ALGORITMO DI JOHNSON PER GRAFI SPARSI

- COMPLESSITA': $O(V^2 \lg V + VE)$

- SE $E = o(V^2)$, ALLORA $V^2 \lg V + VE = o(V^3)$.
  INFATTI:

$$\lim_{V \to \infty} \frac{V^2 \lg V + VE}{V^3} = \lim_{V \to \infty} \frac{\lg V}{V} + \lim_{V \to \infty} \frac{E}{V^2} = 0$$

- UTILIZZA GLI ALGORITMI DI DIJKSTRA E DI BELLMAN-FORD COME SUBROUTINES

- SI BASA SULLA TECNICA DEL RIASSEGNAMENTO DEI PESI

CASO: FUNZIONE PESO A VALORI NON NEGATIVI

Multiple-DIJKSTRA(G, w)
1. let $D = (d_{uv})$ and $\Pi = (\pi_{uv})$ be two new $n \times n$ matrix
2. **for** each vertex $u \in V[G]$ **do**
3. run DIJKSTRA(G, w, u) to compute $\delta(u,v)$ and $Pred_u[v]$ for all $v \in V[G]$
4. **for** each vertex $v \in V[G]$ **do**
5. $d_{uv} := \delta(u,v)$
6. $\pi_{uv} := Pred_u[v]$
7. **return** D, Π

IMPLEMENTANDO L'ALGORITMO DI DIJKSTRA CON HEAP DI FIBONACCI SI OTTIENE UNA COMPLESSITA' DI $O(V^2 \lg V + VE)$

CASO GENERALE (ANCHE PESI NEGATIVI)

STRATEGIA:

- CALCOLARE NUOVI PESI $\hat{w}(u,v) \geq 0$ PER GLI ARCHI DI $G=(V,E)$ ED UTILIZZARE L'ALGORITMO Multiple-DIJKSTRA SUL GRAFO PESATO $(G,\hat{w})$.

- E' PERO' NECESSARIO CHE

$$\boxed{\begin{array}{l} \text{PER OGNI COPPIA } u,v \in V, \\ \text{UN CAMMINO } \pi \text{ DA } u \text{ A } v \text{ SIA MINIMO IN } (G,w) \\ \underline{\text{SE E SOLO SE}} \text{ E' MINIMO IN } (G,\hat{w}) \end{array}}$$

LEMMA

SIA $G=(V,E)$ UN GRAFO CON FUNZIONE PESO $w: E \to \mathbb{R}$ E SIA $h: V \to \mathbb{R}$ UNA FUNZIONE QUALUNQUE.

SI PONGA:

$$\hat{w}(u,v) = w(u,v) + h(u) - h(v)$$

PER OGNI $(u,v) \in E$.

ALLORA PER OGNI CAMMINO π IN G, π E' UN CAMMINO MINIMO IN (G,w) SE E SOLO SE π E' UN CAMMINO MINIMO IN $(G,\hat{w})$.

INOLTRE, (G,w) AMMETTE CAMMINI MINIMI SE E SOLO SE $(G,\hat{w})$ AMMETTE CAMMINI MINIMI.

DIM.

SIA $\pi = (v_0, v_1, ..., v_k)$ UN CAMMINO IN G.

ALLORA $\hat{w}(\pi) = w(\pi) + h(v_0) - h(v_k)$. INFATTI

$$\hat{w}(\pi) = \sum_{i=1}^{k} \hat{w}(v_{i-1}, v_i)$$

$$= \sum_{i=1}^{k} \Big(w(v_{i-1}, v_i) + h(v_{i-1}) - h(v_i) \Big)$$

$$= \sum_{i=1}^{k} w(v_{i-1}, v_i) + h(v_0) - h(v_k)$$

$$= w(\pi) + h(v_0) - h(v_k).$$

- SIA π UN CAMMINO MINIMO IN $(G, \hat{w})$ DA u A v.
SE π NON FOSSE MINIMO IN (G, w), ESISTEREBBE UN CAMMINO π' DA u A v TALE CHE $w(\pi') < w(\pi)$.
MA

$$\hat{w}(\pi') = w(\pi') + h(u) - h(v) < w(\pi) + h(u) - h(v) = \hat{w}(\pi),$$

CONTRADDICENDO LA MINIMALITÀ DI π IN $(G, \hat{w})$.
PERTANTO π È MINIMO IN (G, w).
- ANALOGAMENTE SI VERIFICA CHE SE π È MINIMO IN (G, w), ALLORA È MINIMO ANCHE IN $(G, \hat{w})$.

- INOLTRE, SE $\pi = (v_0, v_1, \dots, v_{k-1}, v_k)$ È UN CICLO, CON $v_0 = v_k$, SI HA

$$\hat{w}(\pi) = w(\pi) + h(v_0) - h(v_k) = w(\pi)$$

E PERTANTO (G, w) AMMETTE CAMMINI MINIMI SE E SOLO SE $(G, \hat{w})$ AMMETTE CAMMINI MINIMI ∎

COME SELEZIONARE LA FUNZIONE $h: V \to \mathbb{R}$ IN MODO TALE CHE RISULTI $\hat{w}(u,v) \geq 0$, PER OGNI ARCO $(u,v) \in E$?

- SI ESTENDA G CON UN NUOVO NODO s (CIOE' $V' = V \cup \{s\}$) E CON TUTTI GLI ARCHI (s,v), PER OGNI $v \in V$ (CIOE' $E' = E \cup \{(s,v) : v \in V\}$). SIA $G' = (V', E')$. SI ESTENDA w SUI NUOVI ARCHI PONENDO $w(s,v) = 0$, PER OGNI $v \in V$.

- OVVIAMENTE (G', w) AMMETTE CAMMINI MINIMI DA s SE E SOLO SE (G, w) AMMETTE CAMMINI MINIMI

- SUPPONIAMO CHE (G', w) AMMETTA CAMMINI MINIMI DA s. SI PONGA $h(v) = \delta_{G'}(s, v)$, PER OGNI $v \in V$.

ALLORA, PER OGNI ARCO $(u, v) \in E$, SE π_{su} E' UN CAMMINO MINIMO DA s A u, SI HA

$$\begin{aligned} \delta_{G'}(s, u) + w(u, v) &= w(\pi_{su}) + w(u, v) \\ &= w(\pi_{su}; (u, v)) \geq \delta_{G'}(s, v) \end{aligned}$$

QUINDI:

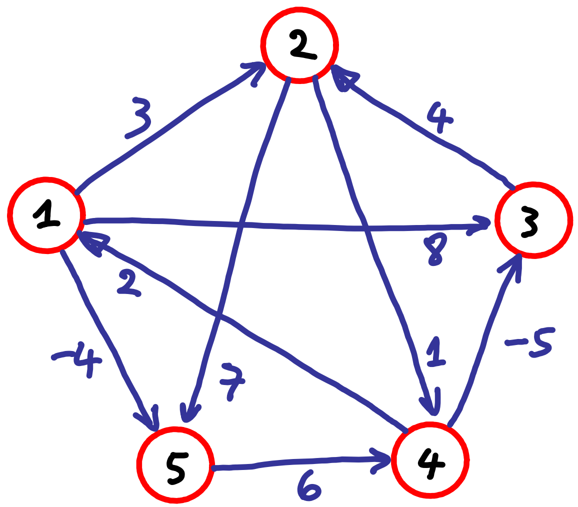
$$\hat{w}(u, v) = w(u, v) + \delta_{G'}(s, u) - \delta_{G'}(s, v) \geq 0. \quad \blacksquare$$

JOHNSON(G, w)

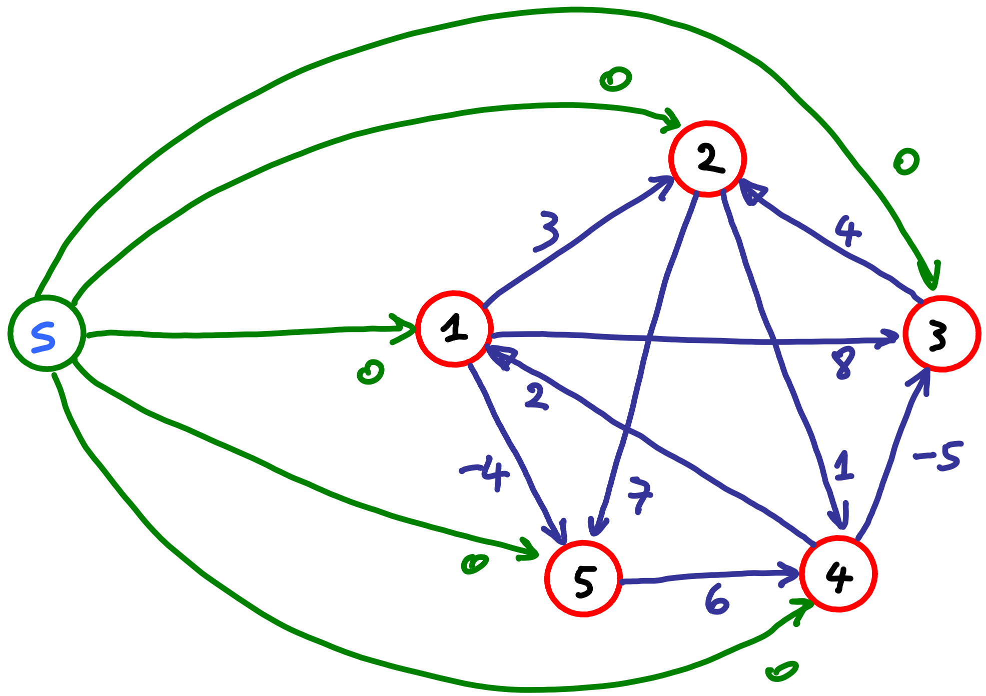
1. compute G', where $V[G'] = V[G] \cup \{s\}$,
   $E[G'] = E[G] \cup \{(s, v) : v \in V[G]\}$, and
   $w(s, v) = 0$ for all $v \in V[G]$
2. **if** BELLMAN-FORD(G', w, s) = FALSE **then**
3.   **print** "the input graph contains a negative-weight cycle"
4. **else**
5.   **for** each vertex $v \in V[G]$ **do**
           set $h(v)$ to the value of $\delta(s, v)$ computed by the
           Bellman-Ford algorithm
6.   **for** each edge $(u, v) \in E[G']$ **do**
7. $\hat{w}(u,v) := w(u,v) + h(u) - h(v)$
8.   let $D = (d_{uv})$ and $\Pi = (\pi_{uv})$ be two new $n \times n$ matrix
9.   **for** each vertex $u \in V[G]$ **do**
10.   run DIJKSTRA(G, $\hat{w}$, u) to compute
            $\hat{\delta}(u,v)$ and $\mathrm{Pred}_u[v]$ for all $v \in V[G]$
11.   **for** each vertex $v \in V[G]$ **do**
12. $d_{uv} := \hat{\delta}(u,v) - h(u) + h(v)$
13. $\pi_{uv} := \mathrm{Pred}_u[v]$
14.   **return** D, Π

ESEMPIO:

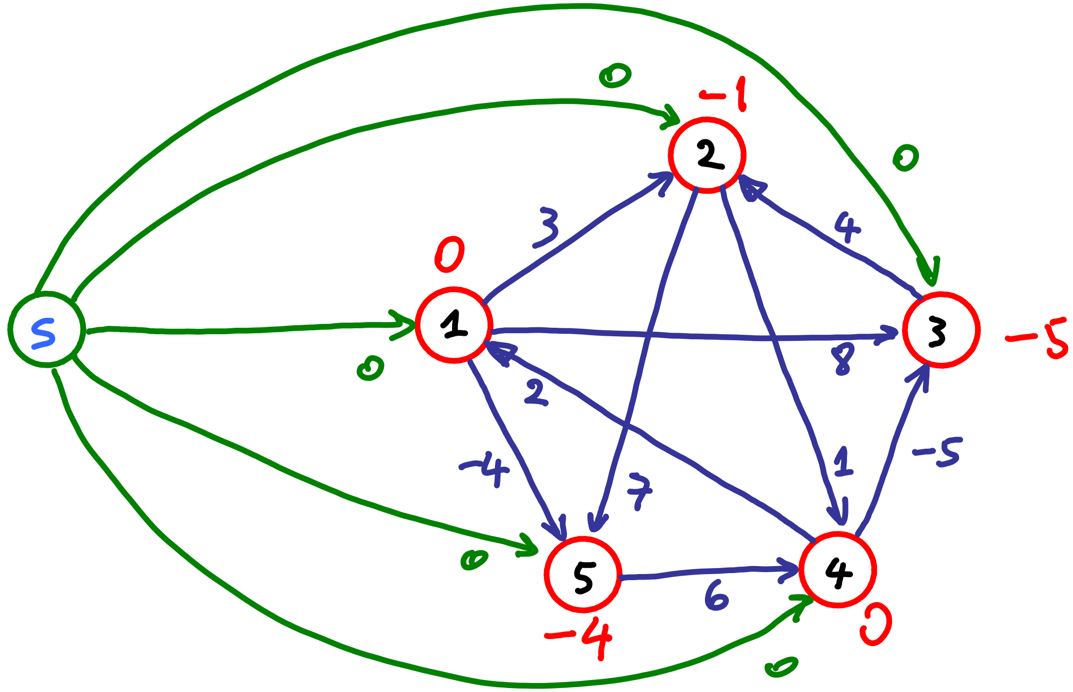
G :



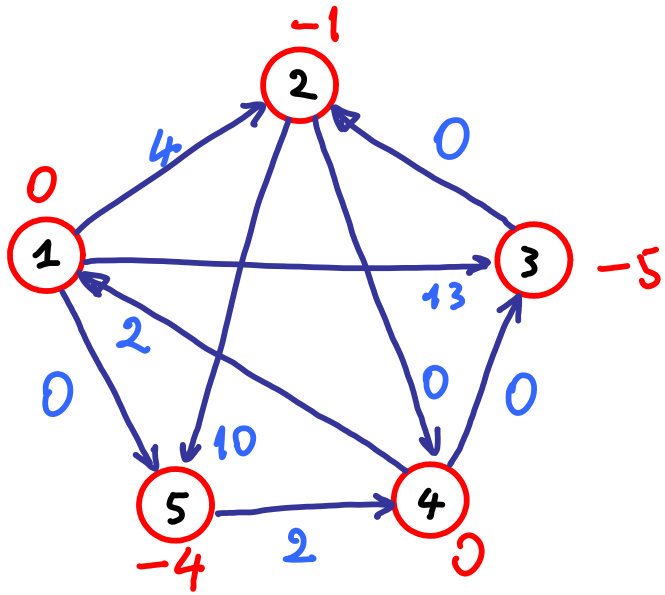
G' :

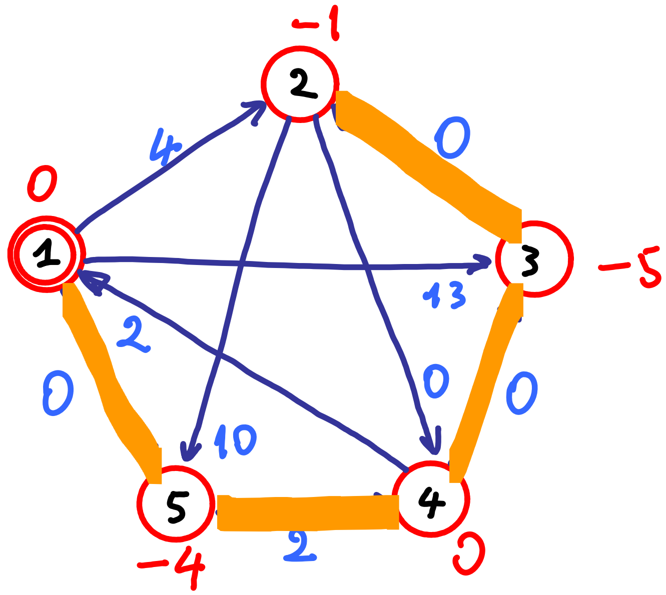


$\delta(s,v)$

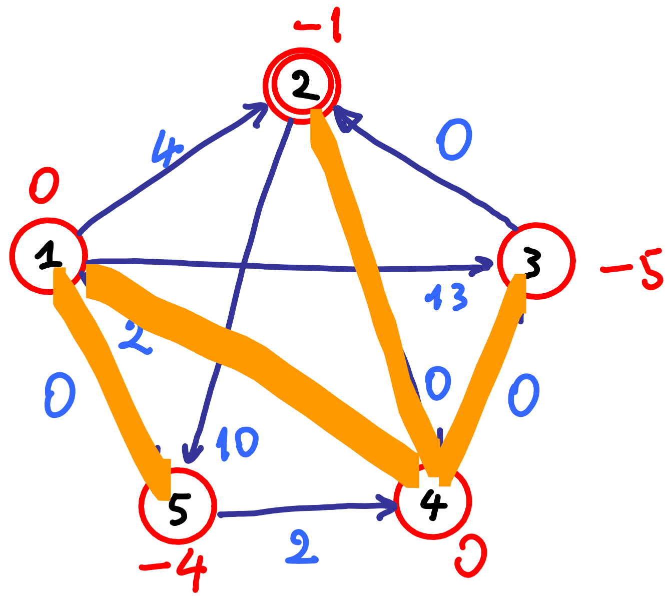


RIASSEGNAMENTO DEI PESI:

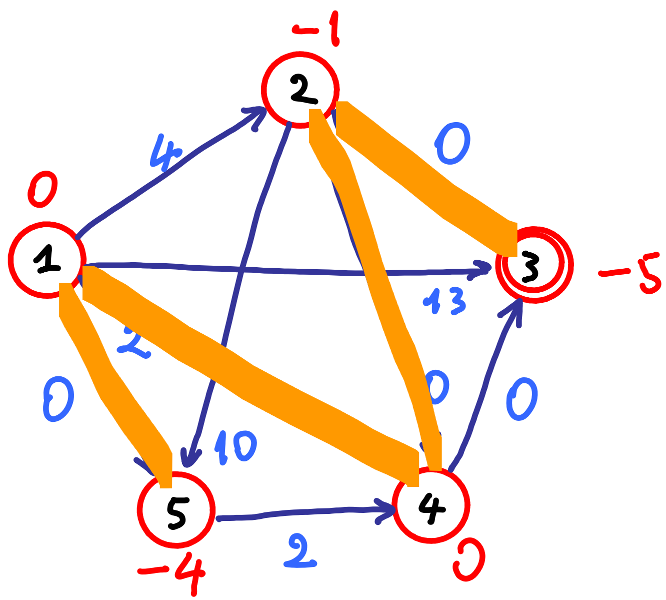




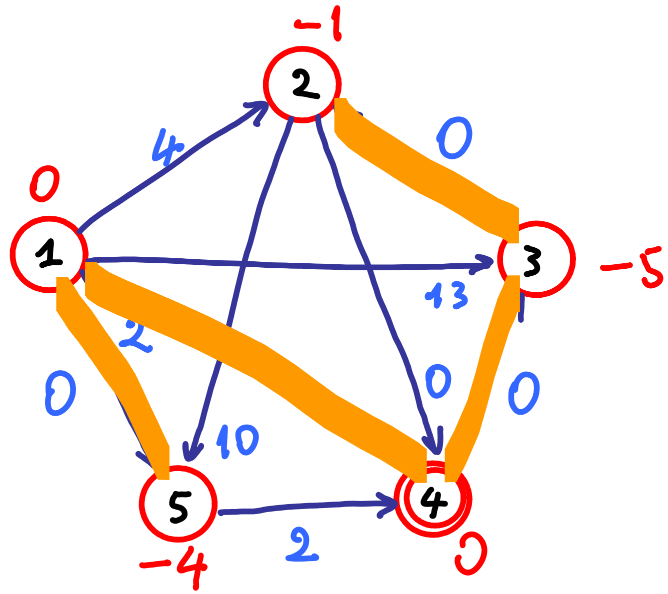
1
2
3
4
5
0
-1
-5
0
-4
4
0
13
2
0
0
10
0
2



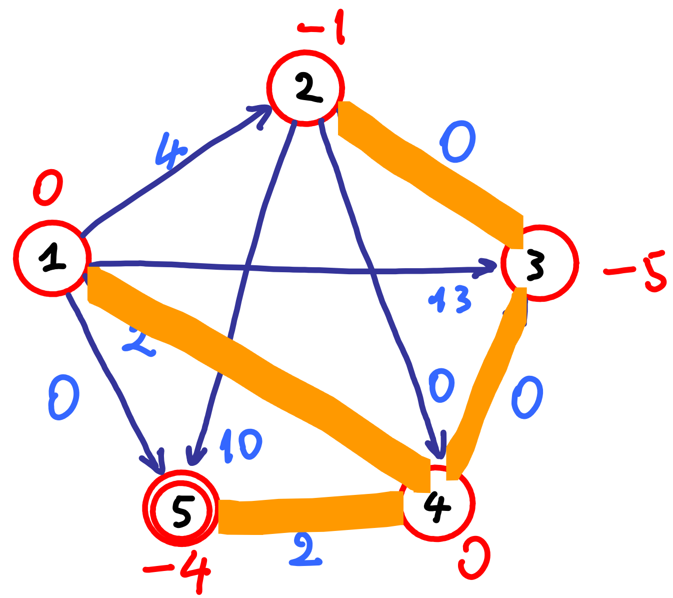
1
2
3
4
5
0
-1
-5
0
-4
4
0
13
2
0
0
10
0
2



1
2
3
4
5
0
-1
-5
0
-4
4
0
13
2
0
0
10
0
2



1
2
3
4
5
0
-1
-5
0
-4
4
0
13
2
0
0
10
0
2



1
2
3
4
5
0
-1
-5
0
-4
4
0
13
2
0
0
10
0
2