	1
$d_S$	2
$d_R$	4
$d_Q$	7
$d_P$	$\infty$
$d_O$	$\infty$
$d_N$	7
$d_Z$	$\infty$

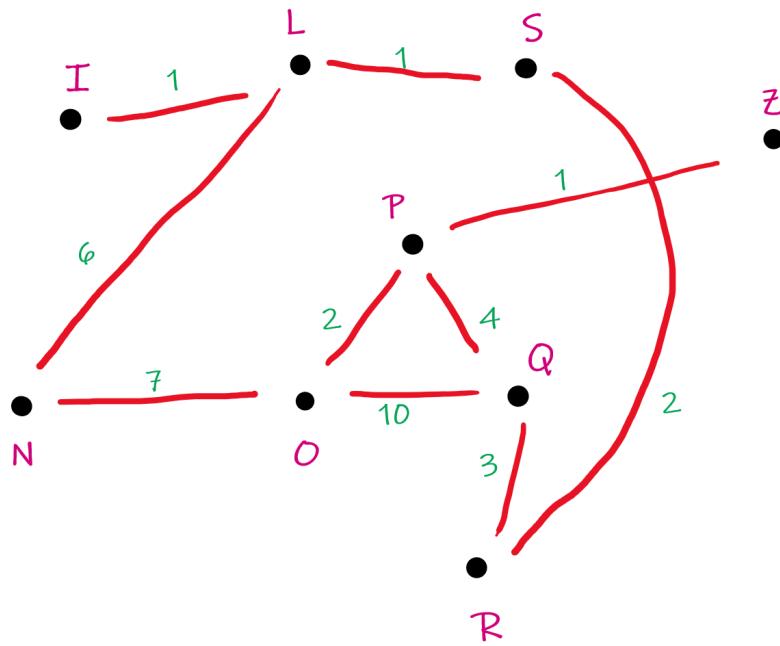
$p_I$	
$p_L$	$p_I$
$p_S$	$p_L$
$p_R$	$p_S$
$p_Q$	$p_R$
$p_P$	
$p_O$	
$p_N$	$p_L$
$p_Z$	

$$V = \{I, L, S, \textcolor{red}{R}\}$$

$$F = \{N, \textcolor{red}{Q}\}$$

$$K = \{Q, P, O, N, Z\}$$

# Esempio



$d_I$	0
$d_L$	1
$d_S$	2
$d_R$	4
$d_Q$	7
$d_P$	$\infty$
$d_O$	$\infty$
$d_N$	7
$d_Z$	$\infty$

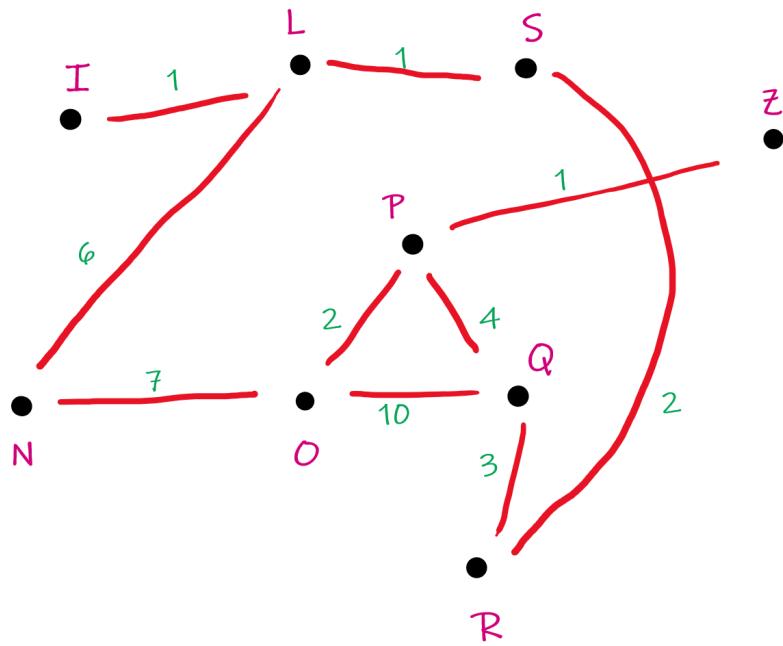
$p_I$	
$p_L$	$p_I$
$p_S$	$p_L$
$p_R$	$p_S$
$p_Q$	$p_R$
$p_P$	
$p_O$	
$p_N$	$p_L$
$p_Z$	

$$V = \{I, L, S, R\}$$

$$F = \{N, Q\}$$

$$K = \{Q, P, O, N, Z\}$$

# Esempio



$d_I$	0
$d_L$	1
$d_S$	2
$d_R$	4
$d_Q$	7
$d_P$	$\infty$
$d_O$	$\infty$
$d_N$	7
$d_Z$	$\infty$

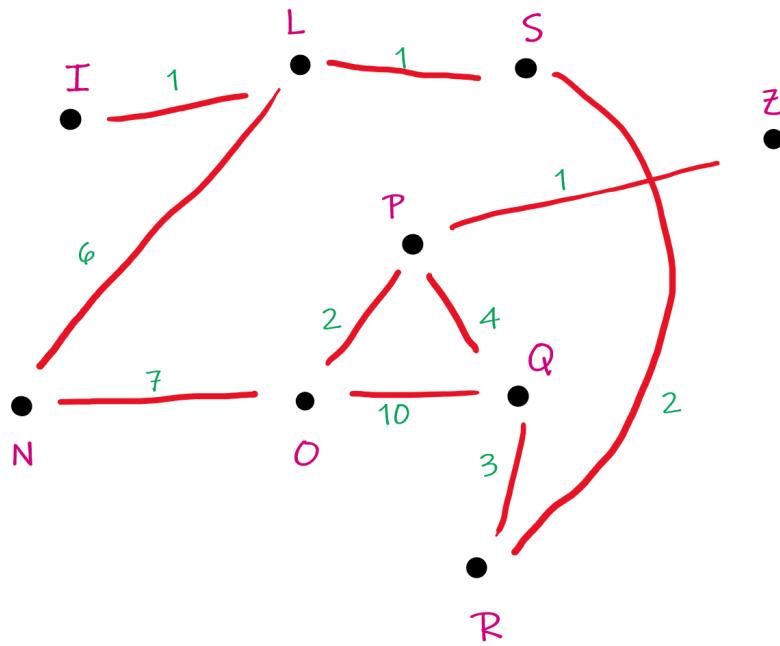
$p_I$	
$p_L$	$p_I$
$p_S$	$p_L$
$p_R$	$p_S$
$p_Q$	$p_R$
$p_P$	
$p_O$	
$p_N$	$p_L$
$p_Z$	

$$V = \{I, L, S, R\}$$

$$F = \{N, Q\}$$

$$K = \{Q, P, O, N, Z\}$$

# Esempio



$d_I$	0
$d_L$	1
$d_S$	2
$d_R$	4
$d_Q$	7
$d_P$	$\infty$
$d_O$	$\infty$
$d_N$	7
$d_Z$	$\infty$

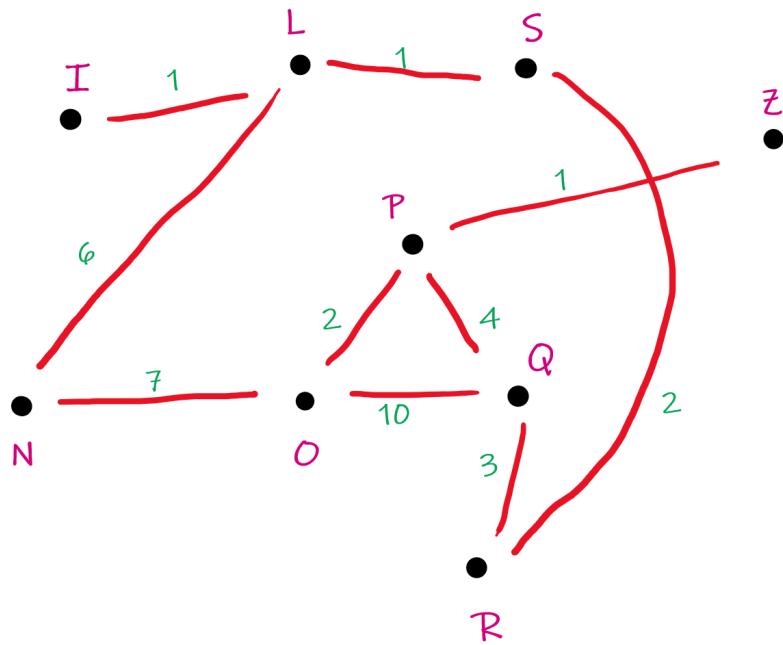
$p_I$	
$p_L$	$p_I$
$p_S$	$p_L$
$p_R$	$p_S$
$p_Q$	$p_R$
$p_P$	
$p_O$	
$p_N$	$p_L$
$p_Z$	

$$V = \{I, L, S, R, \textcolor{red}{N}\}$$

$$F = \{Q\}$$

$$K = \{Q, P, O, Z\}$$

# Esempio



$d_I$	0
$d_L$	1
$d_S$	2
$d_R$	4
$d_Q$	7
$d_P$	$\infty$
$d_O$	$\infty$
$d_N$	7
$d_Z$	$\infty$

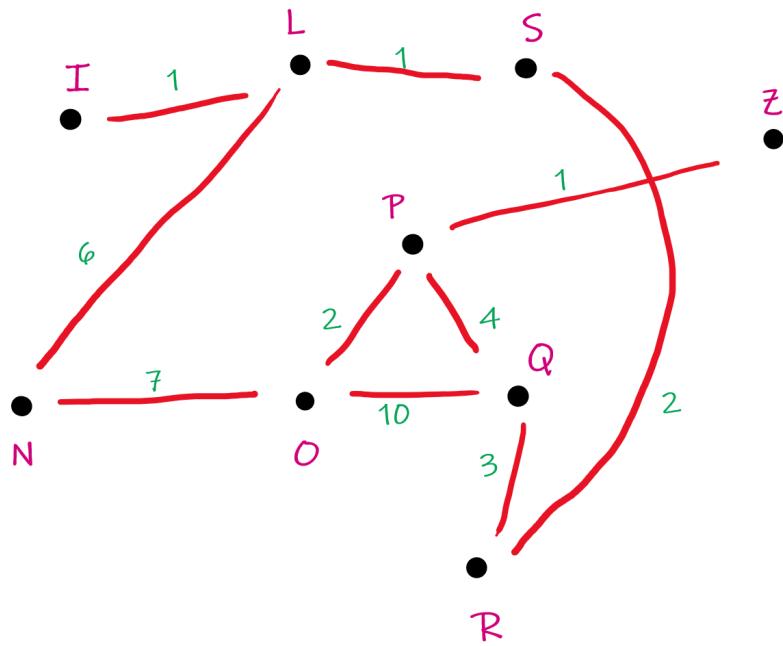
$p_I$	
$p_L$	$p_I$
$p_S$	$p_L$
$p_R$	$p_S$
$p_Q$	$p_R$
$p_P$	
$p_O$	
$p_N$	$p_L$
$p_Z$	

$$V = \{I, L, S, R, \textcolor{red}{N}\}$$

$$F = \{Q, \textcolor{red}{O}\}$$

$$K = \{Q, P, O, Z\}$$

# Esempio



$d_I$	0
$d_L$	1
$d_S$	2
$d_R$	4
$d_Q$	7
$d_P$	$\infty$
$d_O$	14
$d_N$	7
$d_Z$	$\infty$

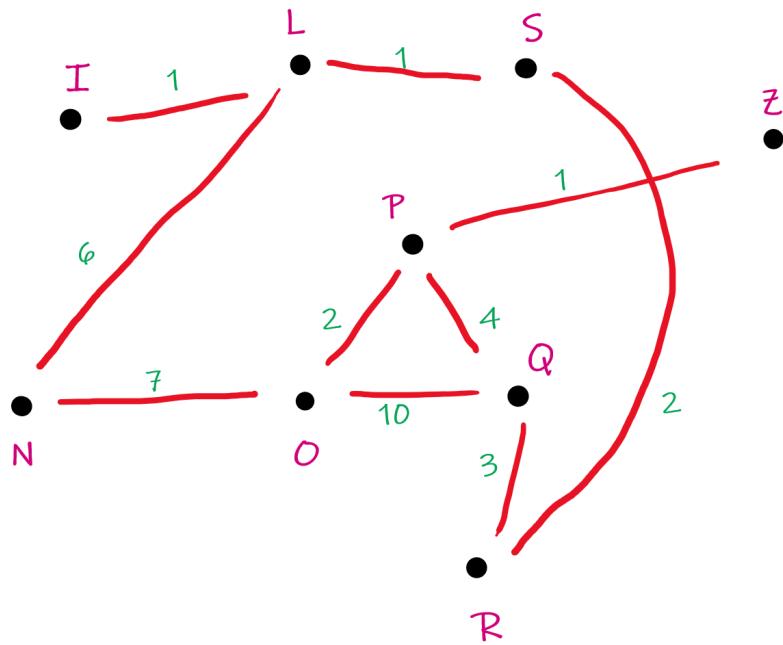
$p_I$	
$p_L$	$p_I$
$p_S$	$p_L$
$p_R$	$p_S$
$p_Q$	$p_R$
$p_P$	
$p_O$	$p_N$
$p_N$	$p_L$
$p_Z$	

$$V = \{I, L, S, R, \textcolor{red}{N}\}$$

$$F = \{Q, \textcolor{red}{O}\}$$

$$K = \{Q, P, O, Z\}$$

# Esempio



$d_I$	0
$d_L$	1
$d_S$	2
$d_R$	4
$d_Q$	7
$d_P$	$\infty$
$d_O$	14
$d_N$	7
$d_Z$	$\infty$

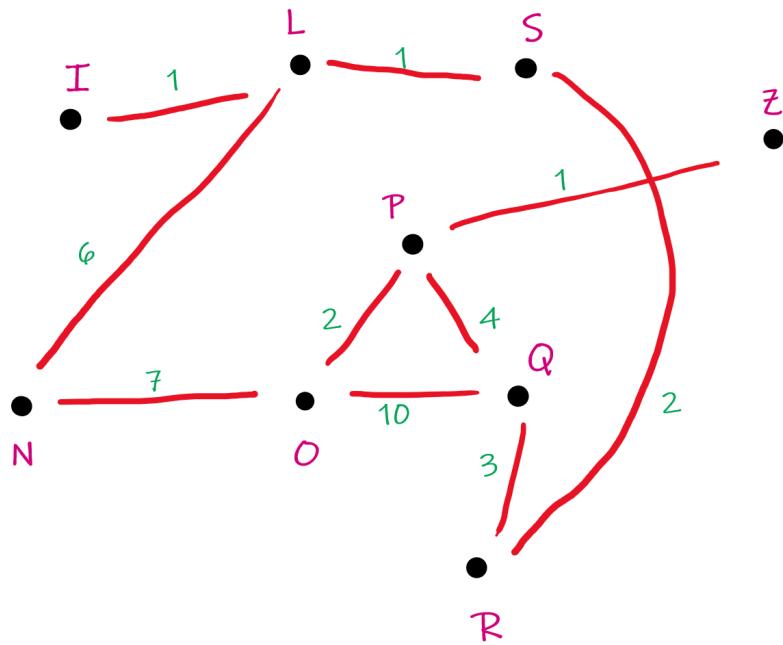
$p_I$	
$p_L$	$p_I$
$p_S$	$p_L$
$p_R$	$p_S$
$p_Q$	$p_R$
$p_P$	
$p_O$	$p_N$
$p_N$	$p_L$
$p_Z$	

$$V = \{I, L, S, R, N\}$$

$$F = \{Q, O\}$$

$$K = \{Q, P, O, Z\}$$

# Esempio



$d_I$	0
$d_L$	1
$d_S$	2
$d_R$	4
$d_Q$	7
$d_P$	$\infty$
$d_O$	14
$d_N$	7
$d_Z$	$\infty$

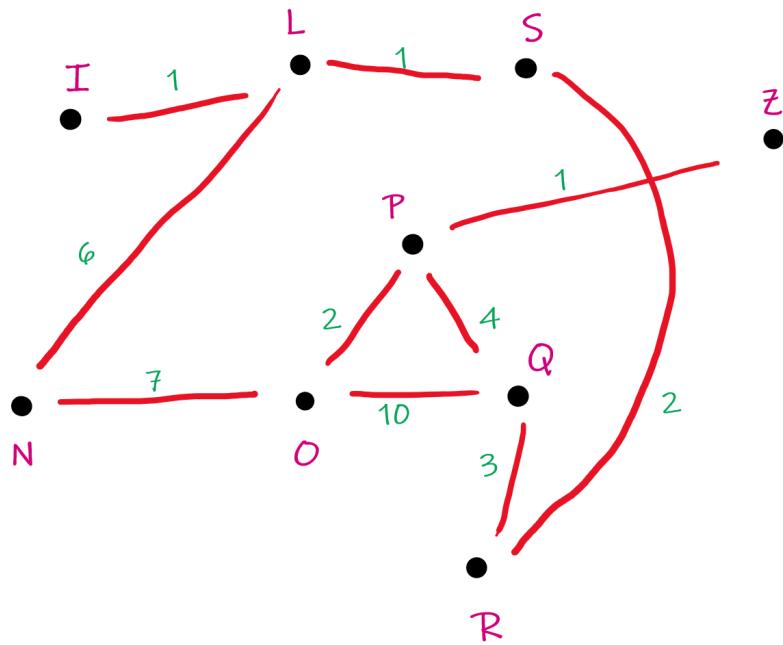
$p_I$	
$p_L$	$p_I$
$p_S$	$p_L$
$p_R$	$p_S$
$p_Q$	$p_R$
$p_P$	
$p_O$	$p_N$
$p_N$	$p_L$
$p_Z$	

$$V = \{I, L, S, R, N\}$$

$$F = \{Q, O\}$$

$$K = \{Q, P, O, Z\}$$

# Esempio



$d_I$	0
$d_L$	1
$d_S$	2
$d_R$	4
$d_Q$	7
$d_P$	$\infty$
$d_O$	14
$d_N$	7
$d_Z$	$\infty$

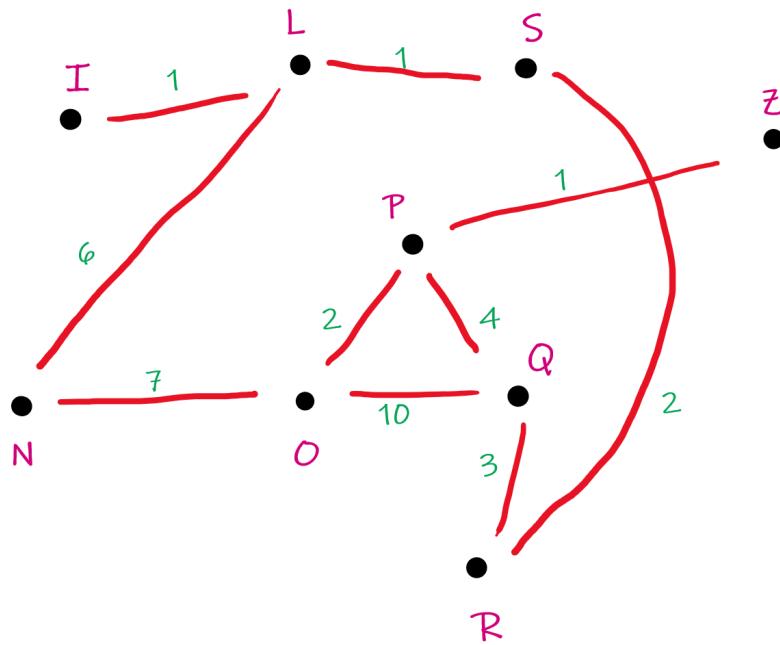
$p_I$	
$p_L$	$p_I$
$p_S$	$p_L$
$p_R$	$p_S$
$p_Q$	$p_R$
$p_P$	
$p_O$	$p_N$
$p_N$	$p_L$
$p_Z$	

$$V = \{I, L, S, R, N, Q\}$$

$$F = \{O\}$$

$$K = \{P, O, Z\}$$

# Esempio



$d_I$	0
$d_L$	1
$d_S$	2
$d_R$	4
$d_Q$	7
$d_P$	$\infty$
$d_O$	14
$d_N$	7
$d_Z$	$\infty$

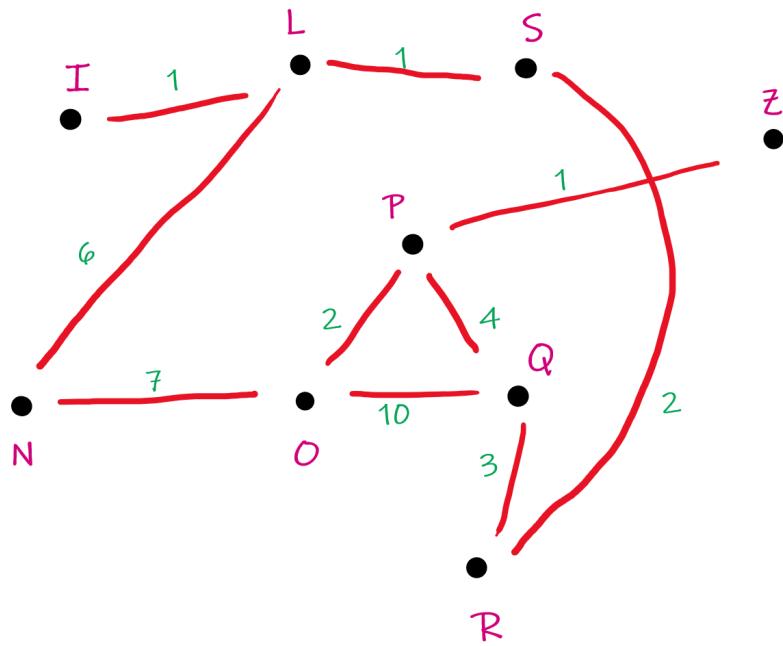
$p_I$	
$p_L$	$p_I$
$p_S$	$p_L$
$p_R$	$p_S$
$p_Q$	$p_R$
$p_P$	
$p_O$	$p_N$
$p_N$	$p_L$
$p_Z$	

$$V = \{I, L, S, R, N, \textcolor{red}{Q}\}$$

$$F = \{\textcolor{red}{O}, \textcolor{red}{P}\}$$

$$K = \{P, O, Z\}$$

# Esempio



$d_I$	0
$d_L$	1
$d_S$	2
$d_R$	4
$d_Q$	7
$d_P$	$\infty$
$d_O$	14
$d_N$	7
$d_Z$	$\infty$

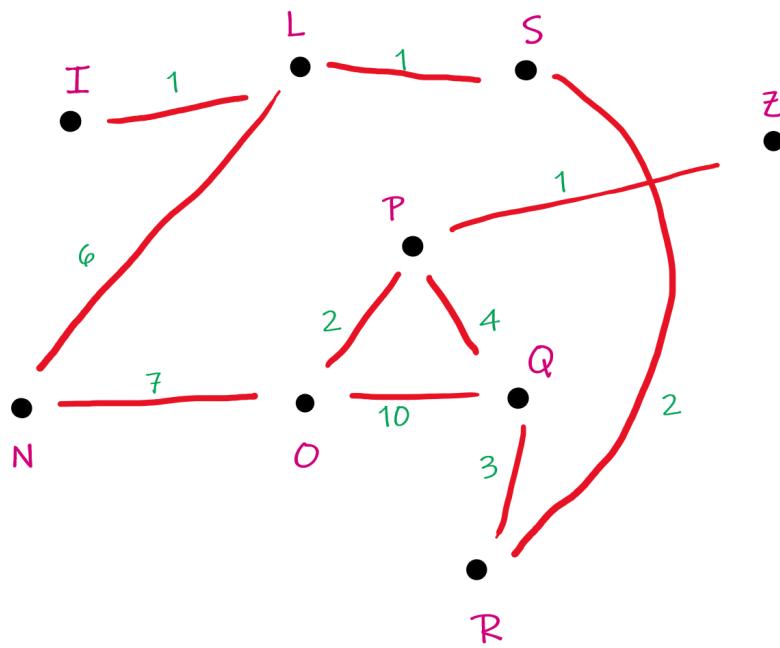
$p_I$	
$p_L$	$p_I$
$p_S$	$p_L$
$p_R$	$p_S$
$p_Q$	$p_R$
$p_P$	
$p_O$	$p_N$
$p_N$	$p_L$
$p_Z$	

$$V = \{I, L, S, R, N, \textcolor{red}{Q}\}$$

$$F = \{\textcolor{red}{O}, P\}$$

$$K = \{P, O, Z\}$$

# Esempio



$d_I$	0
$d_L$	1
$d_S$	2
$d_R$	4
$d_Q$	7
$d_P$	$\infty$
$d_O$	14
$d_N$	7
$d_Z$	$\infty$

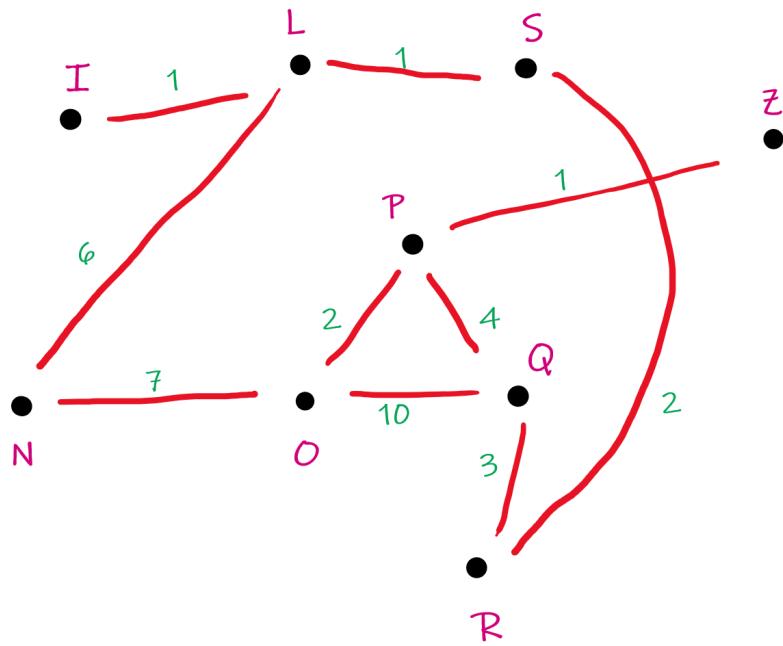
$p_I$	
$p_L$	$p_I$
$p_S$	$p_L$
$p_R$	$p_S$
$p_Q$	$p_R$
$p_P$	
$p_O$	$p_N$
$p_N$	$p_L$
$p_Z$	

$$V = \{I, L, S, R, N, \textcolor{red}{Q}\}$$

$$F = \{\textcolor{red}{O}, P\}$$

$$K = \{P, O, Z\}$$

# Esempio



$d_I$	0
$d_L$	1
$d_S$	2
$d_R$	4
$d_Q$	7
$d_P$	$\infty$
$d_O$	14
$d_N$	7
$d_Z$	$\infty$

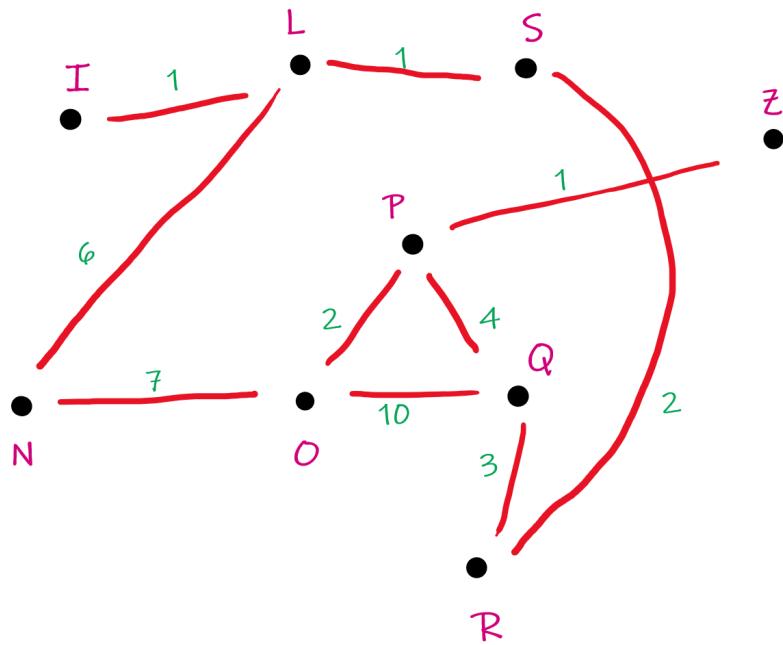
$p_I$	
$p_L$	$p_I$
$p_S$	$p_L$
$p_R$	$p_S$
$p_Q$	$p_R$
$p_P$	
$p_O$	$p_N$
$p_N$	$p_L$
$p_Z$	

$$V = \{I, L, S, R, N, \textcolor{red}{Q}\}$$

$$F = \{\textcolor{red}{O}, P\}$$

$$K = \{P, O, Z\}$$

# Esempio



$d_I$	0
$d_L$	1
$d_S$	2
$d_R$	4
$d_Q$	7
$d_P$	$\infty$
$d_O$	14
$d_N$	7
$d_Z$	$\infty$

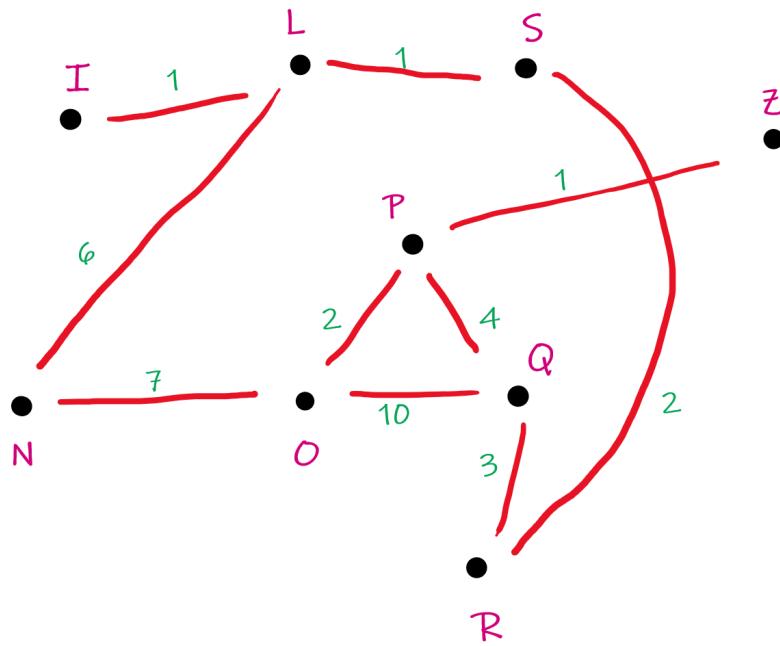
$p_I$	
$p_L$	$p_I$
$p_S$	$p_L$
$p_R$	$p_S$
$p_Q$	$p_R$
$p_P$	
$p_O$	$p_N$
$p_N$	$p_L$
$p_Z$	

$$V = \{I, L, S, R, N, Q\}$$

$$F = \{O, P\}$$

$$K = \{P, O, Z\}$$

# Esempio



$d_I$	0
$d_L$	1
$d_S$	2
$d_R$	4
$d_Q$	7
$d_P$	11
$d_O$	14
$d_N$	7
$d_Z$	$\infty$

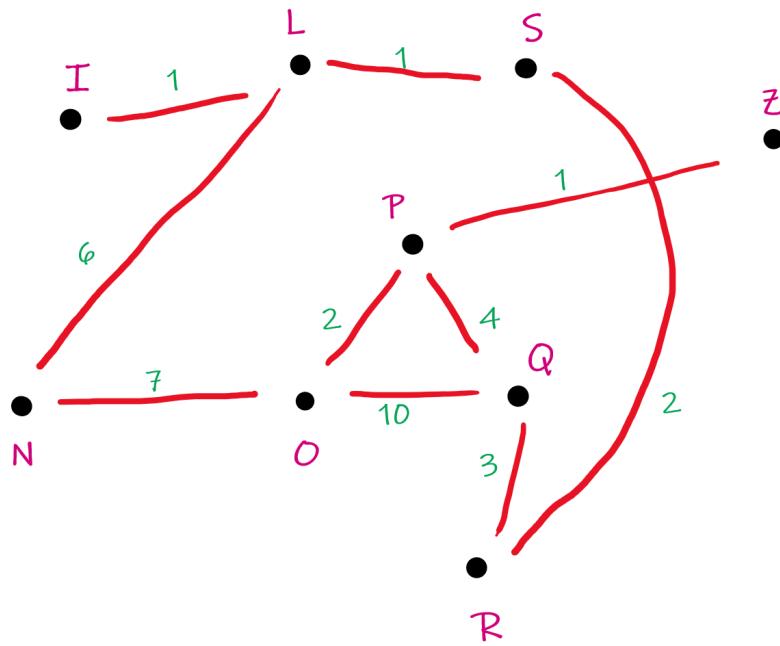
$p_I$	
$p_L$	$p_I$
$p_S$	$p_L$
$p_R$	$p_S$
$p_Q$	$p_R$
$p_P$	$d_Q$
$p_O$	$p_R$
$p_N$	$p_L$
$p_Z$	

$$V = \{I, L, S, R, N, Q\}$$

$$F = \{O, P\}$$

$$K = \{P, O, Z\}$$

# Esempio



$d_I$	0
$d_L$	1
$d_S$	2
$d_R$	4
$d_Q$	7
$d_P$	11
$d_O$	14
$d_N$	7
$d_Z$	$\infty$

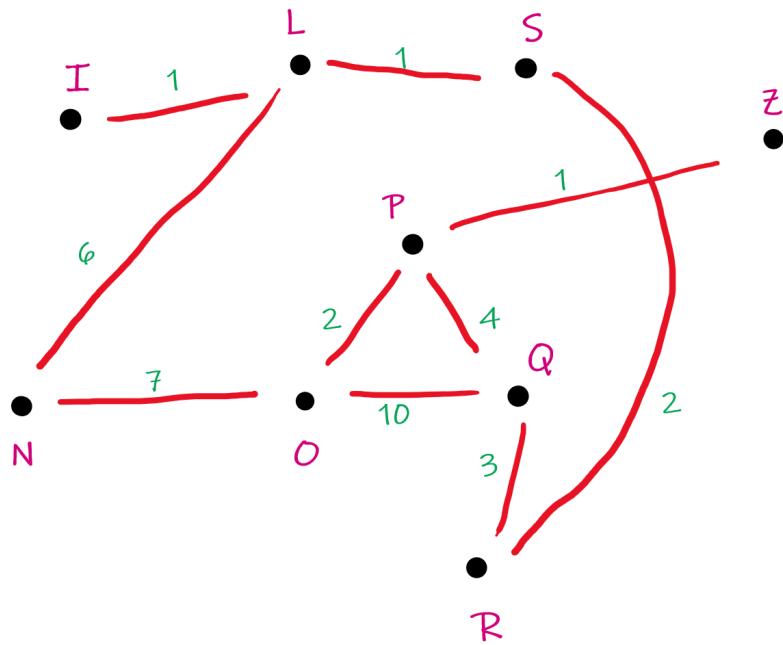
$p_I$	
$p_L$	$p_I$
$p_S$	$p_L$
$p_R$	$p_S$
$p_Q$	$p_R$
$p_P$	$d_Q$
$p_O$	$p_N$
$p_N$	$p_L$
$p_Z$	

$$V = \{I, L, S, R, N, Q\}$$

$$F = \{O, P\}$$

$$K = \{P, O, Z\}$$

# Esempio



$d_I$	0
$d_L$	1
$d_S$	2
$d_R$	4
$d_Q$	7
$d_P$	11
$d_O$	14
$d_N$	7
$d_Z$	$\infty$

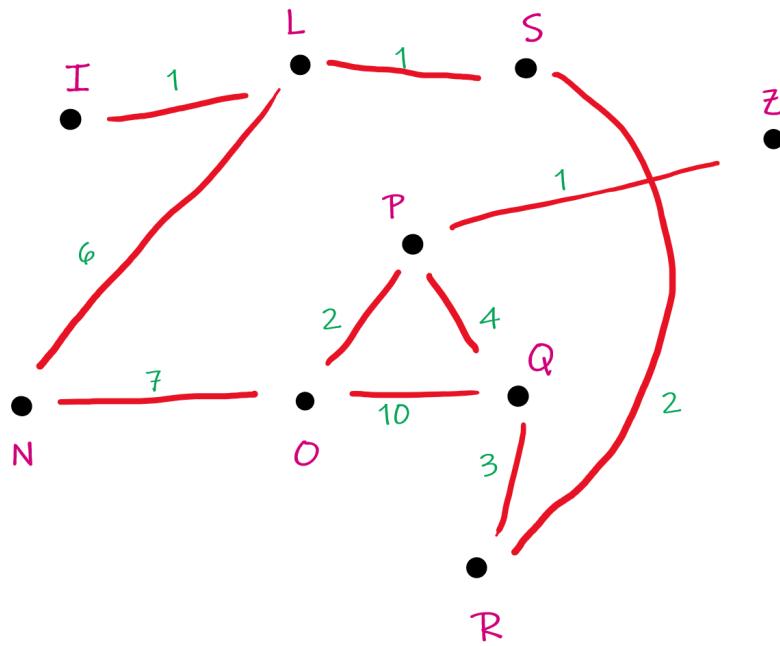
$p_I$	
$p_L$	$p_I$
$p_S$	$p_L$
$p_R$	$p_S$
$p_Q$	$p_R$
$p_P$	$d_Q$
$p_O$	$p_N$
$p_N$	$p_L$
$p_Z$	

$$V = \{I, L, S, R, N, Q\}$$

$$F = \{O, P\}$$

$$K = \{P, O, Z\}$$

# Esempio



$d_I$	0
$d_L$	1
$d_S$	2
$d_R$	4
$d_Q$	7
$d_P$	11
$d_O$	14
$d_N$	7
$d_Z$	$\infty$

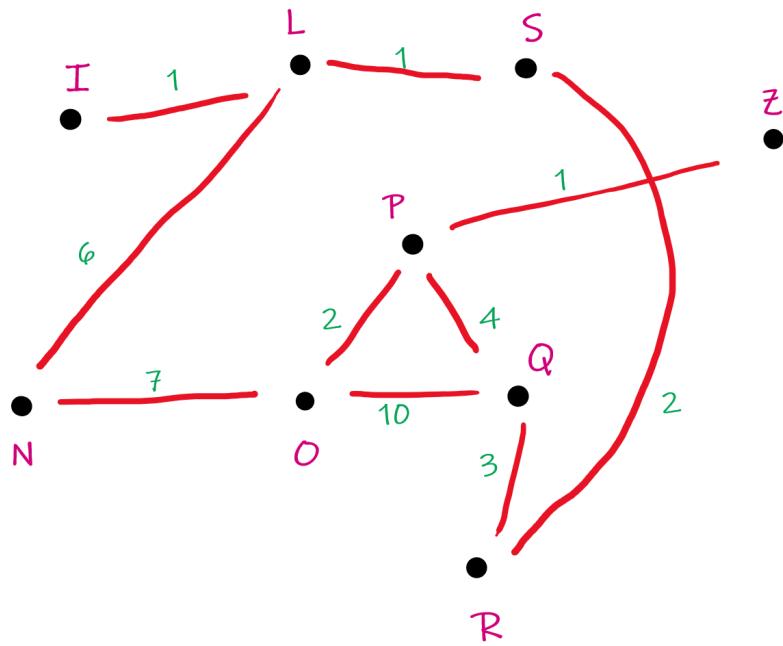
$p_I$	
$p_L$	$p_I$
$p_S$	$p_L$
$p_R$	$p_S$
$p_Q$	$p_R$
$p_P$	$d_Q$
$p_O$	$p_N$
$p_N$	$p_L$
$p_Z$	

$$V = \{I, L, S, R, N, Q, \textcolor{red}{P}\}$$

$$F = \{O\}$$

$$K = \{O, Z\}$$

# Esempio



$d_I$	0
$d_L$	1
$d_S$	2
$d_R$	4
$d_Q$	7
$d_P$	11
$d_O$	14
$d_N$	7
$d_Z$	$\infty$

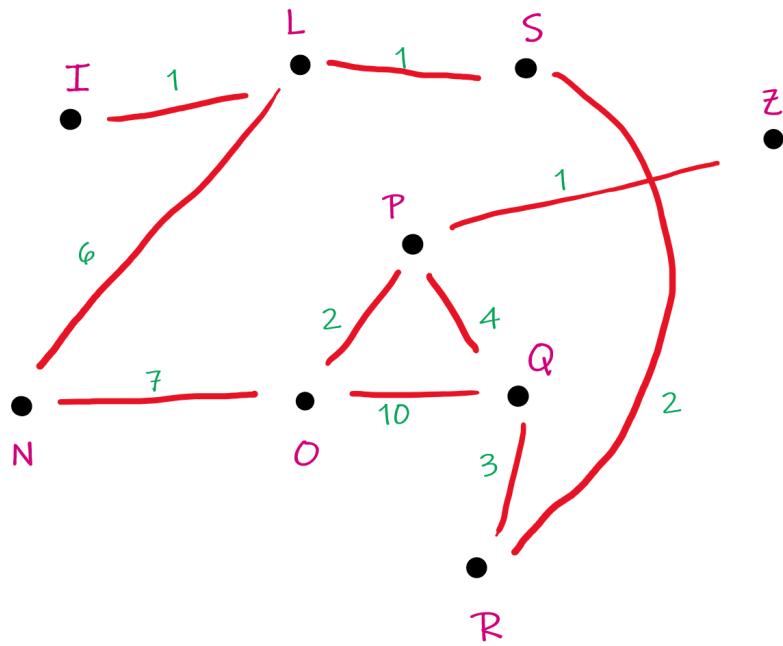
$p_I$	
$p_L$	$p_I$
$p_S$	$p_L$
$p_R$	$p_S$
$p_Q$	$p_R$
$p_P$	$d_Q$
$p_O$	$p_N$
$p_N$	$p_L$
$p_Z$	

$$V = \{I, L, S, R, N, Q, \textcolor{red}{P}\}$$

$$F = \{\textcolor{red}{O}, \textcolor{red}{Z}\}$$

$$K = \{O, Z\}$$

# Esempio



$d_I$	0
$d_L$	1
$d_S$	2
$d_R$	4
$d_Q$	7
$d_P$	11
$d_O$	14
$d_N$	7
$d_Z$	$\infty$

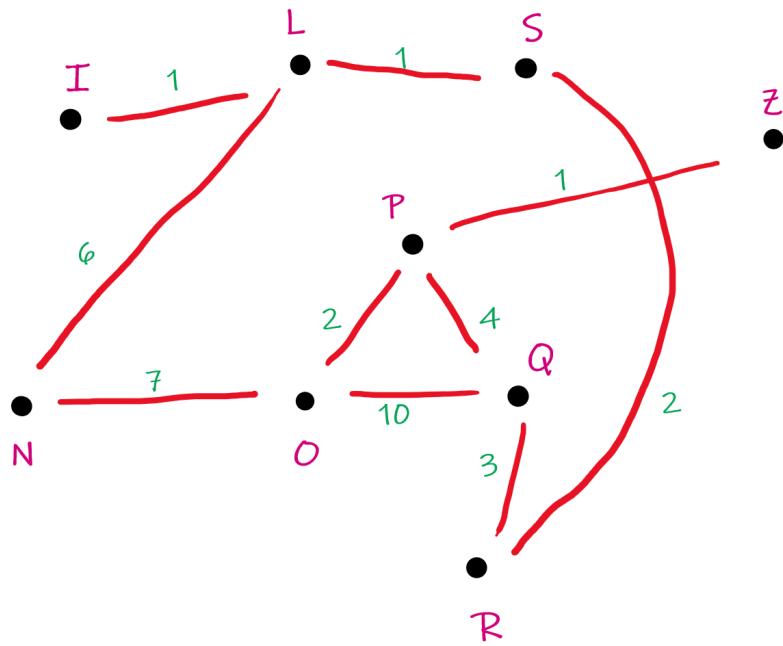
$p_I$	
$p_L$	$p_I$
$p_S$	$p_L$
$p_R$	$p_S$
$p_Q$	$p_R$
$p_P$	$d_Q$
$p_O$	$p_N$
$p_N$	$p_L$
$p_Z$	

$$V = \{I, L, S, R, N, Q, \textcolor{red}{P}\}$$

$$F = \{\textcolor{red}{O}, Z\}$$

$$K = \{O, Z\}$$

# Esempio



$d_I$	0
$d_L$	1
$d_S$	2
$d_R$	4
$d_Q$	7
$d_P$	11
$d_O$	13
$d_N$	7
$d_Z$	$\infty$

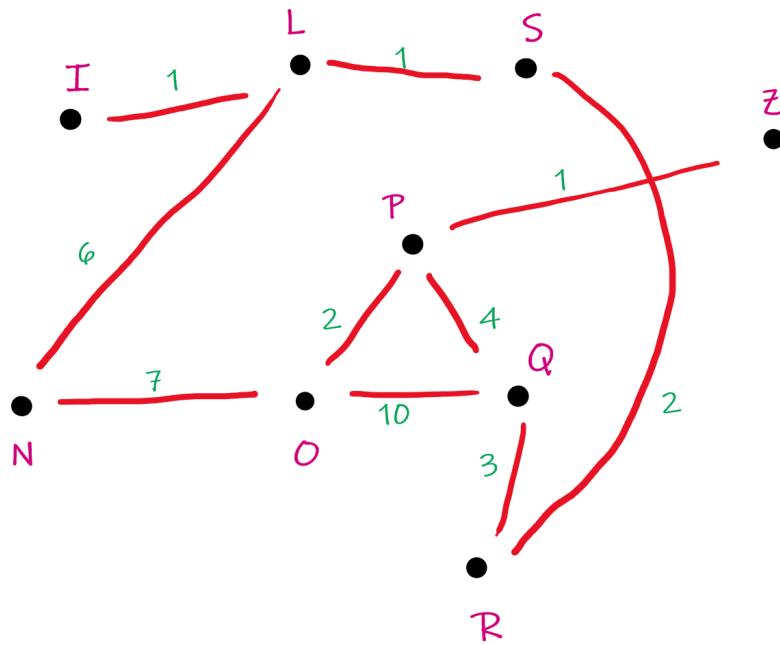
$p_I$	
$p_L$	$p_I$
$p_S$	$p_L$
$p_R$	$p_S$
$p_Q$	$p_R$
$p_P$	$d_Q$
$p_O$	$p_P$
$p_N$	$p_L$
$p_Z$	

$$V = \{I, L, S, R, N, Q, \textcolor{red}{P}\}$$

$$F = \{\textcolor{red}{O}, Z\}$$

$$K = \{O, Z\}$$

# Esempio



$d_I$	0
$d_L$	1
$d_S$	2
$d_R$	4
$d_Q$	7
$d_P$	11
$d_O$	13
$d_N$	7
$d_Z$	$\infty$

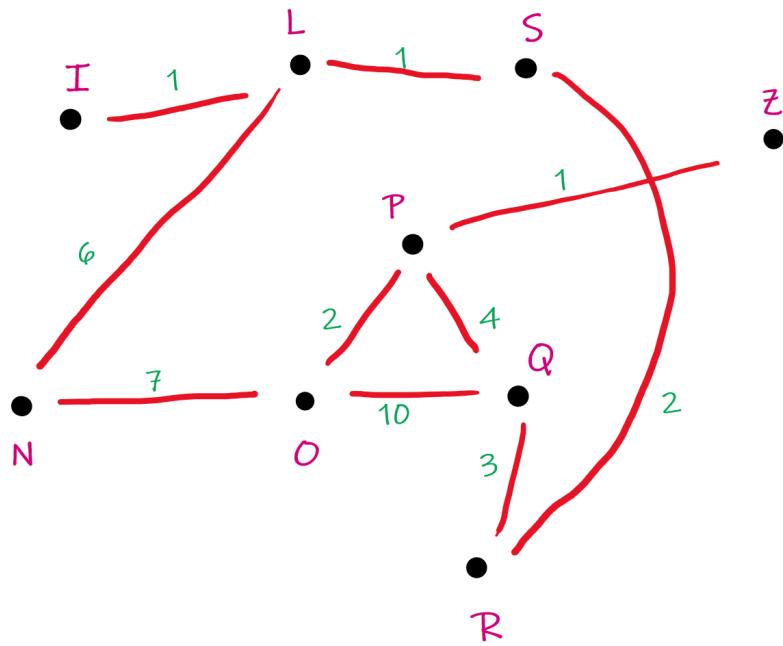
$p_I$	
$p_L$	$p_I$
$p_S$	$p_L$
$p_R$	$p_S$
$p_Q$	$p_R$
$p_P$	$d_Q$
$p_O$	$p_P$
$p_N$	$p_L$
$p_Z$	

$$V = \{I, L, S, R, N, Q, \textcolor{red}{P}\}$$

$$F = \{O, \textcolor{red}{Z}\}$$

$$K = \{O, Z\}$$

# Esempio



$d_I$	0
$d_L$	1
$d_S$	2
$d_R$	4
$d_Q$	7
$d_P$	11
$d_O$	13
$d_N$	7
$d_Z$	12

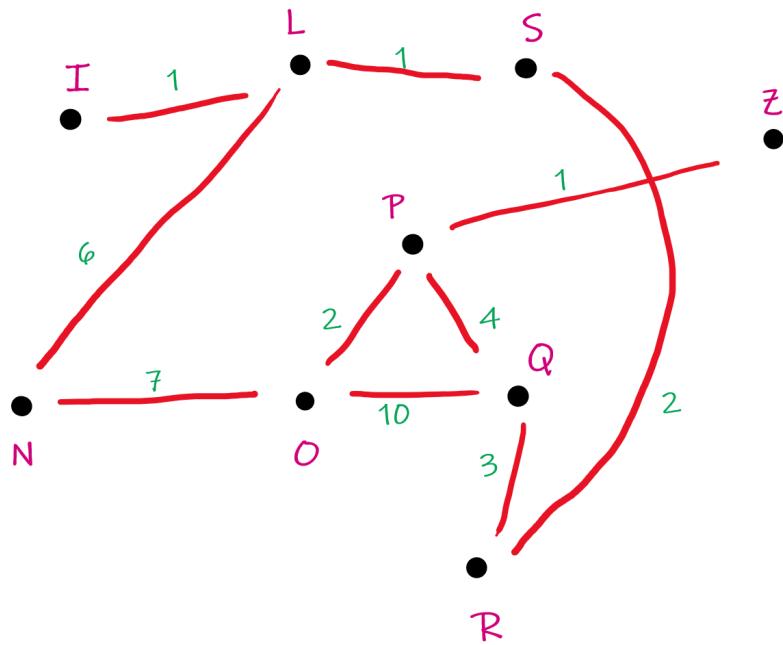
$p_I$	
$p_L$	$p_I$
$p_S$	$p_L$
$p_R$	$p_S$
$p_Q$	$p_R$
$p_P$	$d_Q$
$p_O$	$p_P$
$p_N$	$p_L$
$p_Z$	$p_P$

$$V = \{I, L, S, R, N, Q, \textcolor{red}{P}\}$$

$$F = \{O, \textcolor{red}{Z}\}$$

$$K = \{O, Z\}$$

# Esempio



$d_I$	0
$d_L$	1
$d_S$	2
$d_R$	4
$d_Q$	7
$d_P$	11
$d_O$	13
$d_N$	7
$d_Z$	12

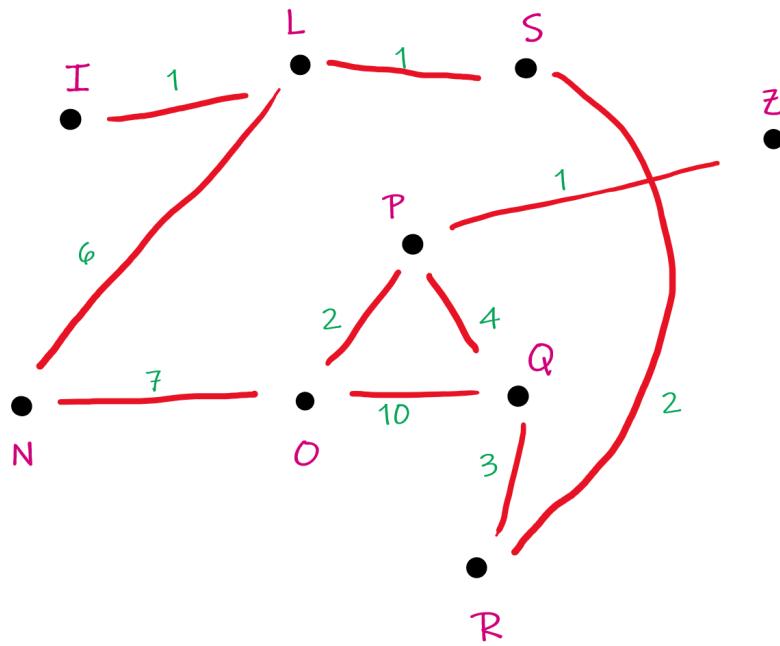
$p_I$	
$p_L$	$p_I$
$p_S$	$p_L$
$p_R$	$p_S$
$p_Q$	$p_R$
$p_P$	$d_Q$
$p_O$	$p_P$
$p_N$	$p_L$
$p_Z$	$p_P$

$$V = \{I, L, S, R, N, Q, P\}$$

$$F = \{O, Z\}$$

$$K = \{O, Z\}$$

# Esempio



$d_I$	0
$d_L$	1
$d_S$	2
$d_R$	4
$d_Q$	7
$d_P$	11
$d_O$	13
$d_N$	7
$d_Z$	12

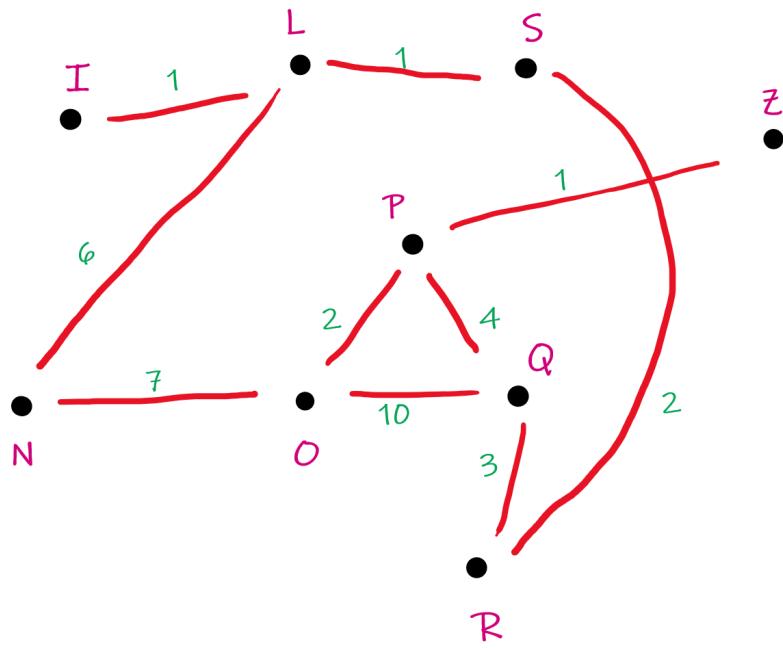
$p_I$	
$p_L$	$p_I$
$p_S$	$p_L$
$p_R$	$p_S$
$p_Q$	$p_R$
$p_P$	$d_Q$
$p_O$	$p_P$
$p_N$	$p_L$
$p_Z$	$p_P$

$$V = \{I, L, S, R, N, Q, P\}$$

$$F = \{O, Z\}$$

$$K = \{O, Z\}$$

# Esempio



$d_I$	0
$d_L$	1
$d_S$	2
$d_R$	4
$d_Q$	7
$d_P$	11
$d_O$	13
$d_N$	7
$d_Z$	12

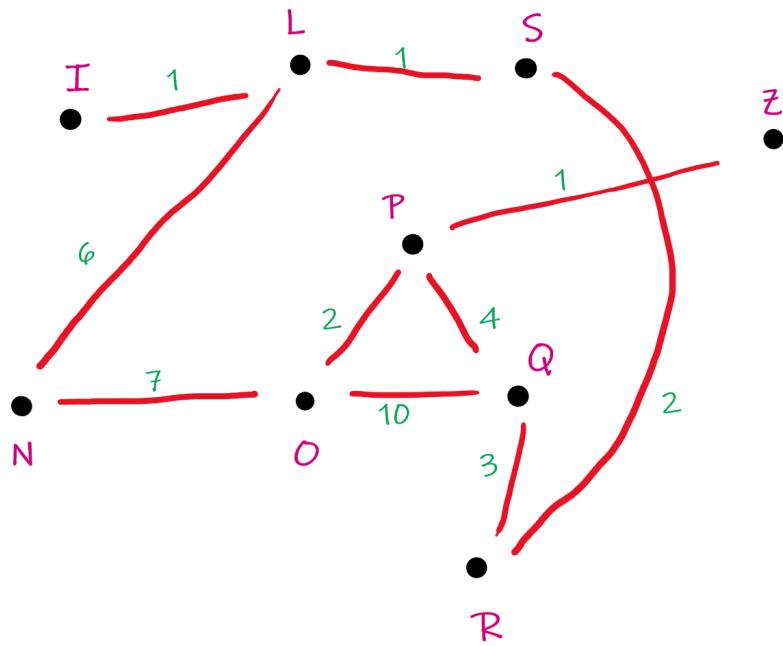
$p_I$	
$p_L$	$p_I$
$p_S$	$p_L$
$p_R$	$p_S$
$p_Q$	$p_R$
$p_P$	$d_Q$
$p_O$	$p_P$
$p_N$	$p_L$
$p_Z$	$p_P$

$$V = \{I, L, S, R, N, Q, P, \textcolor{red}{Z}\}$$

$$F = \{O\}$$

$$K = \{O\}$$

# Esempio



$d_I$	0
$d_L$	1
$d_S$	2
$d_R$	4
$d_Q$	7
$d_P$	11
$d_O$	13
$d_N$	7
$d_Z$	12

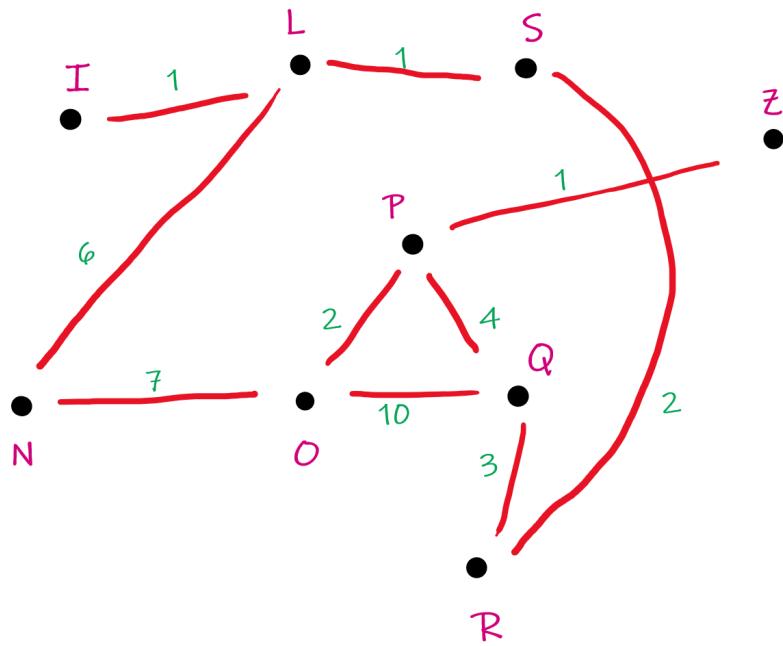
$p_I$	
$p_L$	$p_I$
$p_S$	$p_L$
$p_R$	$p_S$
$p_Q$	$p_R$
$p_P$	$d_Q$
$p_O$	$p_P$
$p_N$	$p_L$
$p_Z$	$p_P$

$$V = \{I, L, S, R, N, Q, P, Z\}$$

$$F = \{O\}$$

$$K = \{O\}$$

# Esempio



$d_I$	0
$d_L$	1
$d_S$	2
$d_R$	4
$d_Q$	7
$d_P$	11
$d_O$	13
$d_N$	7
$d_Z$	12

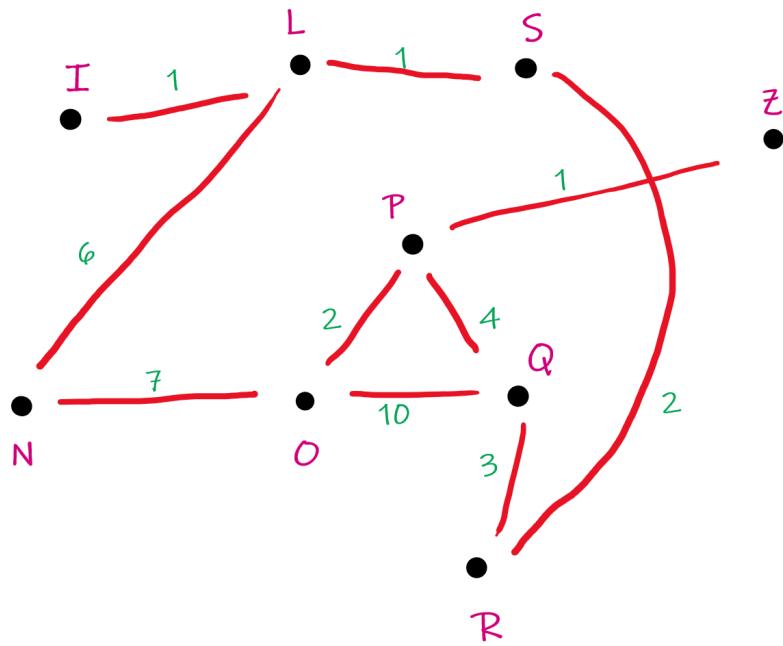
$p_I$	
$p_L$	$p_I$
$p_S$	$p_L$
$p_R$	$p_S$
$p_Q$	$p_R$
$p_P$	$d_Q$
$p_O$	$p_P$
$p_N$	$p_L$
$p_Z$	$p_P$

$$V = \{I, L, S, R, N, Q, P, Z\}$$

$$F = \{\textcolor{red}{O}\}$$

$$K = \{O\}$$

# Esempio



$d_I$	0
$d_L$	1
$d_S$	2
$d_R$	4
$d_Q$	7
$d_P$	11
$d_O$	13
$d_N$	7
$d_Z$	12

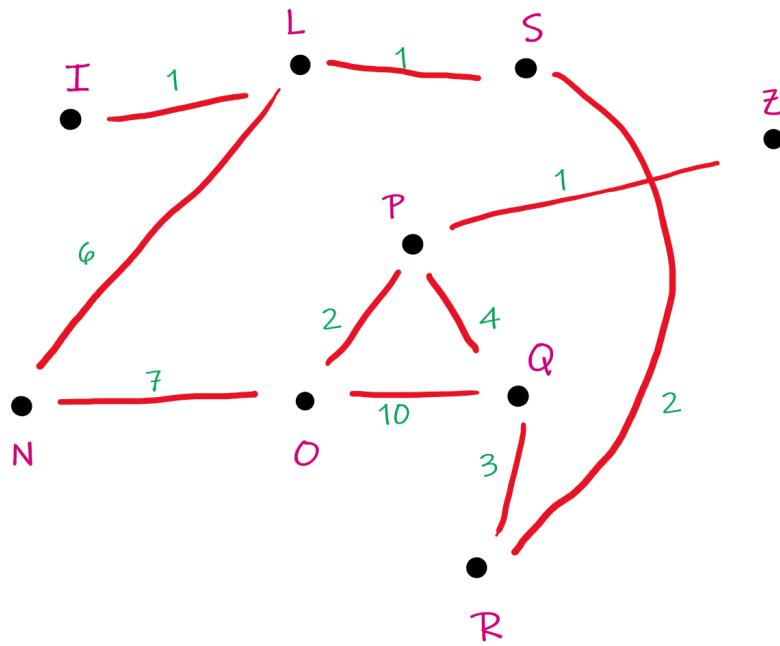
$p_I$	
$p_L$	$p_I$
$p_S$	$p_L$
$p_R$	$p_S$
$p_Q$	$p_R$
$p_P$	$d_Q$
$p_O$	$p_P$
$p_N$	$p_L$
$p_Z$	$p_P$

$$V = \{I, L, S, R, N, Q, P, Z, \textcolor{red}{O}\}$$

$$F = \{\}$$

$$K = \{\}$$

# Esempio



$d_I$	0
$d_L$	1
$d_S$	2
$d_R$	4
$d_Q$	7
$d_P$	11
$d_O$	13
$d_N$	7
$d_Z$	12

$p_I$	
$p_L$	$p_I$
$p_S$	$p_L$
$p_R$	$p_S$
$p_Q$	$p_R$
$p_P$	$d_Q$
$p_O$	$p_P$
$p_N$	$p_L$
$p_Z$	$p_P$

$$V = \{I, L, S, R, N, Q, P, Z, O\}$$

$$F = \{\}$$

$$K = \{\}$$

## RISORSE

 [github.com/mariocuomo/talks](https://github.com/mariocuomo/talks)



# MARIO CUOMO

-  [mariocuomo.github.io](https://mariocuomo.github.io)
-  [linkedin/in/mariocuomo](https://linkedin/in/mariocuomo)
-  [@mariocuomo.exe](https://instagram.com/@mariocuomo.exe)
-  [@mariocuomoEXE](https://twitter.com/@mariocuomoEXE)



# Grazie

