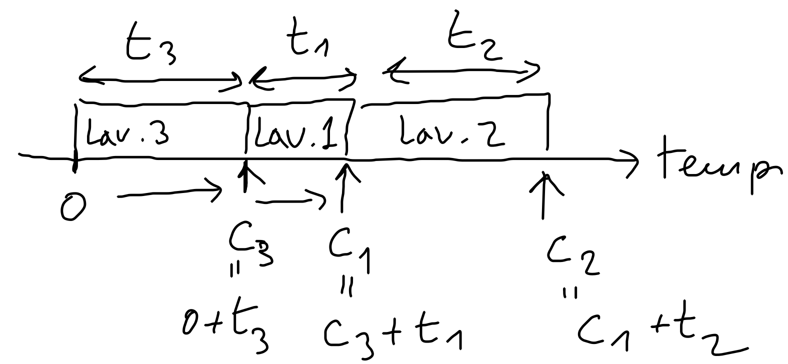
Es. 4.13 "Lavori" $1, 2, 3, \ldots, n$

Tempi $t_1, t_2, t_3, \ldots, t_n$ (interi positivi)

Pesi $w_1, w_2, w_3, \ldots, w_n$ (interi positivi)

Tempi di completamento dei lavori: dipendono dall'ordine dei lavori

Esempio: 3 lavori
Possibile ordinamento
3, 1, 2 →



C_j è il tempo di completamento del lavoro j

Se j è eseguito subito dopo i, allora $C_j = C_i + t_j$

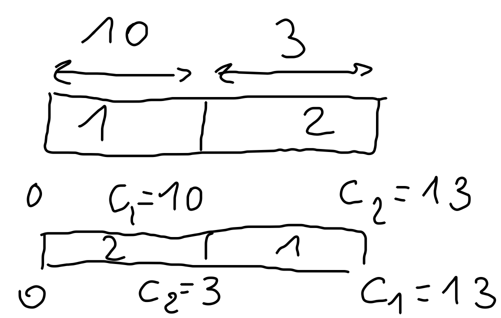
Obiettivo: trovare un ordinamento che minimizzi $\sum_{j=1}^{n} w_j \cdot C_j$

Esempio:

$t_1 = 10$ $t_2 = 3$

$w_1 = 5$ $w_2 = 3$

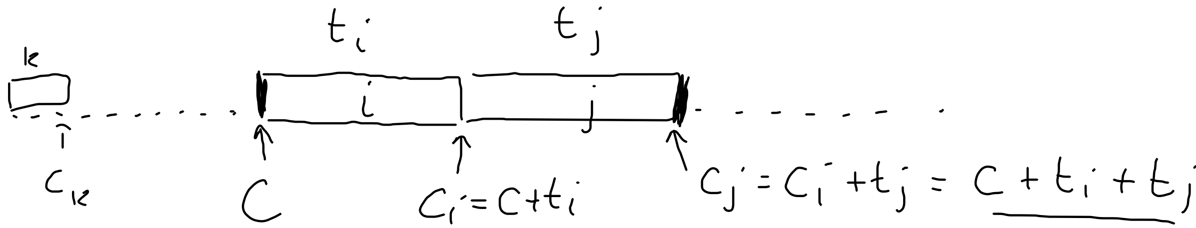
$\frac{w_1}{t_1} = \frac{1}{2}$ $\frac{w_2}{t_2} = 1$



costo $= 5 \cdot 10 + 3 \cdot 13 = 50 + 39 = 89$

costo $= 5 \cdot 13 + 3 \cdot 3 = 65 + 9 = 74$

Soluzione generica



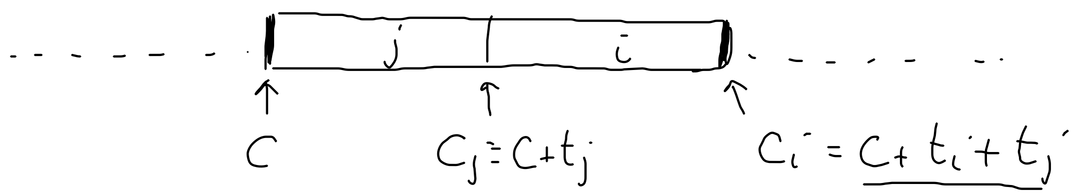
$$\frac{w_{37}}{t_{37}} \geq \frac{w_{21}}{t_{21}} \geq \dots \geq \dots$$

$$Costo^{(1)} = \sum_{k \neq i,j} w_k C_k + w_i C_i + w_j C_j$$

$$= \sum_{k \neq i,j} w_k C_k + w_i (C + t_i) + w_j (C + t_i + t_j)$$

$$= \sum_{k \neq i,j} w_k C_k + \cancel{w_i C} + \cancel{w_i t_i} + \cancel{w_j C} + w_j t_i + \cancel{w_j t_j}$$

Soluzione alternativa con i e j scambiati



$$Costo^{(2)} = \sum_{k \neq i,j} w_k C_k + w_j C_j + w_i C_i$$

$$= \sum_{k \neq i,j} w_k C_k + w_j (C + t_j) + w_i (C + t_i + t_j)$$

$$= \sum_{k \neq i,j} w_k C_k + \cancel{w_j C} + \cancel{w_j t_j} + \cancel{w_i C} + \cancel{w_i t_i} + w_i t_j$$

$Costo^{(1)} \leq Costo^{(2)}$?

$$\Updownarrow$$

$$w_j t_i \leq w_i t_j$$

$$\Updownarrow$$

$$\frac{w_j}{t_j} \leq \frac{w_i}{t_i}$$

Algoritmo:

Fase 1 — Calcolo w_k/t_k per ogni $k = 1, 2, \dots, n$ (totale tempo $O(n)$)

Fase 2 — Ordina i lavori per valore del rapporto decrescente

(tempo $O(n \log n)$)

---

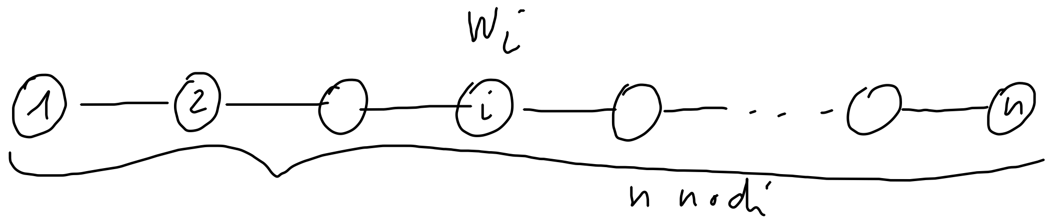
tempo complessivo

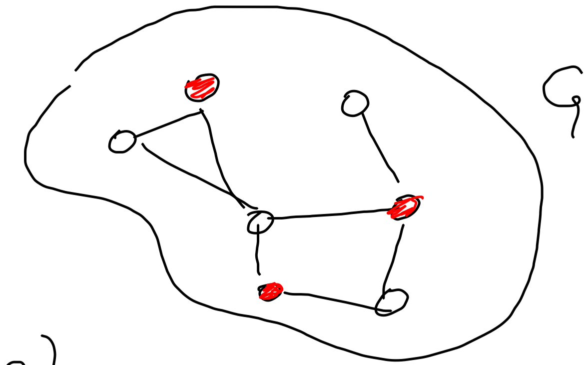
$$\underbrace{O(n + n \log n)} = O(n \log n)$$

$$O(f+g) \equiv O(\max(f,g))$$

$$O(f(n)+g(n)) = O(\max(f(n), g(n)))$$

Es. 6.1.

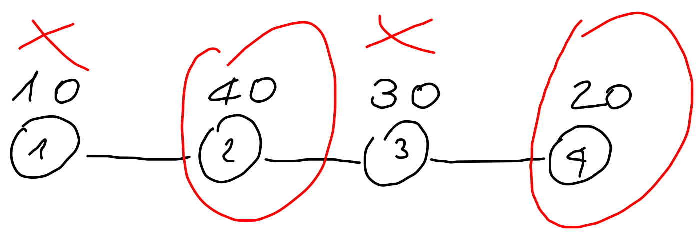




Catena di n nodi, ciascuno con un peso w_i (intero positivo)

Obiettivo: trovare un insieme indipendente di nodi, di peso totale massimo.

Esempio:



{1,3} $10+30=40$

{1,4} $10+20=30$

{2,4} $40+20=60$

~~{1}~~

~~{2}~~

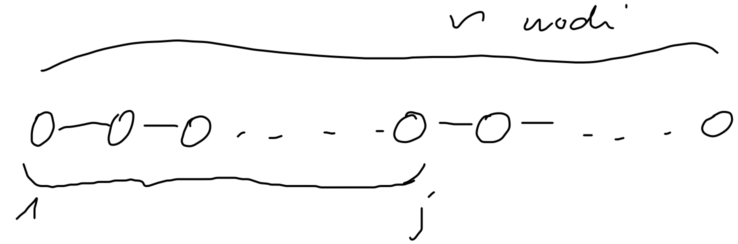
~~{3}~~

Sol. ottima è {2,4} (valore 60).
(peso totale)

(a) Mostrare che un approccio avido basato sullo scegliere il nodo di peso massimo non funziona

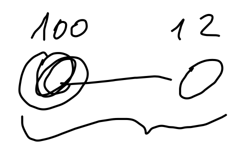
(b) Mostrare che dividere i nodi nell'insieme dei nodi di indice pari e dei nodi di indice dispari e restituire la migliore tra le due soluzioni, non funziona.

Sia OPT(j) il valore (peso totale) delle soluzione ottima per i primi j nodi

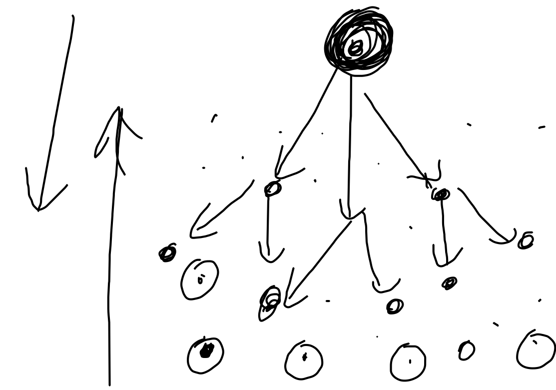


$$
\begin{cases}
OPT(j) = \max \begin{cases} OPT(j-1) & \text{(se } j \text{ non è scelto)} \\ OPT(j-2) + w_j & \text{(se } j \text{ è scelto)} \end{cases}
\end{cases}
$$

$OPT(1) = w_1$



$OPT(2) = 100$



Pseudocodice:

$O(1)$ OPT[0] ← 0

$O(1)$ OPT[1] ← w_1,

For $j = 2, \ldots, n$:

$O(n)$ $\{ (n-1) \times O(1)$ → OPT[j] = max(OPT(j-1), w_j + OPT(j-2))

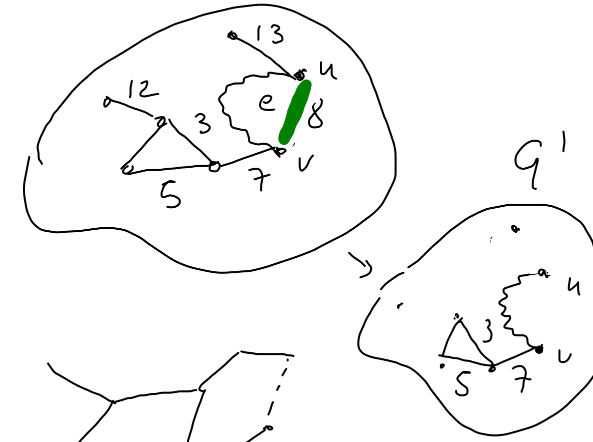
Totale $O(n)$

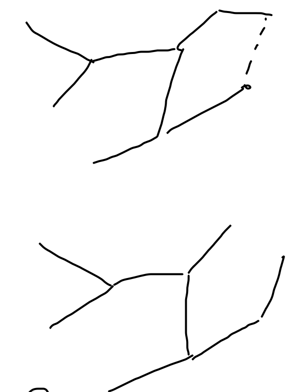
Es. 4.3

Grafo G connesso, con pesi sugli archi, tutti distinti tra loro.

Input: G (pesato) + un arco e di G

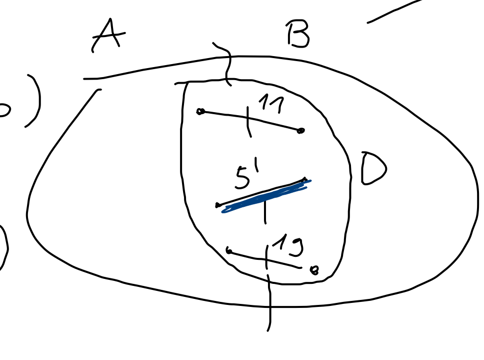
Domanda: esiste un albero ricoprente minimo del grafo che contiene l'arco e?

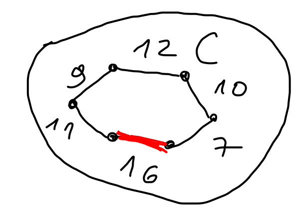




Regola blu: Se D è un cutset (e nessun arco di D è blu), scegli un arco non colorato di D a costo minimo e coloralo di blu (includilo nella soluzione)

Regola rosso: Se C è un ciclo (e nessun arco di C è rosso) scegli un arco non colorato di C a costo massimo e coloralo di rosso (escludilo dalla soluzione)





Costruisco un secondo grafo G' ottenuto da G rimuovendo l'arco $e=(u,v)$.

tutti gli archi con peso maggiore del peso di e.

Se u e v sono connessi in G'

$\Rightarrow$ in G esiste un ciclo contenente l'arco e in cui tutti gli altri archi hanno peso $<$ peso(e)

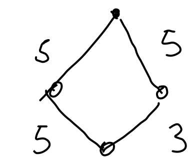
$\Rightarrow$ e non fa parte di nessun albero ricoprente minimo

Se u e v non sono connessi in G' ?

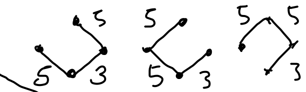
Esiste un cutset D in cui "e" è l'arco a peso minimo

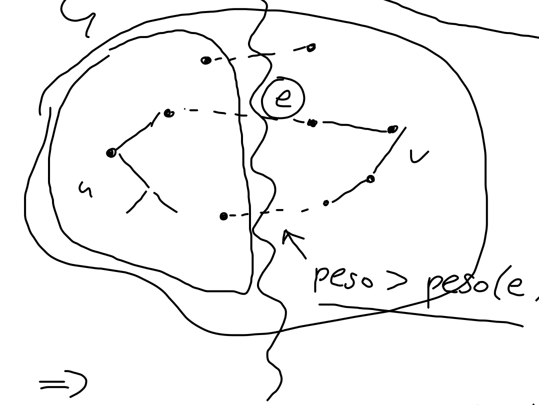
$\Rightarrow$ e fa parte di un albero ricoprente minimo

Controesempio:



ha 3 alberi ricoprenti minimi:





$\Rightarrow$ Posso applicare la regola blu

Algoritmo: costruisci G' a partire da G $\leftarrow O(m)$

Tempo $O(m)$ per una visita di G' (DFS, BFS, ...)

- Se u e v sono connessi in G', la risposta è <u>NO</u>
- Se u e v non sono connessi in G', la risposta è <u>SI</u>

Totale $O(m)$