Cammini minimi tra tutte le coppie

Cammini minimi tra tutte le coppie

Dato un grafo (orientato o non orientato) $G = (V,E,W)$ con funzione di peso $w: E \rightarrow \mathbf{R}$, trovare per ogni coppia di vertici $u,v \in V$ il minimo peso di un cammino da u a v.

Verrà calcolata anche una matrice di predecessori $\Pi(\pi_{uv})$ dove π_{uv} è NIL se $u=v$ o se non c'è un cammino da u a v, altrimenti è un predecessore di v su di un cammino minimo da u.

Il sottografo indotto dall'i-esima riga della matrice Π sarà un albero di cammini minimi con radice in i.

Algoritmo di Floyd-Warshall

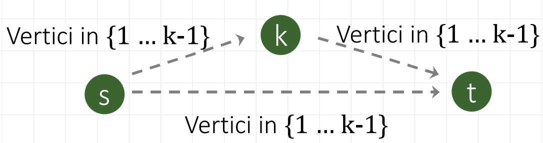
E' un algoritmo di programmazione dinamica, può gestire archi di peso negativo ma si assume che non ci siano cicli negativi.

Idea:

$d_{s,t}(i)$: cammino minimo da s a t contenente solo i vertici intermedi $v_1, \ldots, v_i$

$d_{s,t}(0) = w(s,t)$

Floyd-Warshall, idea



$d_{s,t}(k\text{-}1)$: cammino minimo da s a t contenente solo i vertici intermedi $v_1, \ldots, v_{k-1}$

$d_{s,t}(0) = w(s,t)$

$$d_{s,t}(k) = \begin{cases} w(s,t) & \text{se } k = 0 \\ \min\{d_{s,t}(k-1),\ d_{s,k}(k-1) + d_{k,t}(k-1)\} & \text{se } k > 0 \end{cases}$$

k non è nel cammino minimo s-t

k è nel cammino minimo s-t

Determinazione Π

Π è ottenibile da D in $O(n^3)$.

Oppure, "on line" (mentre si aggiorna la D):

$$\pi_{ij}(0) = \begin{cases} \text{NIL} & \text{se } i=j \text{ o } w_{ij} = \infty \\ i & \text{se } i \neq j \text{ e } w_{ij} < \infty \end{cases}$$

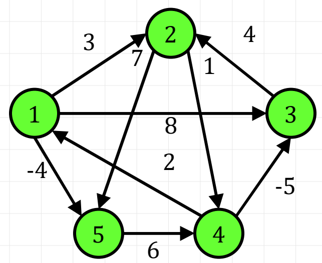
poi

$$\pi_{ij}(k) = \begin{cases} \pi_{ij}(k-1) & \text{se } d_{ij}(k-1) \leq d_{ik}(k-1) + d_{kj}(k-1) \\ \pi_{kj}(k-1) & \text{se } d_{ij}(k-1) > d_{ik}(k-1) + d_{kj}(k-1) \end{cases}$$

Algoritmo di Floyd-Warshall

```
Floyd-Warshall(W)
n=rows[W]
D(0)=W
for k = 1 to n do
  for i = 1 to n do
    for j = 1 to n do
      d_ij(k) = min(d_ij(k-1), d_ik(k-1) + d_kj(k-1))
return D(n)
```

Floyd-Warshall, esempio



W

0	3	8	∞	-4
∞	0	∞	1	7
∞	4	0	∞	∞
2	∞	-5	0	∞
∞	∞	∞	6	0

Floyd-Warshall, esempio

$D^{(0)}$

0	3	8	∞	-4
∞	0	∞	1	7
∞	4	0	∞	∞
2	∞	-5	0	∞
∞	∞	∞	6	0

$\Pi^{(0)}$

	1	1		1
			2	2
	3			
4		4		
			5	

Floyd-Warshall, esempio

$D^{(1)}$

0	3	8	∞	-4
∞	0	∞	1	7
∞	4	0	∞	∞
2	5	-5	0	-2
∞	∞	∞	6	0

$\Pi^{(1)}$

	1	1		1
			2	2
	3			
4	1	4		1
			5	

Floyd-Warshall, esempio

$D^{(2)}$

0	3	8	4	-4
∞	0	∞	1	7
∞	4	0	5	11
2	5	-5	0	-2
∞	∞	∞	6	0

$\Pi^{(2)}$

	1	1	2	1
			2	2
	3		2	2
4	1	4		1
			5	

Floyd-Warshall, esempio

$D^{(3)}$

0	3	8	4	-4
∞	0	∞	1	7
∞	4	0	5	11
2	-1	-5	0	-2
∞	∞	∞	6	0

$\Pi^{(3)}$

	1	1	2	1
			2	2
	3		2	2
4	3	4		1
			5	

Floyd-Warshall, esempio

$D^{(4)}$

0	3	-1	4	-4
3	0	-4	1	-1
7	4	0	5	3
2	-1	-5	0	-2
8	5	1	6	0

$\Pi^{(4)}$

	1	4	2	1
4		4	2	1
4	3		2	1
4	3	4		1
4	3	4	5	

Floyd-Warshall, esempio

$D^{(5)}$

0	3	-3	2	-4
3	0	-4	1	-1
7	4	0	5	3
2	-1	-5	0	-2
8	5	1	6	0

$\Pi^{(5)}$

	1	4	5	1
4		4	2	1
4	3		2	1
4	3	4		1
4	3	4	5	

Floyd-Warshall: complessitá

Determinata dai tre cicli for.
Ogni esecuzione dell'istruzione interna è O(1), quindi:

$$T(V,E) = \Theta(n^3) = \Theta(V^3)$$

Stampa dei cammini

Stampa di un cammino fra una coppia di nodi:

```
Print-all-pairs-shortest-path(Π,i,j)
if (i=j)
then STAMPA i
else if π_ij = NIL
     then STAMPA “Non esiste il cammino”
     else Print-all-pairs-shortest-path(Π,i, π_ij)
          STAMPA j
```