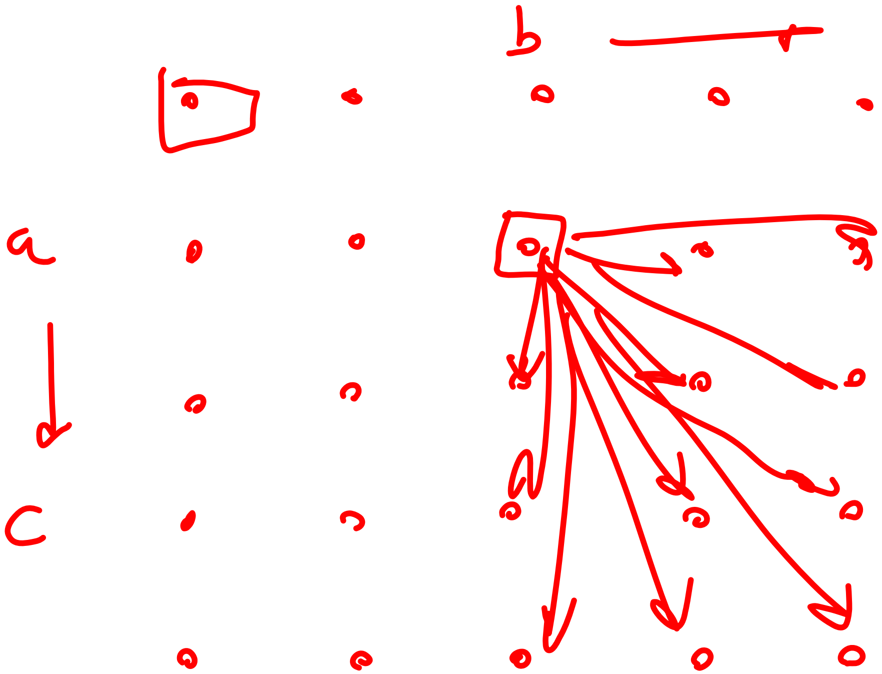
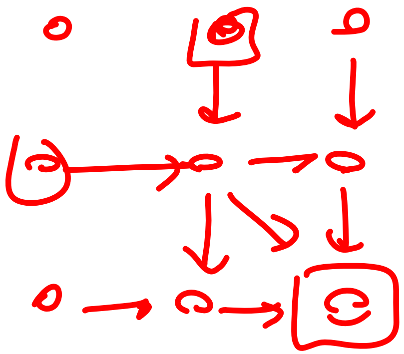
Dato $n \in \mathbb{N}$, si consideri il grafo orientato $G = (V, E)$, ove

$$
\begin{array}{rcl}
V & = & \{v_{ab} : 1 \leq a, b \leq n \ \ (a, b \in \mathbb{N})\} \\
E & = & \{(v_{ab}, v_{cd}) : v_{ab}, v_{cd} \in V,\ v_{ab} \neq v_{cd},\ a \leq c,\ b \leq d\}\,.
\end{array}
$$

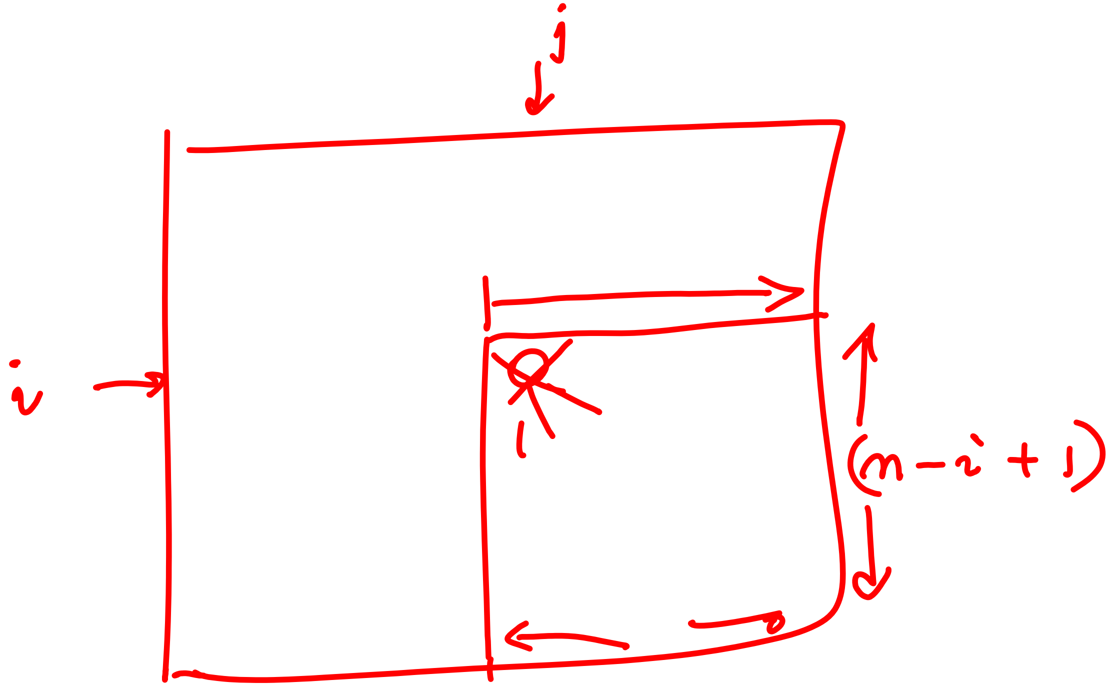
(a) Siano $w_1 : E \to \mathbb{R}$ e $w_2 : E \to \mathbb{R}^+$ due funzioni peso assegnate. Per ciascuno dei grafi pesati (G, w_1) e (G, w_2) si **illustri** un algoritmo per calcolare in maniera efficiente tutti i cammini minimi dal nodo v_{11}, **giustificando** opportunamente la scelta effettuata, e se ne **valuti** la complessità computazionale in funzione di $|V|$ e di $|E|$.

(b) Sapreste **esprimere** le complessità trovate nel punto precedente solo in funzione di n?





$$0 + 1 + 1 + (2^2 - 1) + 2 + 2 + (2^2 + 1) + (2^2 + 1) + (3^2 - 1)$$



$$\sum_{i=1}^{n}\sum_{j=1}^{n}\left[\underbrace{(n-i+1)}\underbrace{(n-j+1)} - 1\right]$$

$n-i+1$	i
$\frac{n}{1}$	$\frac{1}{n}$

$(n-j+1)$

$$\sum_{i=1}^{n}\sum_{j=1}^{n}(ij - 1) = \sum_{i=1}^{n}\sum_{j=1}^{n} ij - n^2$$

$$= \sum_{i=1}^{n} i\left(\sum_{j=1}^{n} j\right) - n^2$$

$$= \left[\frac{n(n+1)}{2}\right]^2 - n^2 = O(n^4)$$

ESERCIZIO 4 (esame completo/II prova in itinere)

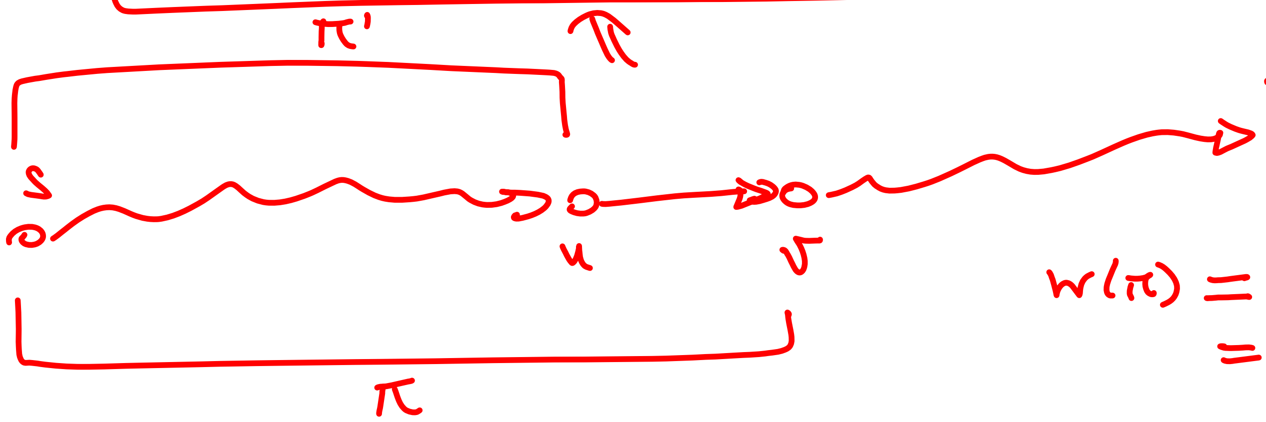
Sia $G = (V, E)$ un grafo orientato con funzione peso $w : E \longrightarrow \mathbf{R}^+$ e sorgente $s \in V$, i cui nodi sono tutti raggiungibili da s.

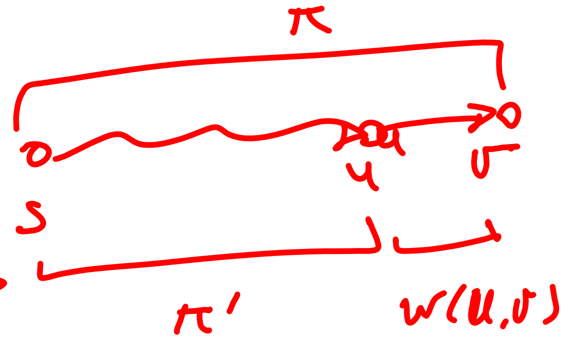
(a) Si definiscano il *grafo* G'_s *dei cammini minimi da* s *in* G e la nozione di *albero dei cammini minimi da* s *in* G (rispetto alla funzione peso w).

(b) Dato un arco $(u, v) \in E$, si dimostri che (u, v) appartiene al grafo dei cammini minimi se e solo se $\delta(s, v) = \delta(s, u) + w(u, v)$, dove δ è la funzione distanza su G indotta da w.

(c) Si illustri un algoritmo efficiente per calcolare il grafo dei cammini minimi da s per il grafo pesato (G, w).

$$\boxed{\delta(s,v) = \delta(s,u) + w(u,v)}$$





$$w(\pi) = w(\pi') + w(u,v) =$$
$$= \delta(s,v) + w(u,v) = \delta(s,v)$$

$$\boxed{\delta(s,v) = w(\pi) = w(\pi') + w(u,v) = \delta(s,u) + w(u,v)}$$

Sia $G = (V, E)$ un grafo orientato, $q \in V$ un nodo assegnato di G e $w : E \to \mathbb{R}^+$ una funzione peso in G a valori positivi.

Un q-*cammino* in G da u a v è un cammino in G da u a v passante per q.

Un q-cammino in G da u a v si dice *minimo* se il suo peso è minimo (relativamente alla funzione peso w) rispetto a tutti i q-cammini in G da u a v.

Si descriva un algoritmo efficiente che calcoli per ciascuna coppia di nodi $(u, v) \in V \times V$ un q-cammino minimo da u a v e se ne valuti la complessità computazionale.

In analogia con la nozione di *distanza* (minima) tra due nodi, si proponga una definizione della nozione di *distanza massima* tra due nodi in un grafo orientato pesato.

Quindi si descriva un algoritmo per determinare le distanze massime di tutti i nodi da una data sorgente $s \in V$ in un grafo orientato aciclico $G = (V, E)$ con funzione peso $w : E \longrightarrow \mathbf{R}$ e se ne valuti la complessità computazionale.

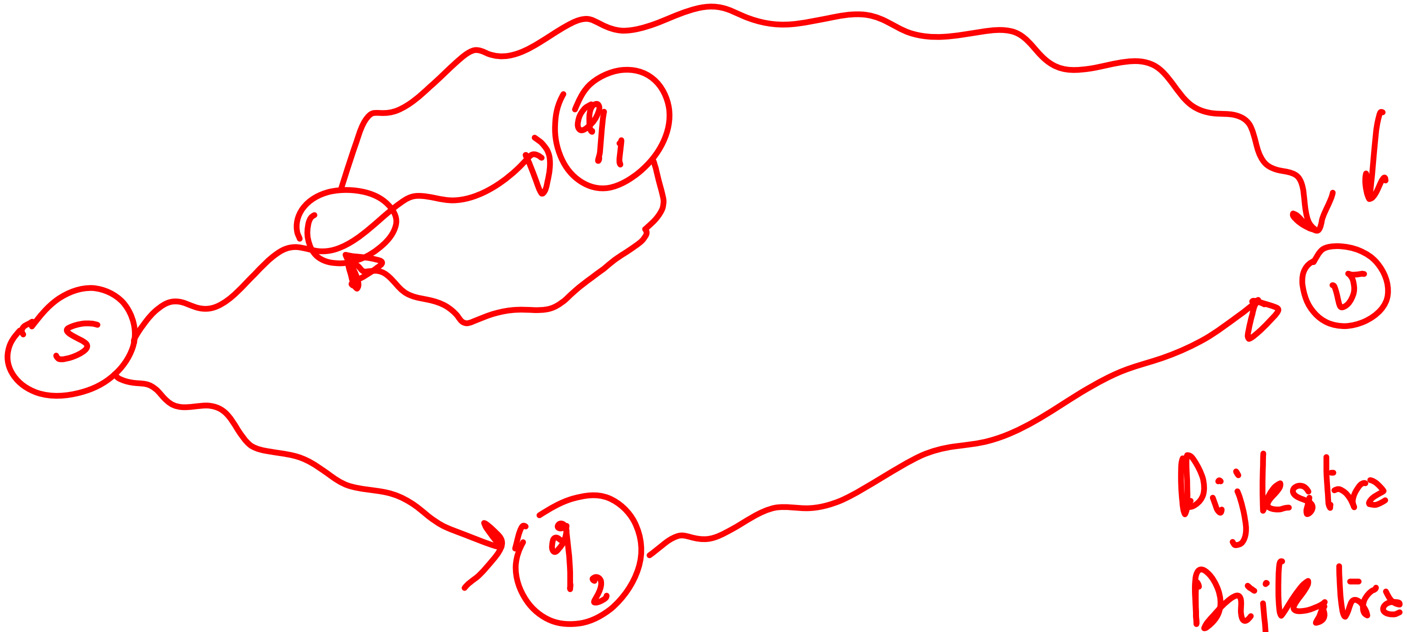
$$\Delta(u,v) = \sup(\{w(\pi) : \pi \in PATH(u,v)\})$$

$$= -\underbrace{\inf(\{-w(\pi) : \pi \in PATH(u,v)\})}$$

$$\Downarrow$$

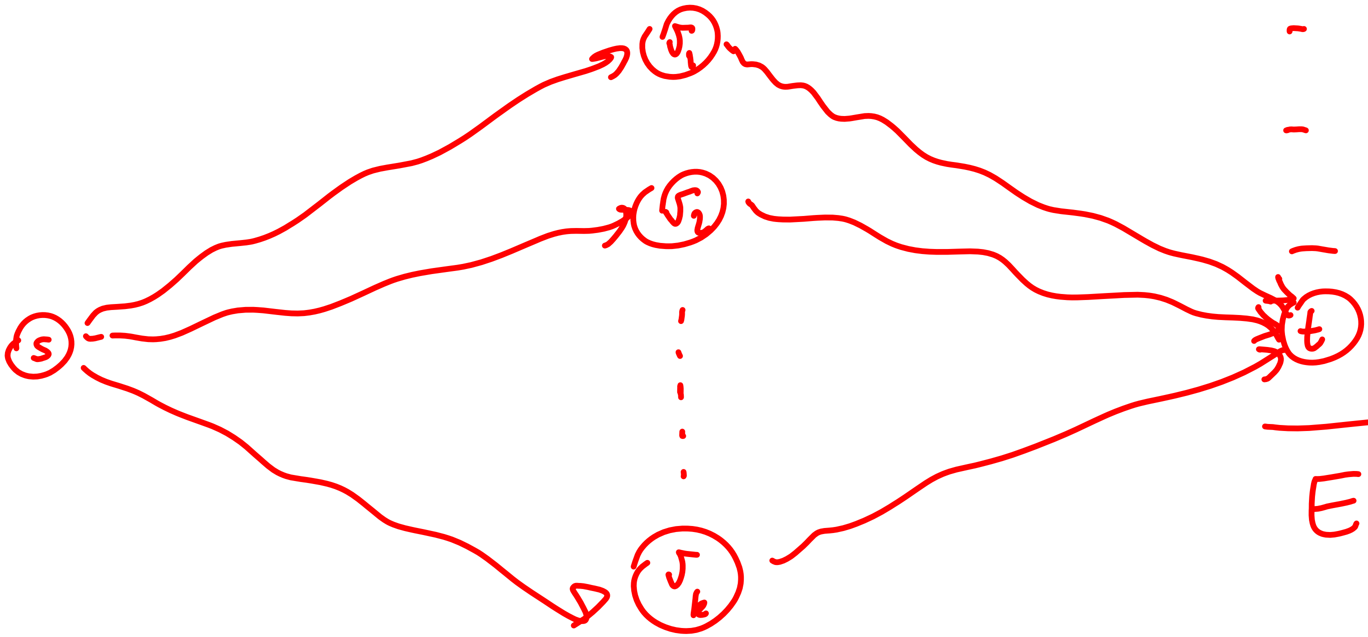
$$w'(u,v) = -w(u,v)$$

Sia $G = (V, E)$ un grafo orientato con una funzione peso non negativa $w : E \to \mathbb{R}^+$ e siano assegnati in G tre nodi distinti $s, q_1, q_2 \in V$.
Si descriva un algoritmo efficiente, valutandone anche la complessità computazionale, per calcolare tutti i cammini minimi (non necessariamente semplici) da s a ciascun nodo v di G, passanti per *almeno* uno dei due nodi assegnati q_1, q_2.



Dijkstra(G, q_1)
Dijkstra(G, q_2)
Dijkstra(G, s)

Sia $G = (V, E)$ un grafo orientato con una funzione peso non negativa $w : E \to \mathbb{R}^+$ e siano assegnati $(k+2)$ nodi distinti $s, t, v_1, v_2, \ldots, v_k$ di G.
Si descriva un algoritmo efficiente, valutandone anche la complessità computazionale in funzione di $|V|$, $|E|$ e k, per calcolare un cammino minimo (non necessariamente semplice) da s a t, passante per *almeno* uno dei k nodi assegnati $v_1, v_2, \ldots, v_k$.

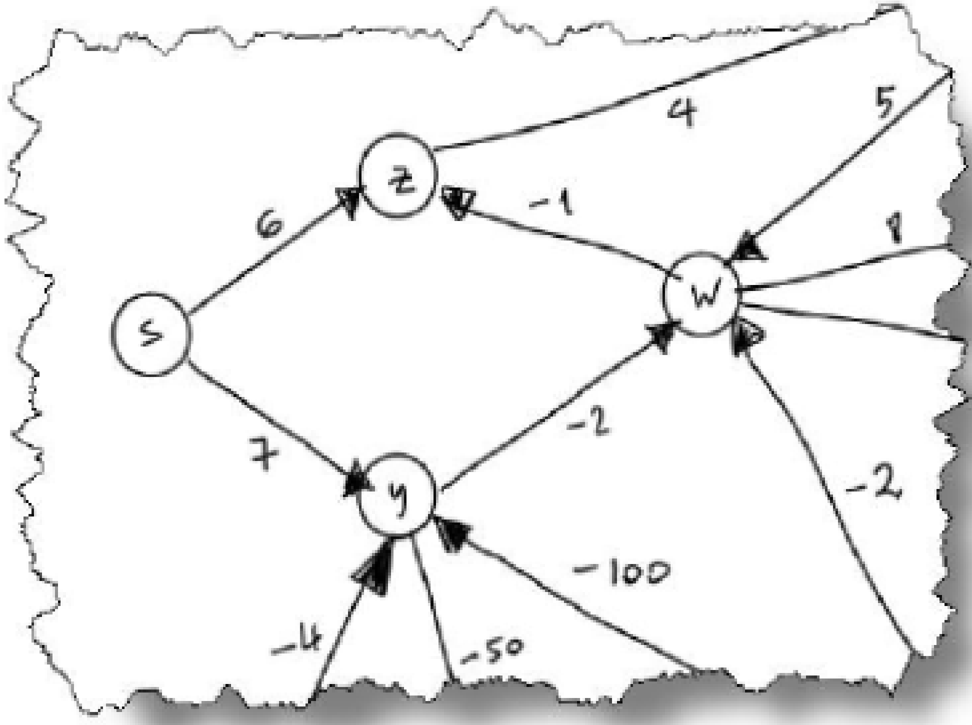


- Dijkstra(G^T, t)
- Dijkstra(G, s)
- $O(k)$

$E + V \log V + k$

$= O(E + V \log V)$

ESERCIZIO 1



Un importante problema di trasporto per l'azienda PIGA Petroli è stato ricondotto ad un problema di cammini minimi in un dato grafo orientato $G = (V, E)$, con funzione peso $w : E \to \mathbb{R}$, privo di cicli di peso negativo. In particolare, per effettuare le prime scelte strategiche, inizialmente serve solo calcolare la distanza in (G, w) dal nodo $s \in V$ al nodo $y \in V$. Di ciò è stato incaricato il brillante programmatore Turi N.G.[§] Questi preleva soltanto il frammento a lato dall'enorme rappresentazione grafica di (G, w) (si sappia infatti che $|V| = 10^8$) ed in men che non si dica calcola la distanza da s ad y e un cammino minimo da s ad y in (G, w).

Qual è il ragionamento effettuato dal nostro esperto programmatore Turi N.G.? *(Si applichino preferibilmente le proprietà sviluppate nel corso.)*
È possibile escludere con i dati in nostro possesso l'esistenza di altri cammini minimi da s ad y?
Che cosa si può dire della distanza di z da s?

[§]Per il rispetto della privacy sono state indicate solo le iniziali del cognome.

$\boxed{\delta(s,z) \leq 4}$

$$\frac{\delta(s,z) = 4}{\pi(w) < -3}$$

Sia $G = (V, E)$ un grafo orientato con funzione peso $w : E \to \mathbb{R}^+$ e siano $s_1, s_2 \in V$ due nodi distinti di G.

Un cammino $\pi = (v_0, v_1, \ldots, v_k)$ in G si dice $\{s_1, s_2\}$-MINIMO se

- $v_0 \in \{s_1, s_2\}$
- per ogni cammino π' da s_1 a v_k o da s_2 a v_k si ha: $w(\pi) \leq w(\pi')$.

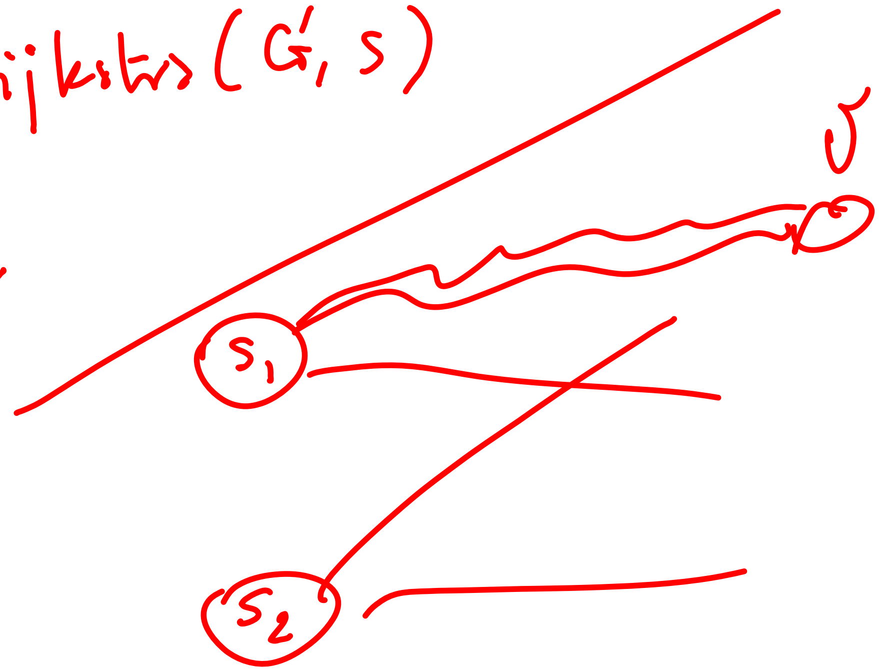
Inoltre, se $\pi = (v_0, v_1, \ldots, v_k)$ è un cammino $\{s_1, s_2\}$-minimo, poniamo $\delta_{\{s_1,s_2\}}(v_k) =_{Def} w(\pi)$.

Si illustri un algoritmo efficiente che per ciascun nodo v di G calcoli $\delta_{\{s_1,s_2\}}(v)$ nonché almeno un cammino $\{s_1, s_2\}$-minimo terminante in v.

Dijkstra(G', s)

$O(E + V \log V)$

$w(\pi) \leq w(\pi')$



$\boxed{\begin{array}{l} \text{Dijkstra}(G, s_1) \\ \text{Dijkstra}(G, s_2) \end{array}}$

$\Downarrow$

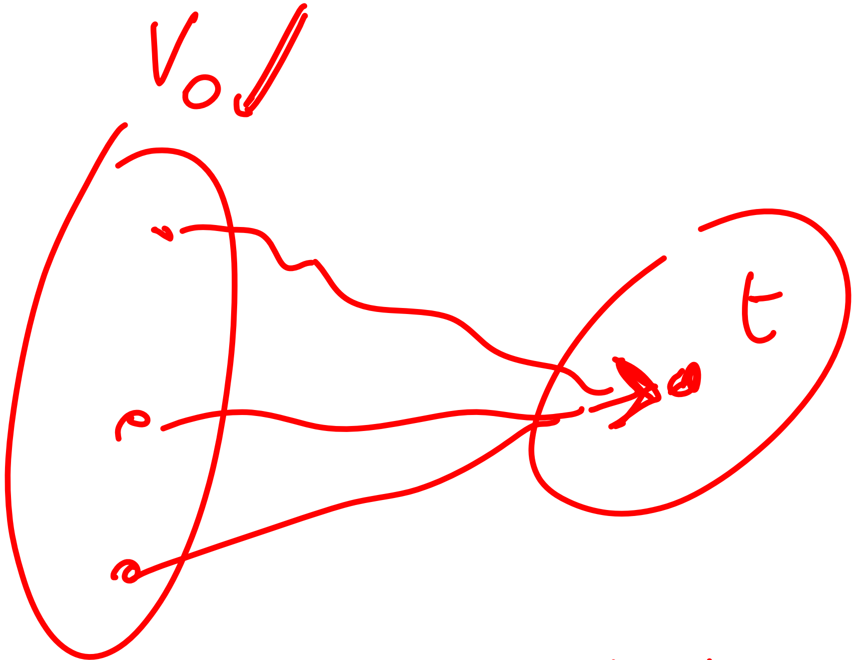
$\min(\delta(s_1, v_k), \delta(s_2, v_k))$

Sia dato un grafo orientato $G = (V, E)$ con funzione peso $w : E \longrightarrow \mathbb{R}^+$. Assegnati inoltre un insieme di nodi $V_0 \subsetneq V$ ed un nodo $t \in V \setminus V_0$, si ponga

$$\delta(V_0, t) =_{\text{Def}} \min_{v \in V_0} \delta_G(v, t),$$

dove δ_G è la funzione distanza in (G, w).

Si progetti un algoritmo per calcolare $\delta(V_0, t) = \min_{v \in V_0} \delta_G(v, t)$ e se ne valuti la complessità computazionale in funzione di $|V|$, $|V_0|$ e $|E|$.



Dijkstra(G^T, t)

$\boxed{\forall V_0, t}$

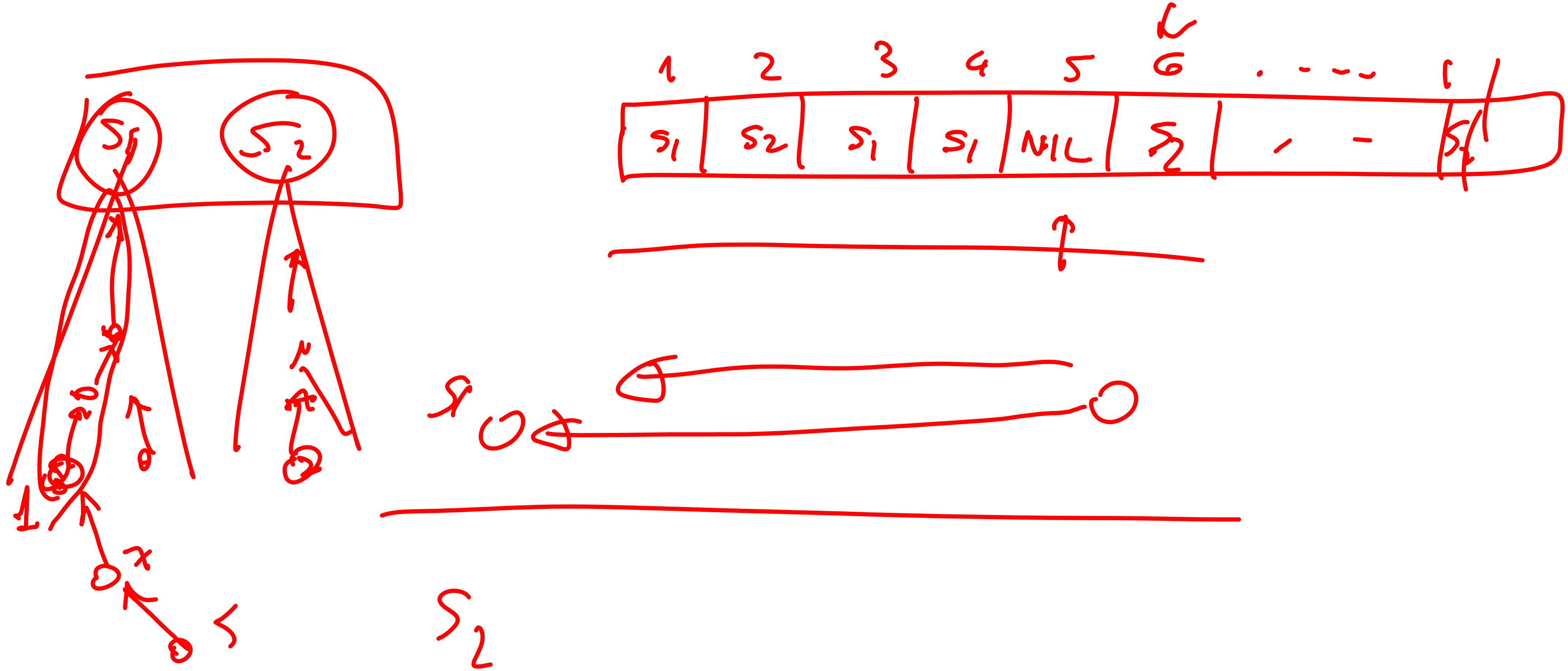
$|V|$ chiamate a Dijkstra, $O(2^{|V|}) \longrightarrow O((E + V \lg V) V 2^{|V|})$

$2^{|V|}$ chiamate a Dijkstra $\quad O((E + V \lg V) 2^{|V|})$

Sia $G = (V, E)$ un grafo orientato con funzione peso $w : E \longrightarrow \mathbf{R}^+$ e siano s_1 ed s_2 due nodi distinti di G.
Si descriva un algoritmo efficiente per calcolare una partizione (V_1, V_2) di V tale che:

- $\delta(s_1, u) \leq \delta(s_2, u)$, per ogni $u \in V_1$, e
- $\delta(s_2, v) \leq \delta(s_1, v)$, per ogni $v \in V_2$,

dove δ è la funzione distanza in (G, w).



Si descriva un algoritmo efficiente per determinare se un grafo pesato orientato contiene cicli di peso negativo, se ne calcoli la complessità e se ne verifichi la correttezza.

$O(V^2 E)$

$\cancel{O(V^4)}$

$O(V^3)$

$\boxed{O(VE)}$

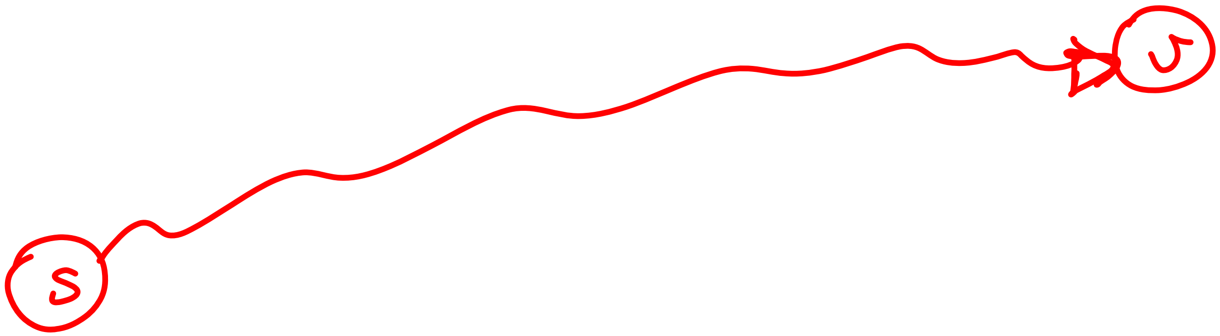
$E = o(V^2)$

$VE = o(V^3)$

RELAX(v_1, v_2)

RELAX(v_1, v_2)

Sia $G = (V, E)$ un grafo orientato con funzione peso $w : E \longrightarrow \mathbb{R}^+$, sorgente $s \in V$ e tale inoltre che contenga esattamente due nodi in V contrassegnati come “speciali”.
Diciamo che un cammino da u a v in G è *ammissibile* se esso contiene al più un solo nodo speciale (compresi gli estremi).
Si descriva un algoritmo per il calcolo dei cammini ammissibili minimi dalla sorgente s a tutti i nodi del grafo.



Si dia un algoritmo in grado di calcolare i cammini minimi nel seguente grafo dalla sorgente s a tutti i rimanenti nodi e si esibisca la traccia della relativa computazione. Inoltre si dimostri la correttezza dell'algoritmo dato e se ne valuti la complessità computazionale.