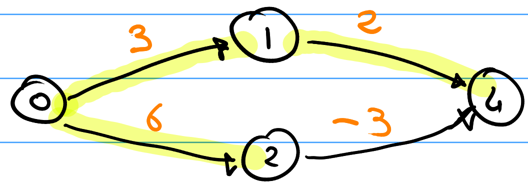
Cammini minimi e archi negativi

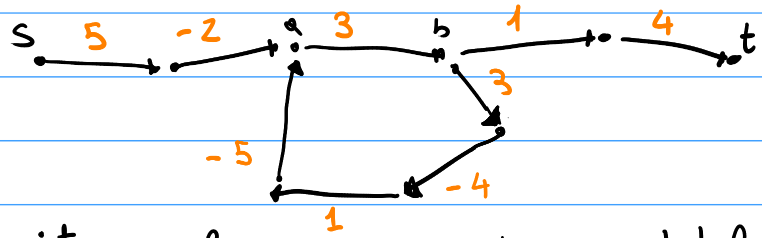


- Dijkstra su questo grafo, a partire da ⓪, produce un cammino subottimale per ④. Quando ④ esce dalla coda di priorità, il cammino minimo scoperto ha peso $3+2=5$.

- Il vertice ② con è nella coda con un valore ≥ 5 quindi non può (nell'assunzione di pesi ≥ 0) permettere di raggiungere ④ in meno di peso 5

Vedremo quindi algoritmi che gestiscono archi negativi

- La presenza di archi negativi può causare un problema: i CICLI NEGATIVI



Tra s e t esiste un solo cammino, di peso totale 11. Essendo l'unico, è il minimo.

Tuttavia vediamo che la nozione di cammino minimo non modella più il minimo costo da pagare per andare da s a t.

È possibile usare una passeggiata $s \rightsquigarrow a$, $a \circlearrowleft$, $a \to b$, $b \rightsquigarrow t$ ed il costo è 9.

Percorrendo t volte il ciclo si ha costo $11 - 2 \cdot t$

Estendiamo la definizione di $\delta(s,v) = \begin{cases} +\infty & \text{se } v \text{ non è raggiungibile da } s \\ -\infty & \text{se esiste } s \rightsquigarrow u \rightsquigarrow v \text{ con } u \text{ coinvolto in un ciclo negativo} \\ \min\{w(p) : p \text{ cammino da } s \text{ a } v\} & \text{altrimenti} \end{cases}$

Vedremo un algoritmo che permette di gestire archi negativi e riconoscere se ci sono CICLI NEGATIVI nel grafo

Ora peró vediamo un caso piú semplice

- G con archi negativi MA <u>aciclico</u>

- In questo caso sará possibile calcolare i cammini minimi in maniera efficiente. Anche questo algoritmo é basato sul RILASSAMENTO DI UNA SEQUENZA DI ARCHI.

Sorgente

1. Abbiamo $P = [NIL, NIL, \dots NIL]$ $dist = [+\infty, +\infty, \dots, 0, +\infty, +\infty]$
2. Ad ogni passo scegliamo un arco (u,v) e lo rilassiamo

```
if dist[v] > dist[u] + W(u,v):
    dist[v] = dist[u] + W(u,v)
    P[v] = u
```

<u>OSS</u>. Dijkstra é un algoritmo di questo tipo

Proprietá dei cammini minimi e degli algoritmi per rilassamento

- Disuguaglianza Triangolare (Lemma 24.10)

per ogni arco (u,v) $\delta(s,v) \leq \delta(s,u) + W(u,v)$

- Proprietá del limite superiore (Lemma 24.11)
  durante l'algoritmo $dist[v] \geq \delta(s,v)$ e quando $dist[v] = \delta(s,v)$ questo non cambia piú

- Proprietà della convergenza (Lemma 24.14)

  Se $s \rightsquigarrow u \rightarrow v$ è un cammino minimo in G, e se (u,v) viene rilassato in un momento in cui $dist[u] = \delta(s,u)$ allora $dist[v] = \delta(s,v)$ da quel momento in poi

- Proprietà del rilassamento del cammino (Lemma 24.15)

  - Sia $\overset{s}{\overset{\shortparallel}{v_0}} e_1 v_1 e_2 v_2 e_3 v_3 \dots e_\ell v_\ell$ un cammino di costo $\delta(s, v_\ell)$
  - Siano (v_0, v_1), (v_1, v_2) (v_2, v_3) ..... archi che vengono rilassati in quest'ordine

  Allora dopo il rilassamento di (v_{i-1}, v_i) abbiamo $dist[v_i] = \delta(s, v_i)$
  - questa proprietà vale anche se altri passi di rilassamento sono interposti nella sequenza

- Proprietà del sottografo dei predecessori (Lemma 24.17)

  Una volta che $dist[v] = \delta(s,v)$ $\forall v \in V$, allora P è
  - un albero radicato in s
  - contiene un cammino minimo $s \rightsquigarrow v$ di costo $\delta(s,v)$ per ogni v raggiungibile da s

ALCUNE DI QUESTE PROPRIETÀ LE ABBIAMO VISTE INFORMALMENTE

ESERCIZIO Dimostrare, o leggere la dimostrazione, di queste proprietà

Cammini Minimi per DAG

- Un semplice algoritmo per i cammini minimi deriva dalla proprietà del rilassamento del cammino

Considerate il seguente algoritmo:

Input $G = (V, E, W)$ diretto aciclico e $s \in V$
output P, dist

① Inizializza $P = [NIL, \dots NIL]$ dist$[+\infty, +\infty, \dots, \overset{s}{0}, \dots +\infty\ +\infty]$ — $O(|V|)$
② Calcola un ordinamento topologico di G, $v_1, v_2, \dots, v_n$ — $O(|V|+|E|)$
③ Per ogni $i = 1 \dots n$
  per ogni arco $(v_i, w) \in E$
    rilassa (v_i, w)

$\Big\}$ — $O(|V|+|E|)$

La correttezza di questo algoritmo deriva dal fatto che

- qualunque cammino minimo percorre vertici in ordine topologico.
- quando si rilassa un arco (v_i, w) tutti gli archi che portano a v_i sono già stati rilassati perché provengono da vertici già analizzati

- Per induzione: all'inizio del passo i, $dist[v_i] = \delta(s, v_i)$
  - Caso base. Se $s = v_j$ allora $dist[v_i] = +\infty$ per $i < j$
    $dist[s] = 0$
  - passo induttivo: se v_i non è raggiungibile da s, $dist[v_i] = +\infty$
    (Prop. del limite superiore)

    se esiste un cammino $s \rightsquigarrow v_{i'} \to v_i$ allora $i' < i$
    e quindi $dist[v_{i'}] = \delta(s, v_{i'})$ al passo i' quando $(v_{i'}, v_i)$ è stato rilassato. Perciò $dist[v_i] = \delta(s, v_i)$.
    (Prop. convergenza)

$\boxed{\text{Visualizza Esempio Precalcolato}}$

CAMMINI MINIMI PER GRAFI CON ARCHI NEGATIVI

Abbiamo visto
- Dijkstra — solo archi non negativi
- Rilassamento rispetto all'ordinamento topologico — nessun ciclo

se ci possono essere archi negativi e cicli, ci possono essere cicli negativi.

ALGORITMO DI BELLMAN-FORD

- Dato un grafo $G=(V,E,w)$ pesato e orientato e una sorgente s l'algoritmo è un algoritmo per rilassamento e alla fine
  - segnala CICLO NEGATIVO se la sorgente s può raggiungere un vertice coinvolto in un ciclo negativo
  - altrimenti $dist[v]=\delta(s,v)$ per ogni v e P codifica l'albero dei cammini minimi

```
BellmanFord(s, G, W):
    P = [NIL, ... NIL]   dist = [+∞, +∞, ..., 0, +∞, +∞]     (0 in posizione s)
    for i = 1 a |V(G)| - 1                      ] Per n-1 volte vengono rilassati tutti gli archi
        for (u,v) ∈ E:
            if dist[v] > dist[u] + W(u,v)       } rilassamento di (u,v)
                dist[v] = dist[u] + W(u,v)
                P[v] = u
    for (u,v) in E:                             ] Verifica dei cicli negativi
        if dist[v] > dist[u] + W(u,v)
            segnala CICLO NEGATIVO
```

Complessità $O(|V|\cdot|E|)$

Dimostrazione di correttezza

- Caso in cui s non raggiunge cicli negativi

  - I vertici v coinvolti in cicli negativi hanno tutti $dist[v] = +\infty$ e un qualunque rilassamento tra loro non cambia questa cosa quindi possiamo ignorarli

  - Se s raggiunge un vertice v, poiché non ci sono cicli negativi,

    $$\delta(s,v) = \text{minimo costo tra tutti i cammini } s \rightsquigarrow v$$

    Sia p un cammino minimo, questo è fatto da $n-1$ archi

    Perciò questi archi costituiscono una sottosequenza della sequenza di archi rilassati. Per la proprietà del rilassamento del cammino, $dist[v] = \delta(s,v)$ al termine della procedura

- Caso in cui s raggiunge un ciclo $v_0 \to v_1 \to v_2 \to v_3 \cdots v_{k-1} \to v_k$ di peso negativo $v_k = v_0$

- Tutti i vertici $v_0 \dots v_{k-1}$ sono raggiungibili con un cammino di lunghezza al massimo $n-1$, e quindi è possibile dimostrare che $dist[v_i] < +\infty$ per $i = 0 \dots k-1$ $\boxed{\text{Esercizio } 24.2\text{-}2}$

Se fosse vero che $dist[v_{i+1}] \leq dist[v_i] + w(v_i, v_{i+1})$

avremmo

$$\underbrace{\sum_{i=0}^{k-1} dist[v_{i+1}]}_{A} \leq \underbrace{\sum_{i=0}^{k-1} dist[v_i]}_{A} + \underbrace{\sum_{i=0}^{k-1} w(v_i, v_{i+1})}_{B}$$

il che implicherebbe $B \geq 0$, che sarebbe una contraddizione.

Vediamo l'esempio precalcolato

Vedendo l'esempio su un grafo osserviamo che il fatto che esista un CICLO NEGATIVO non pregiudica l'intero calcolo

A) • I vertici irraggiungibili da s hanno dist $= +\infty$

B) • I vertici v per cui nessun cammino da s a v tocca un CICLO NEGATIVO hanno in effetti $dist[v] = \delta(s,v)$

C) • Per i vertici che non sono in A e in B, abbiamo che dist continua a scendere ad ogni iterazione

ESERCIZIO 24.2-4

Modificate l'algoritmo di Bellman-Ford in modo tale che alla fine

- Per ogni vertice v di tipo A, $dist[v] = +\infty$
- Per ogni vertice v di tipo B, $dist[v] = \delta(s,v)$
- Per ogni vertice v di tipo C, $dist[v] = -\infty$

INDIZI

- La dimostrazione di correttezza fa vedere che in OGNI CICLO NEGATIVO raggiungibile da s, c'è almeno un arco (u,v) con $dist[v] > dist[u] + W[u,v]$ alla fine dell'algoritmo.
- Perciò i vertici C sono tutti quelli raggiungibili da almeno uno di questi.