Esercitazione
14 gennaio 2021

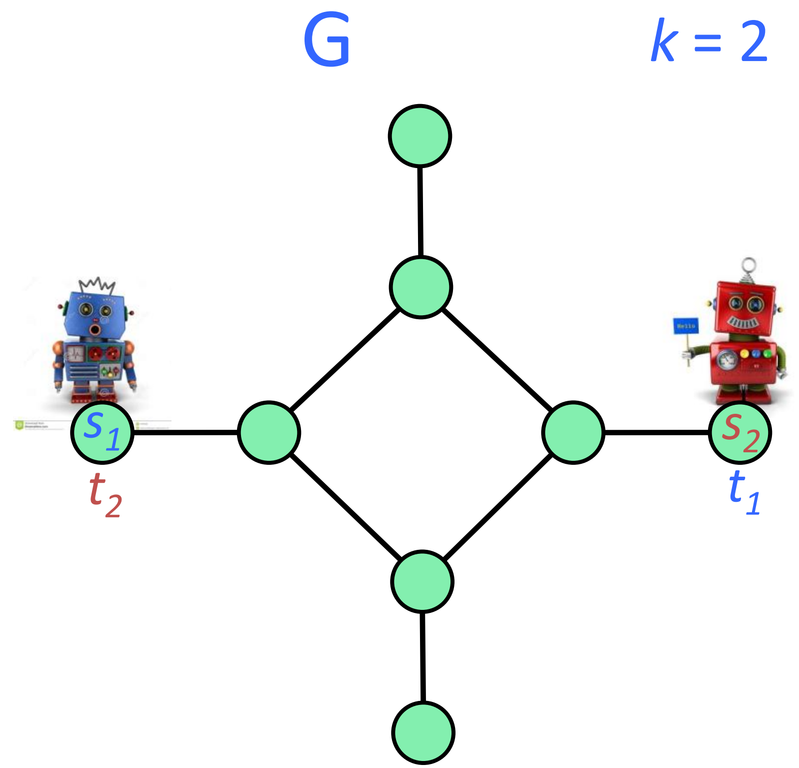
Problem Set 2

Esercizio 1: Babbo Natale e le nuove tecnologie

obiettivo: spostare il robot blu da s_1 a t_1
& il robot rosso da s_2 a t_2

mossa: sposta un robot dal nodo corrente a uno adiacente.

vincolo: i robots devono essere sempre a distanza almeno k fra loro

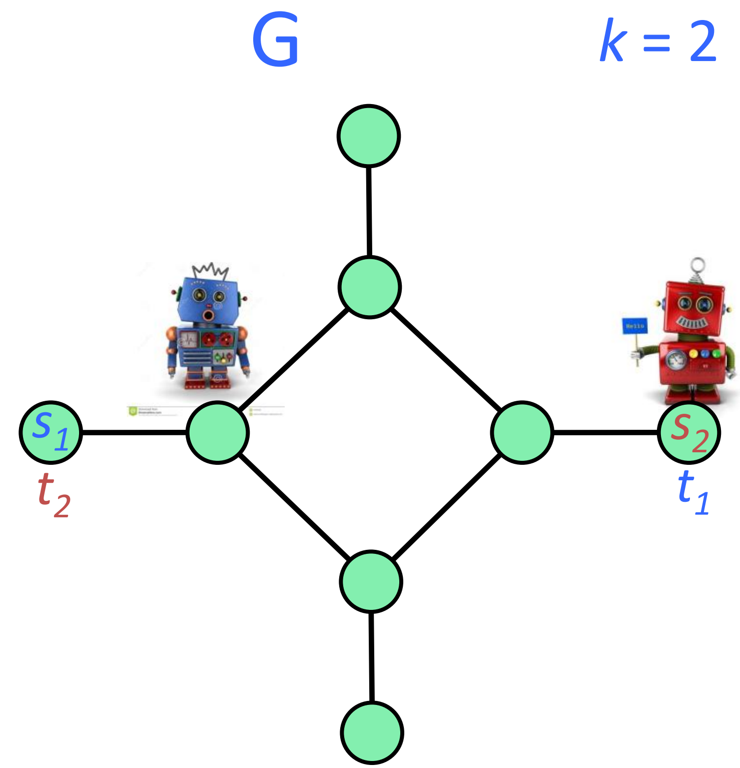


trovare la sequenza minima di mosse!

obiettivo: spostare il robot blu da s_1 a t_1
& il robot rosso da s_2 a t_2

mossa: sposta un robot dal nodo corrente a uno adiacente.

vincolo: i robots devono essere sempre a distanza almeno k fra loro

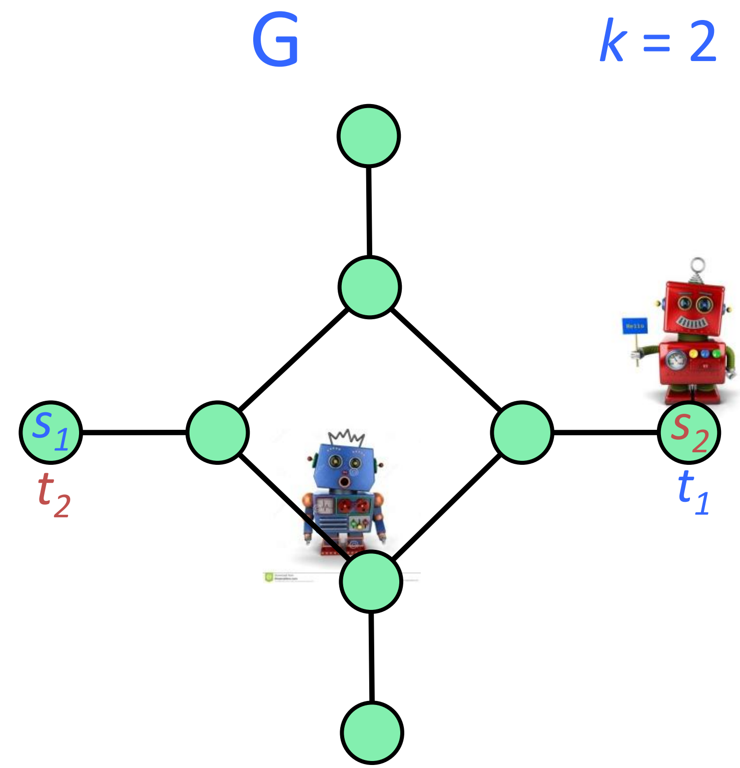


trovare la sequenza minima di mosse!

obiettivo: spostare il robot blu da s_1 a t_1
& il robot rosso da s_2 a t_2

mossa: sposta un robot dal nodo corrente a uno adiacente.

vincolo: i robots devono essere sempre a distanza almeno k fra loro

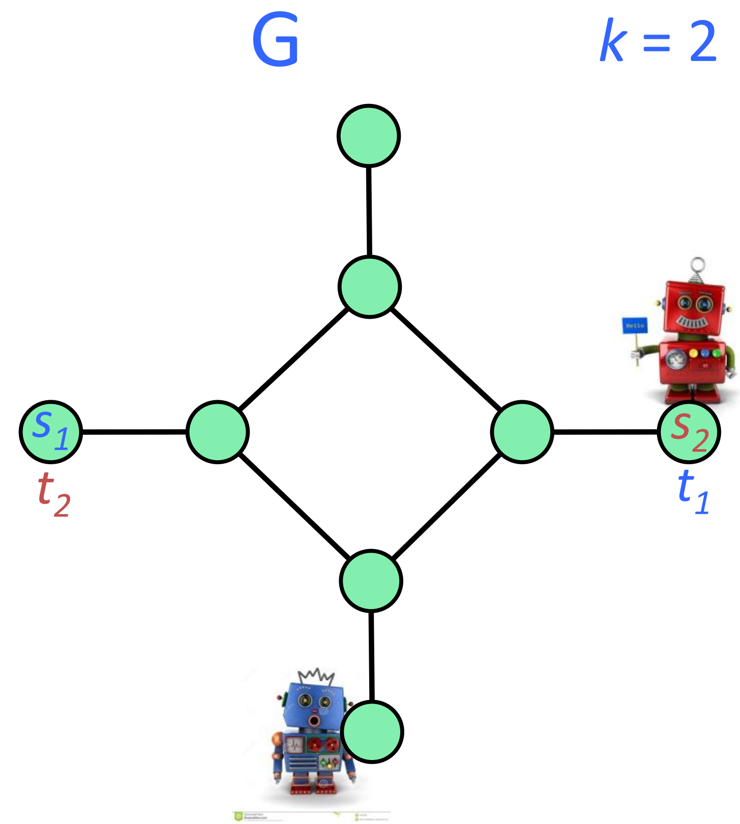


trovare la sequenza minima di mosse!

obiettivo: spostare il robot blu da s_1 a t_1
& il robot rosso da s_2 a t_2

mossa: sposta un robot dal nodo corrente a uno adiacente.

vincolo: i robots devono essere sempre a distanza almeno k fra loro

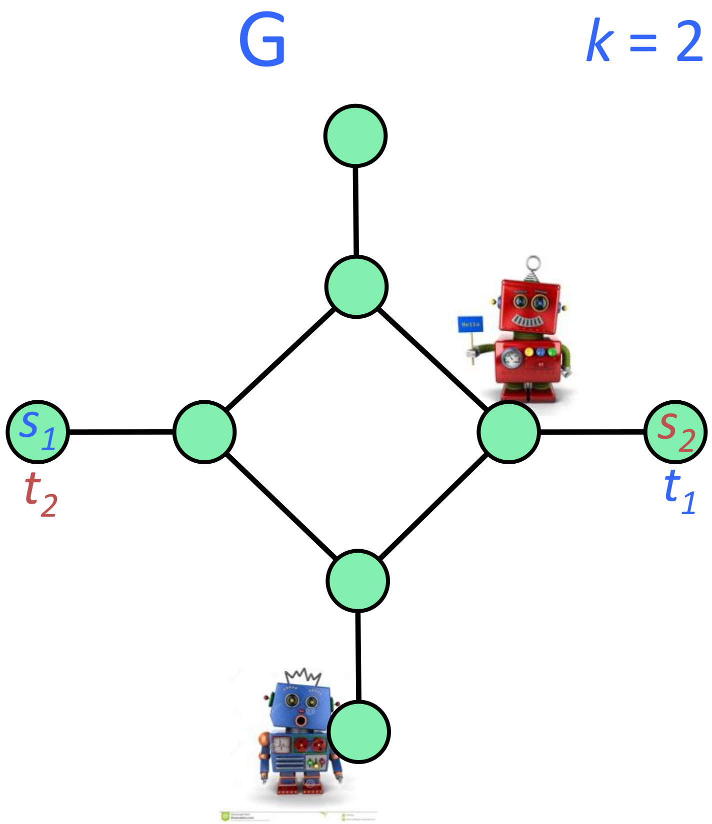


trovare la sequenza minima di mosse!

obiettivo: spostare il robot blu da s_1 a t_1
& il robot rosso da s_2 a t_2

mossa: sposta un robot dal nodo corrente a uno adiacente.

vincolo: i robots devono essere sempre a distanza almeno k fra loro

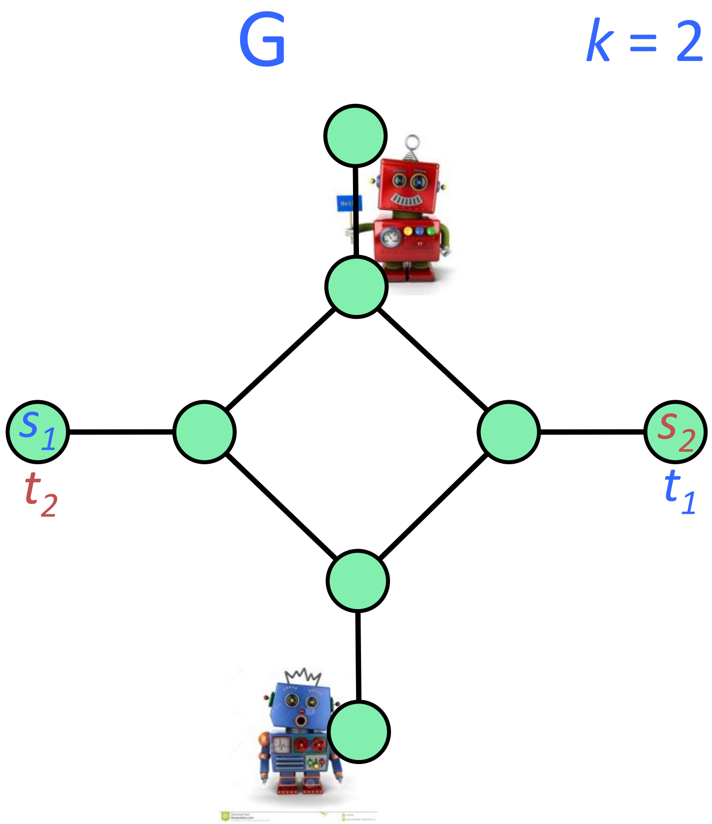


trovare la sequenza minima di mosse!

obiettivo: spostare il robot blu da s_1 a t_1
& il robot rosso da s_2 a t_2

mossa: sposta un robot dal nodo corrente a uno adiacente.

vincolo: i robots devono essere sempre a distanza almeno k fra loro

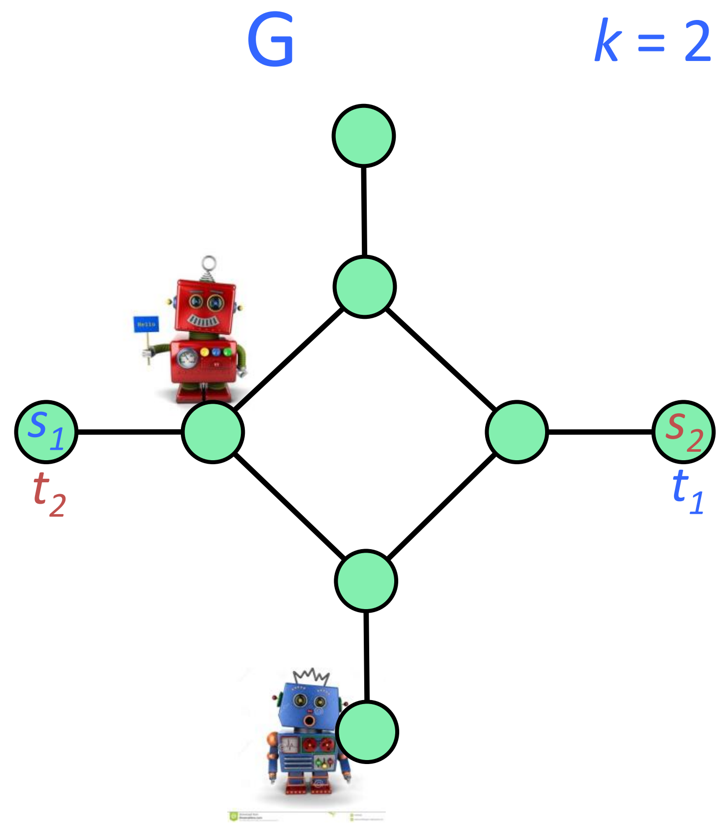


trovare la sequenza minima di mosse!

obiettivo: spostare il robot blu da s_1 a t_1
& il robot rosso da s_2 a t_2

mossa: sposta un robot dal nodo corrente a uno adiacente.

vincolo: i robots devono essere sempre a distanza almeno k fra loro

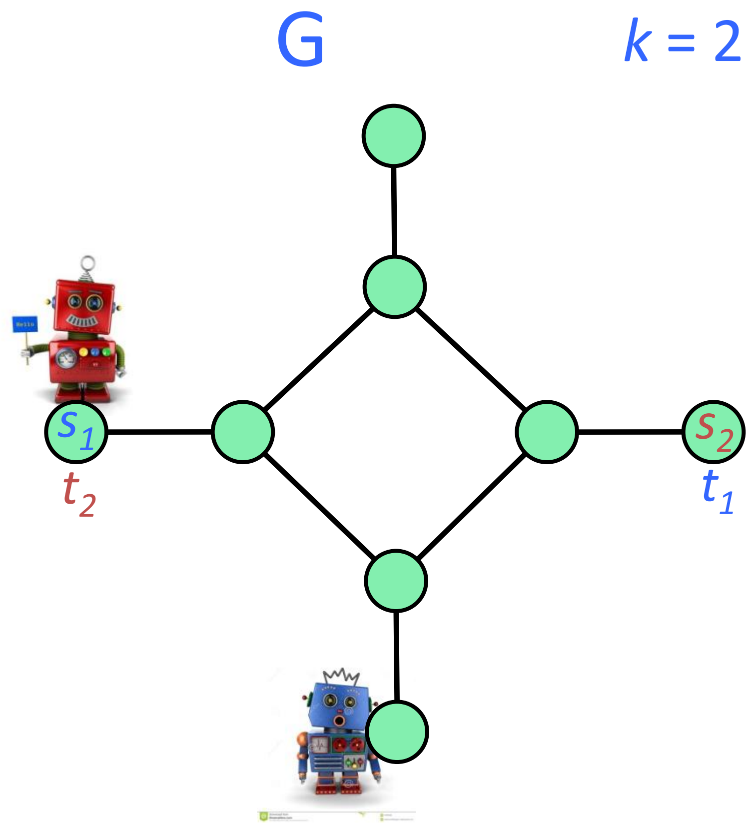


trovare la sequenza minima di mosse!

obiettivo: spostare il robot blu da s_1 a t_1
& il robot rosso da s_2 a t_2

mossa: sposta un robot dal nodo corrente a uno adiacente.

vincolo: i robots devono essere sempre a distanza almeno k fra loro

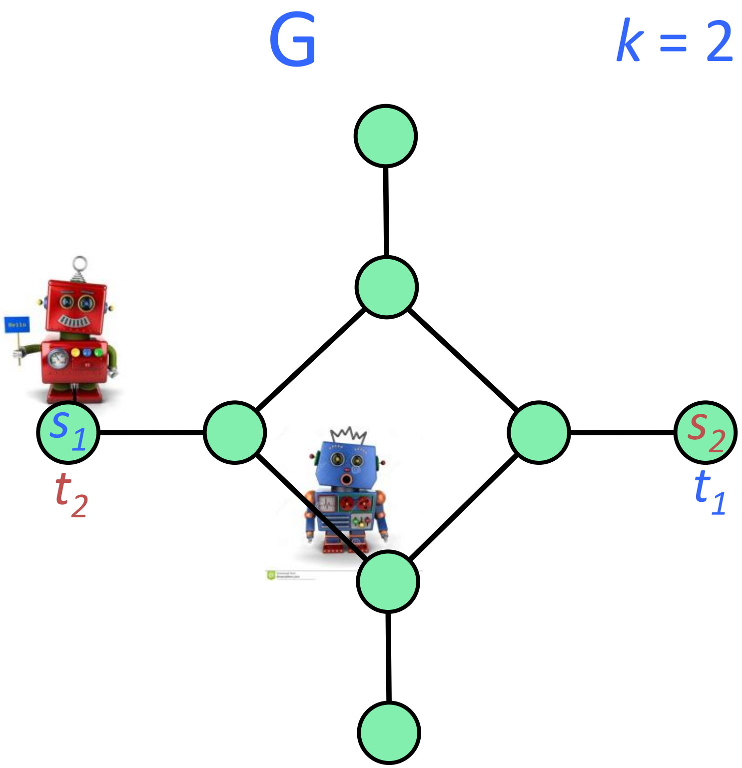


trovare la sequenza minima di mosse!

obiettivo: spostare il robot blu da s_1 a t_1
& il robot rosso da s_2 a t_2

mossa: sposta un robot dal nodo corrente a uno adiacente.

vincolo: i robots devono essere sempre a distanza almeno k fra loro



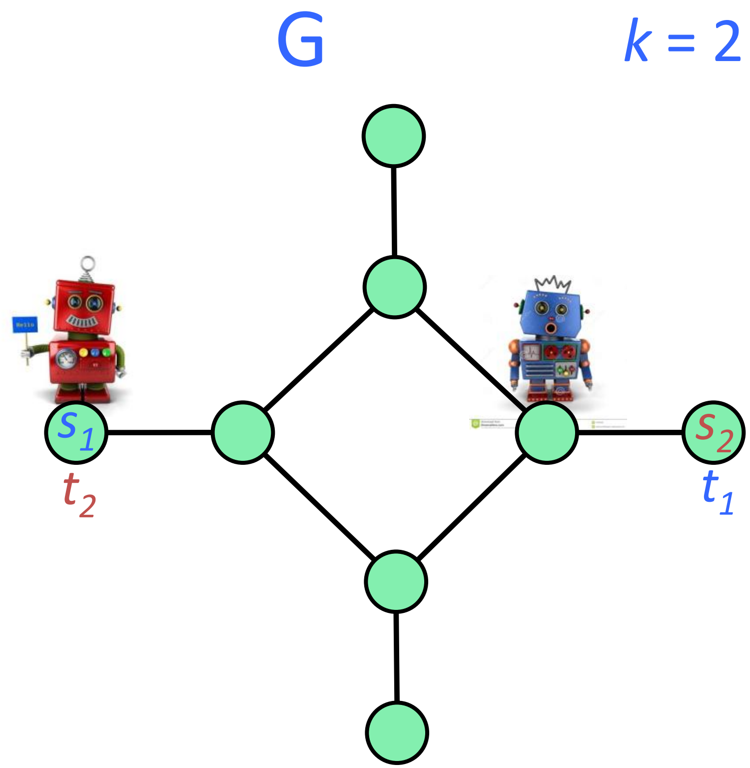
trovare la sequenza minima di mosse!

obiettivo: spostare il robot blu da s_1 a t_1
& il robot rosso da s_2 a t_2

mossa: sposta un robot dal nodo corrente a uno adiacente.

vincolo: i robots devono essere sempre a distanza almeno k fra loro

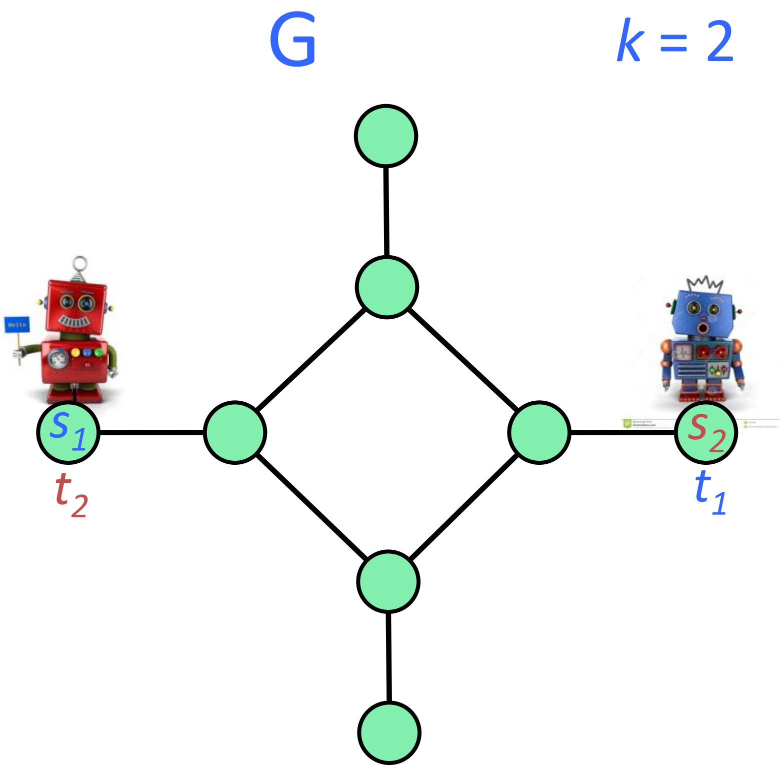
trovare la sequenza minima di mosse!



obiettivo: spostare il robot blu da s_1 a t_1
& il robot rosso da s_2 a t_2

mossa: sposta un robot dal nodo corrente a uno adiacente.

vincolo: i robots devono essere sempre a distanza almeno k fra loro



trovare la sequenza minima di mosse!

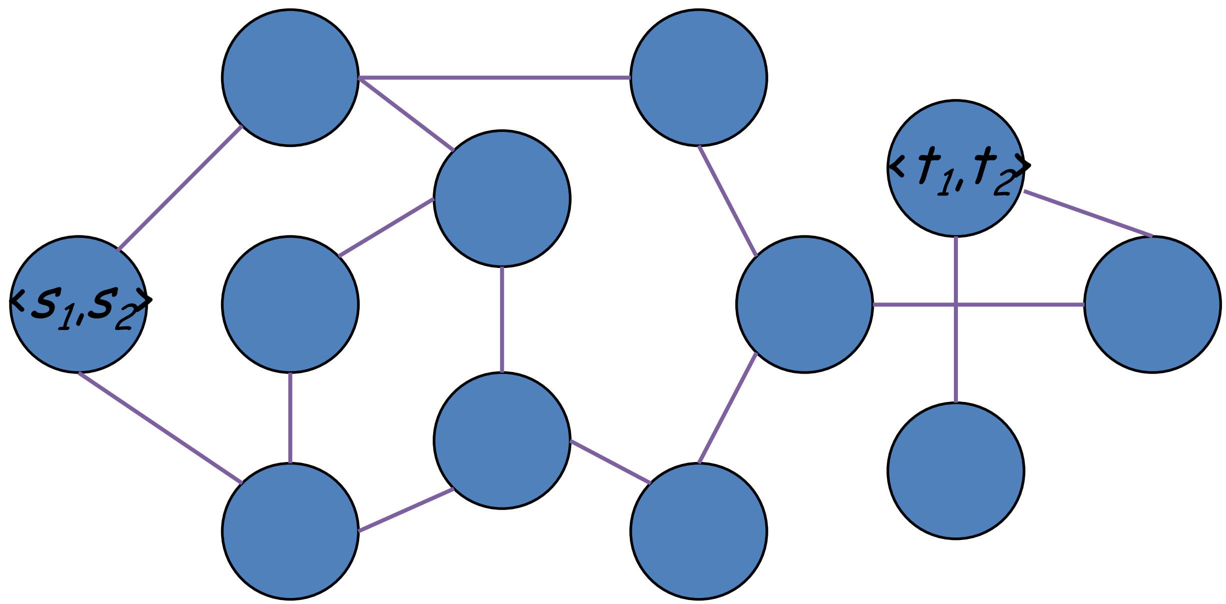
una soluzione

definire un grafo ausiliario $G'=(V',E')$

$$V' = \{\langle u,v \rangle : u,v \in V \text{ e } d_G(u,v) \geq k\}$$

$$E' = \{(\langle u,v \rangle, \langle x,y \rangle) : (u = x \text{ e } (v,y) \in E) \text{ o } (v = y \text{ e } (u,x) \in E)$$

robot blu è su u &
robot rosso è su v



cerco il cammino minimo da $\langle s_1,s_2 \rangle$ a $\langle t_1,t_2 \rangle$

correttezza:

Proprietà:

Esiste una sequenza di k mosse che porta i robot nella posizione finale se e soltanto se esiste un cammino in G' da $\langle s_1,s_2 \rangle$ a $\langle t_1,t_2 \rangle$ di lunghezza k.

complessità:

dimensione di G':

$$|V'| = O(n^2)$$

$$|E'| \leq \sum_{\langle u,v\rangle \in V'} \delta_{G'}(\langle u,v\rangle) \leq \sum_{\langle u,v\rangle \in V'} (\delta_G(u) + \delta_G(v)) = \sum_{\langle u,v\rangle \in V'} \delta_G(u) + \sum_{\langle u,v\rangle \in V'} \delta_G(v) \leq$$

$$\sum_{u,v \in V} \delta_G(u) + \sum_{u,v \in V} \delta_G(v) \leq n \sum_{u \in V} \delta_G(u) + n \sum_{v \in V} \delta_G(v) \leq 2nm + 2nm = O(nm)$$

-costruzione di G':

$O(|V'|+|E'|)$
$=O(mn)$

se trovo le distanze fra
tutte le coppie in $O(mn)$
(n visite BFS, una da ogni vertice)

-calcolo cammino minimo in G':

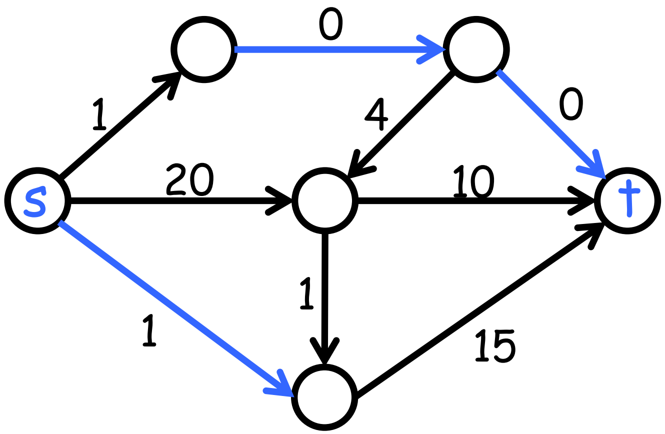
$O(|V'|+|E'|)$
$=O(n^2+mn)$
$=O(mn)$

$O(mn)$

Esercizio 3:
E tornava l'emigrante.

Input: -grafo orientato $G=(V,E,w)$ con pesi non negativi
-$B \subseteq E$ sottoinsieme di archi blu
-k intero, $s,t \in V$

Output: un cammino di costo minimo da s a t che usa al più k archi blu

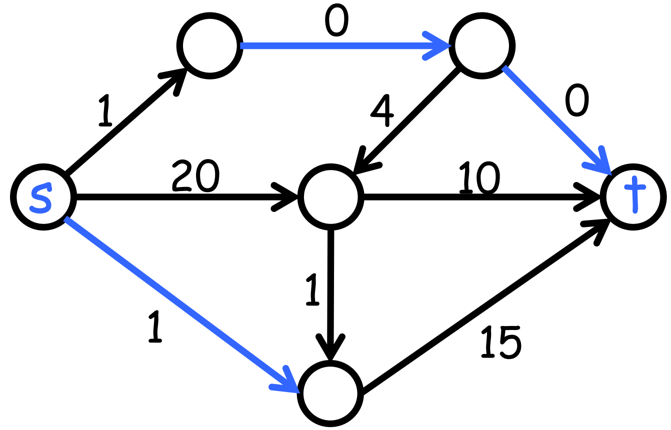


$k=1$

idea 1: "indovinare" i k archi blu della soluzione.

$\overline{G} = (V, E \setminus B)$
per ogni k-tupla F di archi in B:
    - calcola il cammino minimo da s a t nel grafo $\overline{G}$+F
restituisci il miglior cammino trovato

k=1

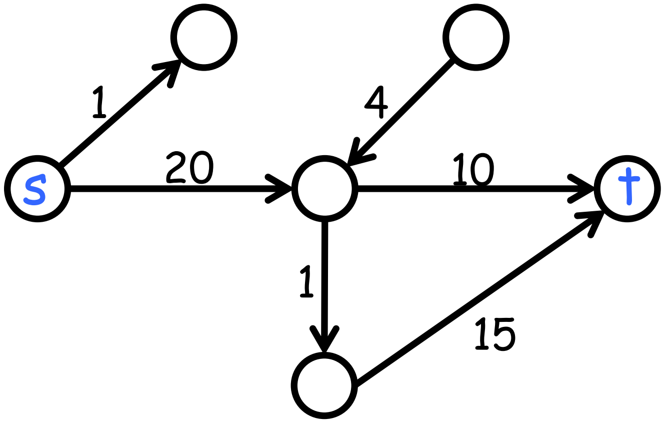


idea 1: "indovinare" i k archi blu della soluzione.

$\overline{G}$ = (V,E\B)
per ogni k-tupla F di archi in B:
- calcola il cammino minimo da s a t nel grafo $\overline{G}$+F

restituisci il miglior cammino trovato

k=1

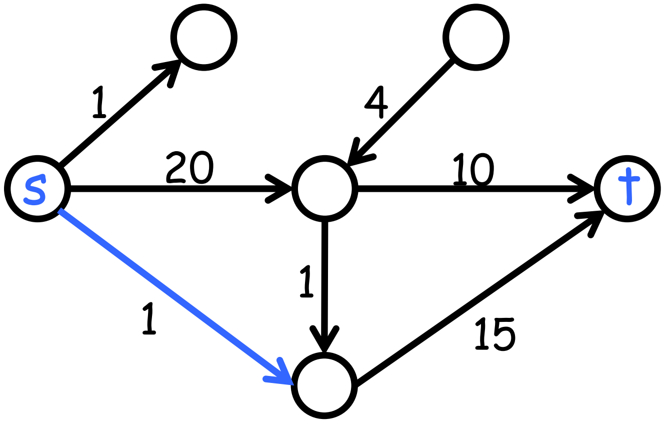


idea 1: "indovinare" i k archi blu della soluzione.

$\overline{G}$ = (V,E\B)
per ogni k-tupla F di archi in B:
- calcola il cammino minimo da s a t nel grafo $\overline{G}$+F

restituisci il miglior cammino trovato

k=1

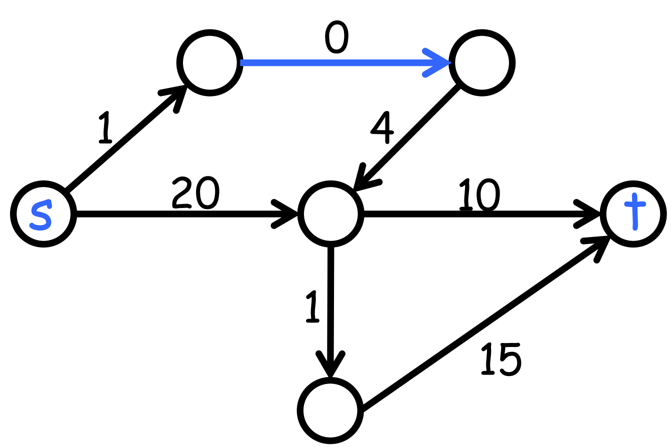


idea 1: “indovinare” i k archi blu della soluzione.

$\overline{G} = (V, E \setminus B)$
per ogni k-tupla F di archi in B:
- calcola il cammino minimo da s a t nel grafo $\overline{G}$+F

restituisci il miglior cammino trovato

k=1



idea 1: “indovinare” i k archi blu della soluzione.

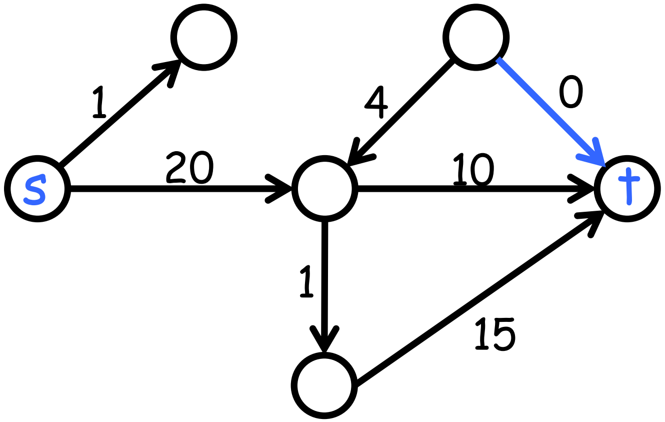
$\overline{G}$ = (V,E\B)

per ogni k-tupla F di archi in B:

- calcola il cammino minimo da s a t nel grafo $\overline{G}$+F

restituisci il miglior cammino trovato

k=1



idea 1: “indovinare” i k archi blu della soluzione.

$\overline{G}$ = (V,E\B)

per ogni k-tupla F di archi in B:

- calcola il cammino minimo da s a t nel grafo $\overline{G}$+F

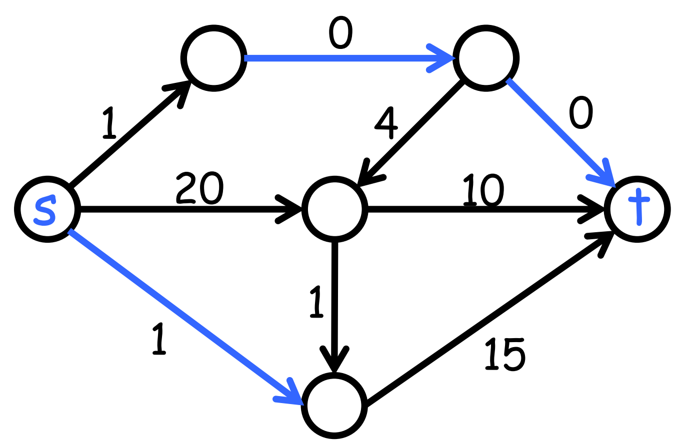
restituisci il miglior cammino trovato

correttezza?

k=1

-ogni cammino calcolato è un cammino ammissibile;

-quando guardo la k-tupla usata dalla soluzione (o un sovrainsieme) il cammino calcolato è quello ottimo cercato

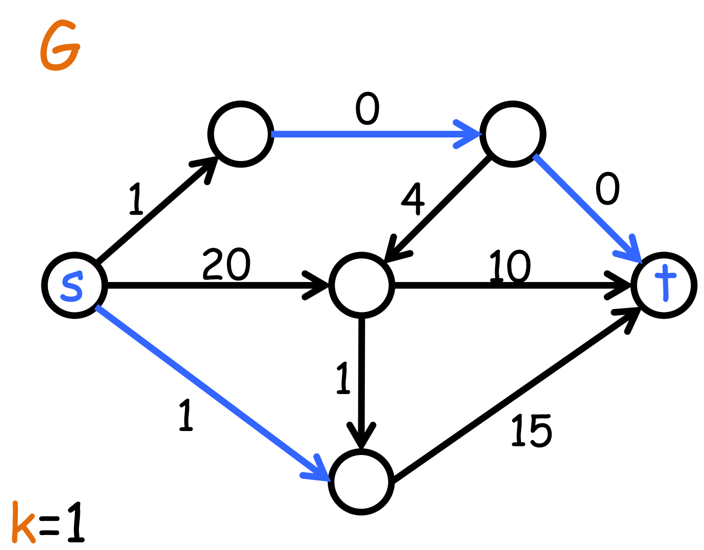


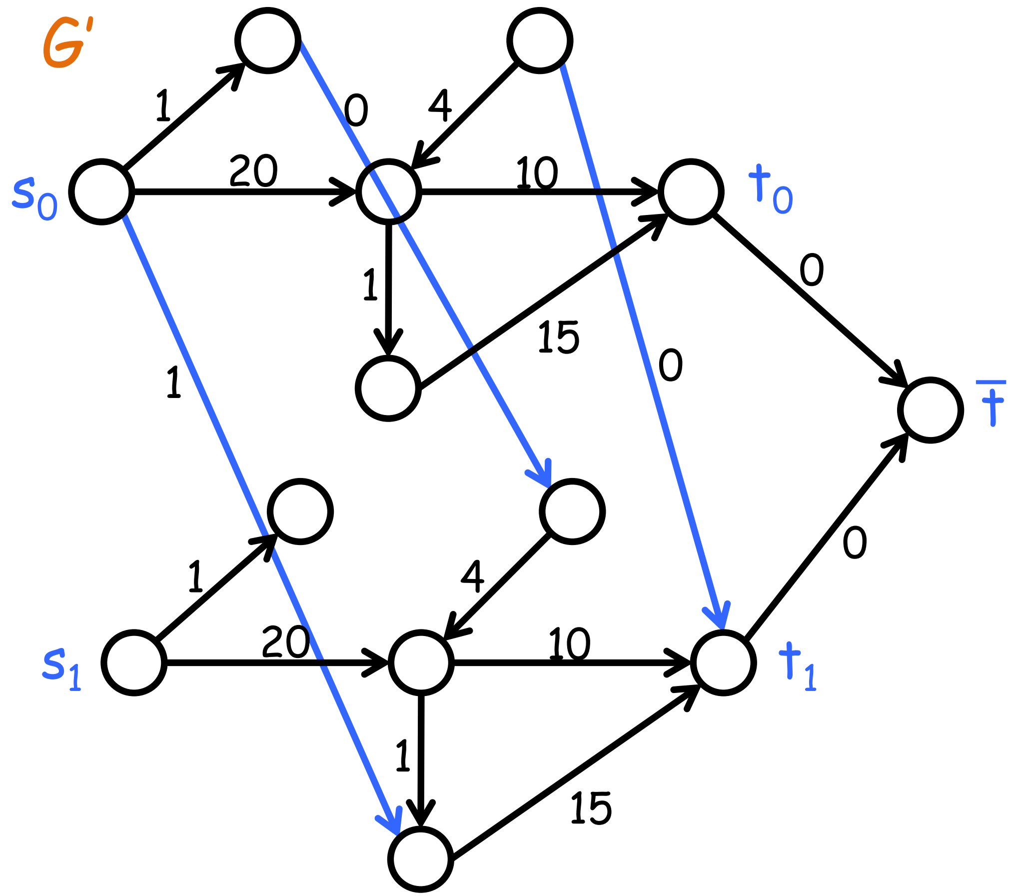
complessità?

$O(|B|^k(m+n \log n))$

idea 2: ridurre il problema al calcolo di un cammino minimo su un opportuno grafo ausiliario G'

- G' fatto "a livelli"
- ogni volta che uso un arco blu sono costretto a cambiare livello
- #livelli dipende da k





cerco il cammino minimo da s_0 a $\bar{t}$ in G'

nodi:

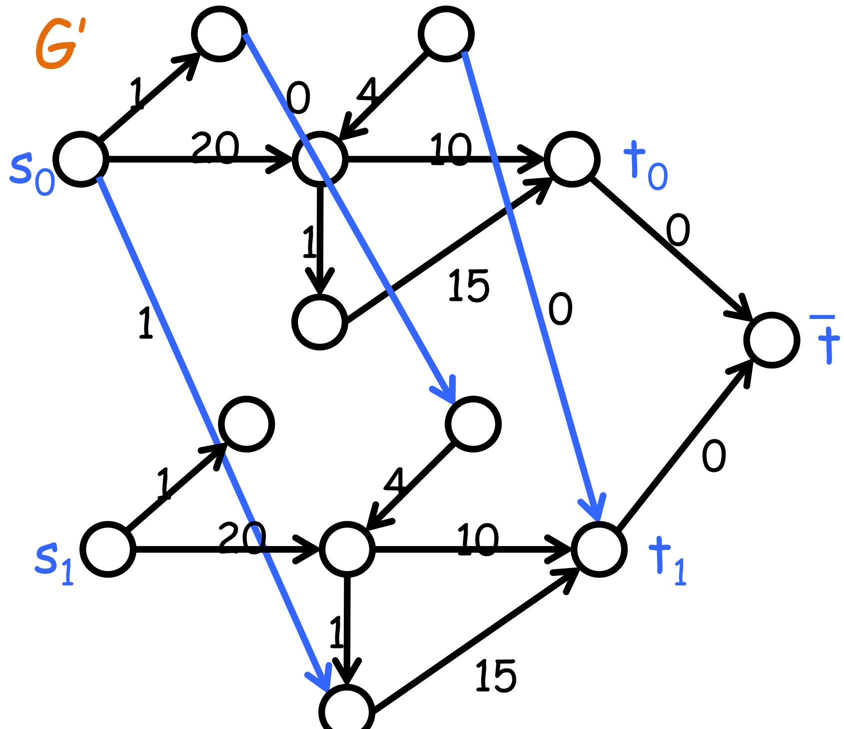
definizione di G'

- per ogni $v \in V$ ho k+1 nodi $v_0, v_1, \ldots v_k$
- un nodo $\bar{t}$

archi:

- per ogni arco (u,v) non blu in G ho gli archi (u_i,v_i), $i=0,1,\ldots k$ di peso $w(u,v)$
- per ogni arco (u,v) blu in G ho gli archi (u_i,v_{i+1}), $i=0,1,\ldots k-1$ di peso $w(u,v)$
- ho archi $(t_i,\bar{t})$, $i=0,1,\ldots k$, di peso 0

soluzione cercata: cammino minimo in G' da s_0 a $\bar{t}$



correttezza:

Proprietà:
Esiste un cammino in G sa s a t che usa al più k archi blu di costo W
se e soltanto se esiste un cammino in G' da s_0 a $\overline{t}$ di costo W.

complessità:

dimensione di G':

$n' = nk + 1 = \Theta(nk)$

$m' \leq (k+1)m + k = \Theta(mk)$

-costruzione di G':

$$O(m'+n') = O(k(m+n))$$

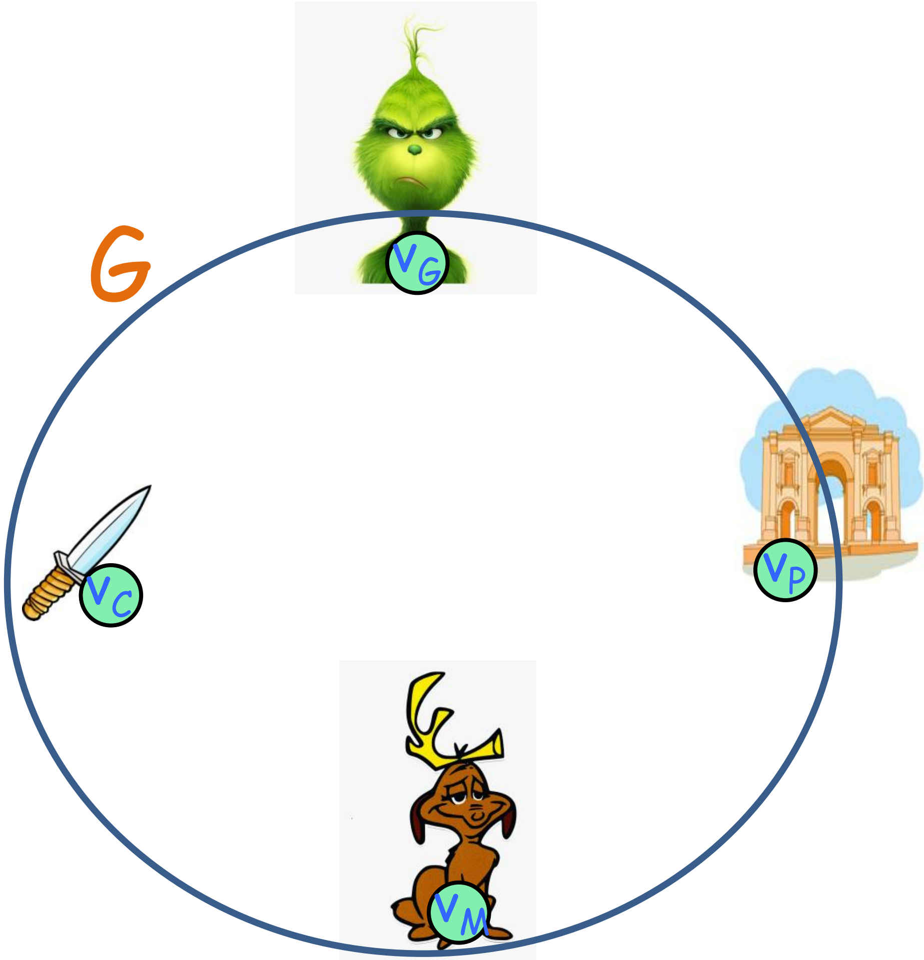
-calcolo cammino minimo in G':

$$O(m'+n' \log n') = O(k(m+n \log n))$$

$$O(k(m+n \log n))$$

Esercizio 2:
L’uccisione di
Babbo Natale.

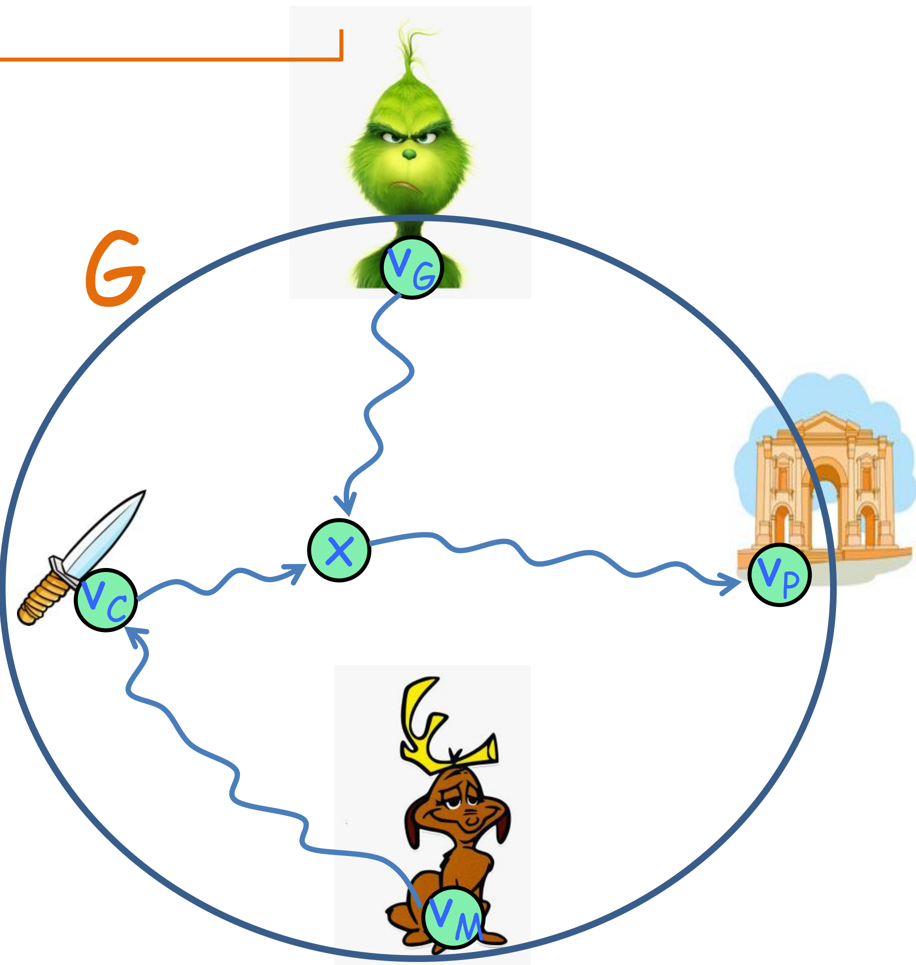
- Il Grinch vuole uccidere Babbo Natale
- Ha bisogno del coltello (che è in v_C)
- deve trovarsi fra Δ ore in v_P
- Max (che è in v_M) può aiutarlo raccogliendo il coltello per lui
- Se lo possono scambiare in qualsiasi nodo (ma solo il Grinch può usarlo)
- attraversare un arco richiedere un'ora per il Grinch e due per Max



idea: "indovinare" il nodo x dove il Grinch e Max si scambiano il coltello

$$\tau(x) := \underbrace{\max\{\underbrace{d(v_G,x)}_{\text{tempo per il Grinch di arrivare a } x}\;,\; \underbrace{2\,(d(v_M,v_C)+d(v_C,x))}_{\text{tempo per Max di portare il coltello su } x}\}}_{\text{tempo in cui si scambiano il coltello}} + d(x,v_P)$$

$\tau(x)$: tempo in cui il Grinch arriva (con il coltello) alle porte della città



osservazione: $\tau(x)$ è diponibile in tempo cosante se ho tutte le distanza a singola sorgente da v_G, v_M, v_C, v_P

$\tau(x) := \max\{\ d(v_G, x)\ ,\ 2\,(d(v_M, v_C) + d(v_C, x))\ \} + d(x, v_P)$

Algoritmo

- calcola distanze/alberi BFS con sorgenti v_G, v_M, v_C, v_P } $O(4(m+n)) = O(m+n)$
- **if** $d(v_G, v_C) + d(v_C, v_P) \leq \Delta$ **then** *il Grinch va da solo* } $O(1)$

  else

  $z = \arg\min_{x \in V} \tau(x)$ } $O(n)$

  if $\tau(z) \leq \Delta$ **then** *Il Grinch e Max si incontrano in z* } $O(1)$

  else *Babbo Natale è salvo*

$\Rightarrow$ $O(m+n)$

Qualche altro esercizio su grafi

Problema (mandare la pedina in buca)

Sia $G = (V, E)$ un grafo non orientato e non pesato, dove ogni arco $e \in E$ ha uno fra tre possibili colori $\{1, 2, 3\}$, e sia $U \subseteq V$ un sottoinsieme di nodi *speciali*. Su un nodo $s \in V$ è posizionata una pedina, inizialmente di colore 1. Il vostro obiettivo è quello di portare la pedina su un nodo *target* $t \in V$. Per far ciò potete usare le seguenti *mosse*:

- `sposta`: se la pedina si trova sul nodo $u \in V$, ha colore i, e c'è un arco $(u, v) \in E$ di colore i, allora potete spostare la pedina da u a v. Il colore della pedina resta i;
- `cambia colore`: se la pedina si trova su un nodo speciale $u \in U$, potete ricolorare la pedina del colore che volete. La pedina in questo caso, dopo la mossa, resta sul nodo u.

Progettate un algoritmo efficiente che trova, se ne esiste una, la sequenza più corta di mosse che risolve l'istanza del gioco.

Problema (Gualà e Clementi vanno a vedere la (maggggica) Roma)

Una città è modellata come un grafo diretto e pesato $G = (V, E, w)$, dove ad ogni arco $e \in E$ è associato un peso $w(e) \geq 0$ che rappresenta il costo, in termini di benzina consumata, per attraversare l'arco (strada) e. In questa città, i vostri docenti del corso di algoritmi, Gualà e Clementi, vogliono andare a vedere la partita della Roma allo stadio, che si trova nel nodo t. Loro sono rispettivamente nei nodi s_1 e s_2, e possiedono una macchina ciascuno. Volendo, possono incontrarsi in un nodo del grafo, parcheggiare una delle due macchine, e proseguire insieme. Ma di solito in questa città, parcheggiare costa. Per ogni nodo v, dunque, conoscono il costo $c(v)$ del parcheggio presente in v (si può assumere per semplicità che $c(s_1) = c(s_2) = c(t) = 0$). Progettate un algoritmo che in tempo $O(m + n \log n)$ calcoli la soluzione che Gualà e Clementi devono adottare per spendere complessivamente il meno possibile in termini di corso della benzina più costo del parcheggio.

E se il grafo G fosse diretto?

Problema

Una rete stradale di una città è modellata come un grafo non orientato e non pesato $G = (V, E)$. Dovete portare un pacco ad un cliente che si trova nel nodo t. Il pacco inizialmente è depositato in un magazzino che si trova nel nodo s. Avete a disposizione due droni, posizionati inizialmente nei nodi d_1 e d_2. I droni sono perfettamente uguali. Ogni drone può:

- spostarsi dal nodo in cui si trova verso un nodo adiacente usando un arco del grafo;
- caricare il pacco, se il drone e il pacco si trovano nello stesso nodo;
- depositare il pacco su un certo nodo u, se il drone si trova nel nodo u.

Per questioni di batteria, i droni possono percorrere una distanza di al più k archi, mentre caricare e scaricare il pacco non consuma energia. Si progetti un algoritmo più efficiente possibile che decida se c'è un modo per portare il pacco a destinazione.