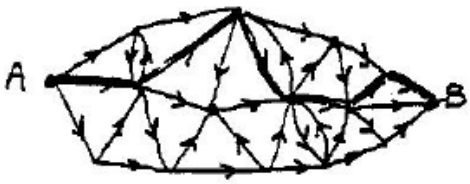
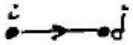
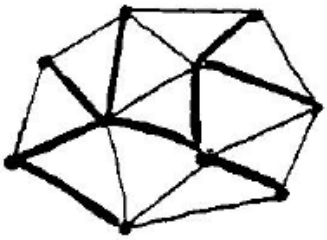
PROBLEMI COMBINATORI (su grafo)

In molti problemi il numero di soluzioni ammissibili è finito. Questi problemi sono quasi sempre descritti su grafi.



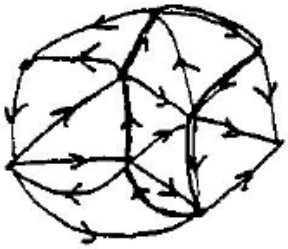
Rete stradale: come andare da A a B in tempo minimo?

- Grafo orientato 
- t_{ij} = tempo per andare da i a j



Rete fognaria: qual è la rete di raccolta a costo minimo?

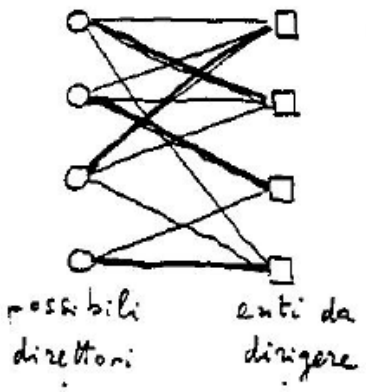
- Grafo non orientato 
- c_{ij} = costo di costruzione del collegamento (i, j)



Percorso circolare: qual è il percorso circolare di un autobus che trasporta in media più passeggeri ?

t_{ij} = tempo n_{ij} = passeggeri

$$\frac{\sum n_{ij}}{\sum t_{ij}} \rightarrow \max$$

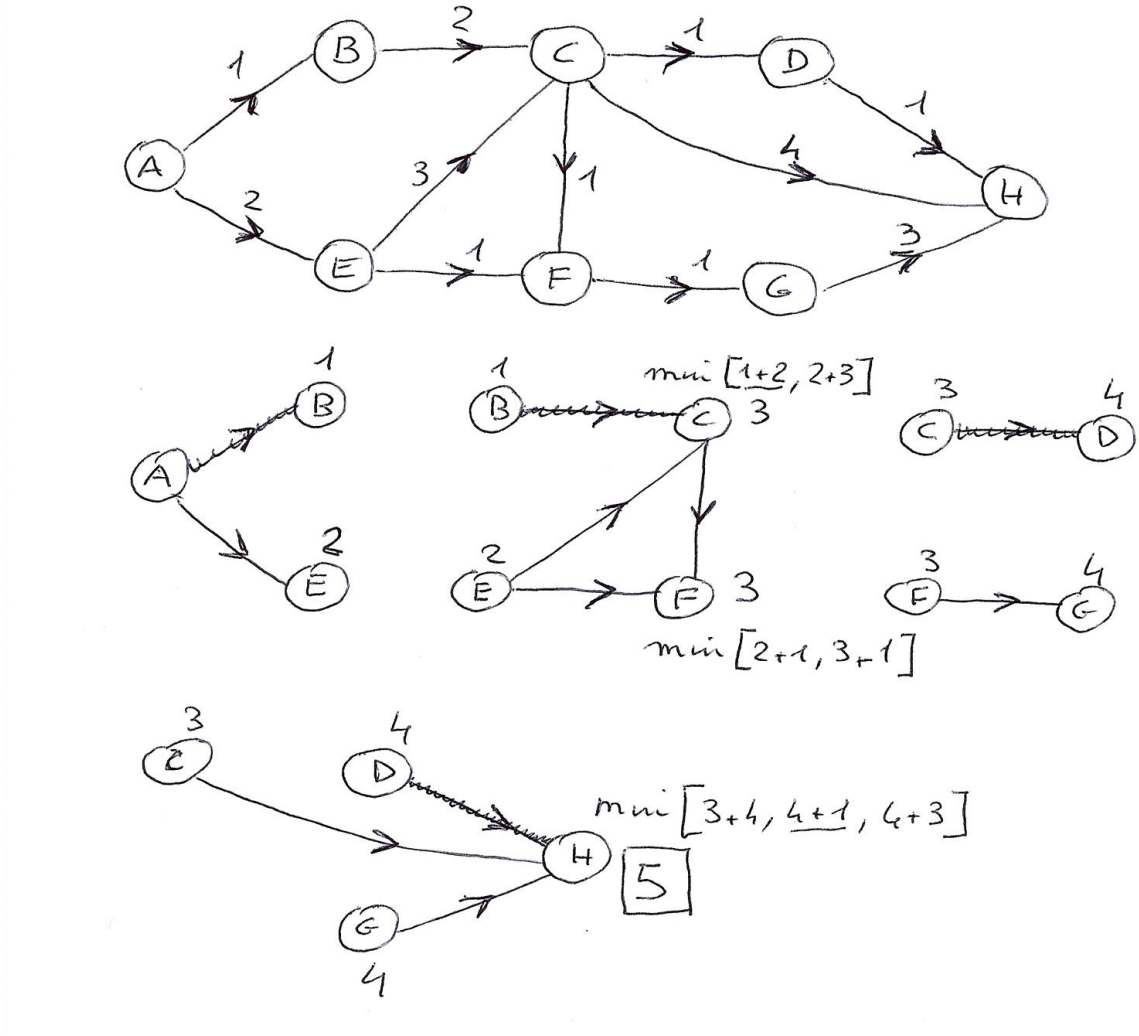


Assegnamento: qual è l'assegnamento più conveniente (direttori←→posti)

r_{ij} = "ricavo" 

$$\max \sum r_{ij}$$

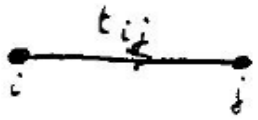
Cammini minimi: algoritmo delle etichette (PD)



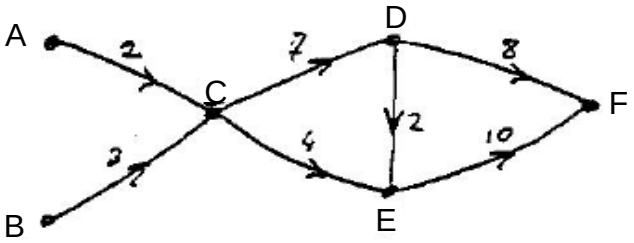
Il percorso ottimo è quindi (A,B,C,D,H)

- L'algoritmo funziona purché il grafo non contenga cicli (cioè sia aciclico) o meglio purché il grafo non contenga cicli a peso negativo.
- Se i nodi sono n si fanno al più n iterazioni. Ad ogni iterazione per ogni nodo si fanno al più n "confronti". Quindi il tempo di calcolo (complessità) cresce con n^3 (o(..)).

PERT (Project Evaluation and Review Technique)

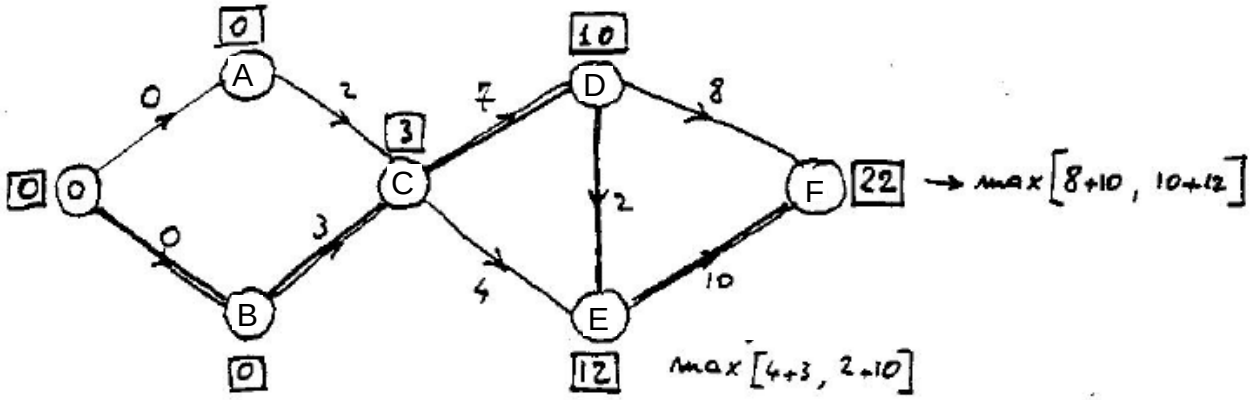


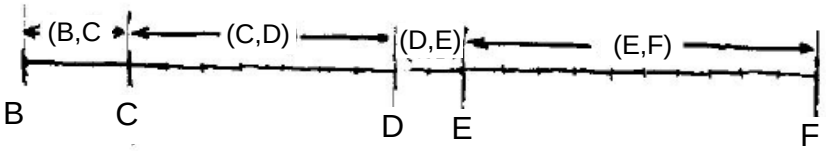
Attività (i.j) (o fase di un progetto)
t_{ij} = tempo di esecuzione



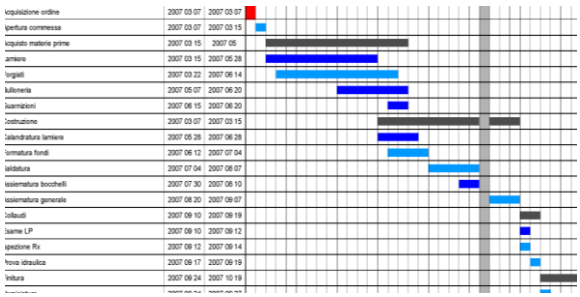
l'attività (C.D) può essere iniziata solo dopo che sono terminate le attività (A.C) e (B.C)

Si tratta di programmare I tempi di esecuzione delle varie fasi del progetto: per questo bisogna ovviamente determinare il cammino massimo nel seguente grafo (ottenuto dal primo aggiungendo il nodo 0):





Il tempo di esecuzione del progetto è pari (al minimo) a 22.

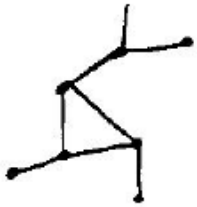


Il diagramma dei tempi è detto "diagramma di Gantt"

Le attività che non costituiscono il cammino critico possono essere eseguite "con comodo" (inizio ritardato o durata prolungata).

Albero minimo: algoritmo “greedy” (ingordo)

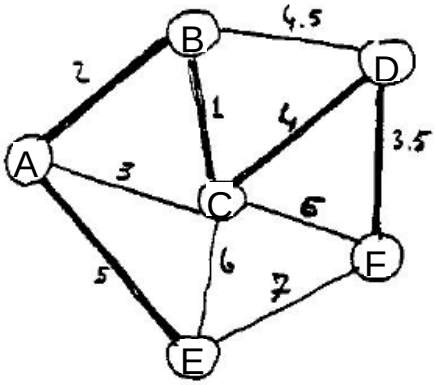
Grafo ad albero



Grafo contenente un ciclo

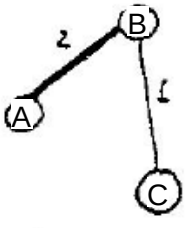
Un albero a n nodi ha n-1 archi

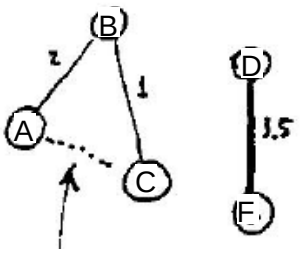
Dato un grafo non orientato con n nodi si cerca l’albero di n nodi con somma dei pesi associati minima.



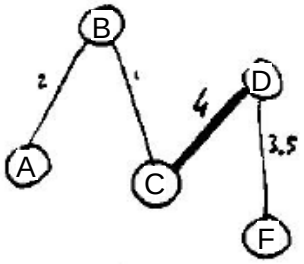
L’arco a peso minimo (B.C) deve far parte dell’albero ottimo (vedi fig. sopra).

L’algoritmo costruisce l’albero scegliendo dal grafo l’arco a peso minimo e iterando (naturalmente scartando quegli archi che chiuderebbero dei cicli).



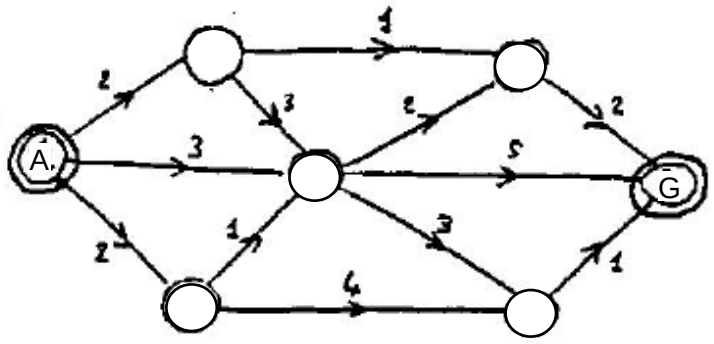


No perché chiuderebbe un ciclo



Al passo seguente si ottiene la soluzione

Massimo flusso: algoritmo di Ford-Fulkerson



Qual è il flusso massimo tra A e G?

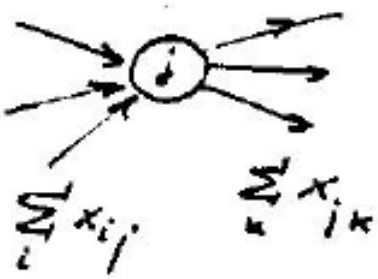
Ogni arco ha una capacità massima c_{ij}

Il problema può essere formulato come L.P.

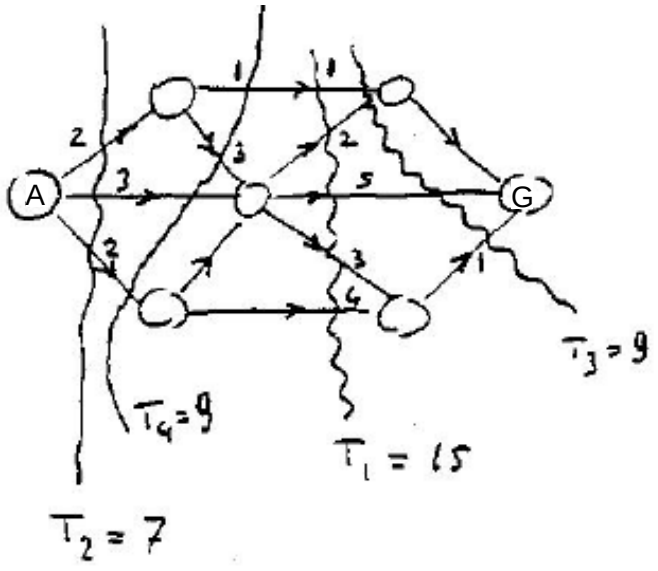
$$\max\left[\, flusso \,\right] = \max\left[\sum_j x_{1j} \right]$$

$$\sum_j x_{ij} = \sum_k x_{jk} \qquad j = 2,3,\ldots,n-1$$

$$x_{ij} \geq 0 \qquad x_{ij} \leq c_{ij}$$



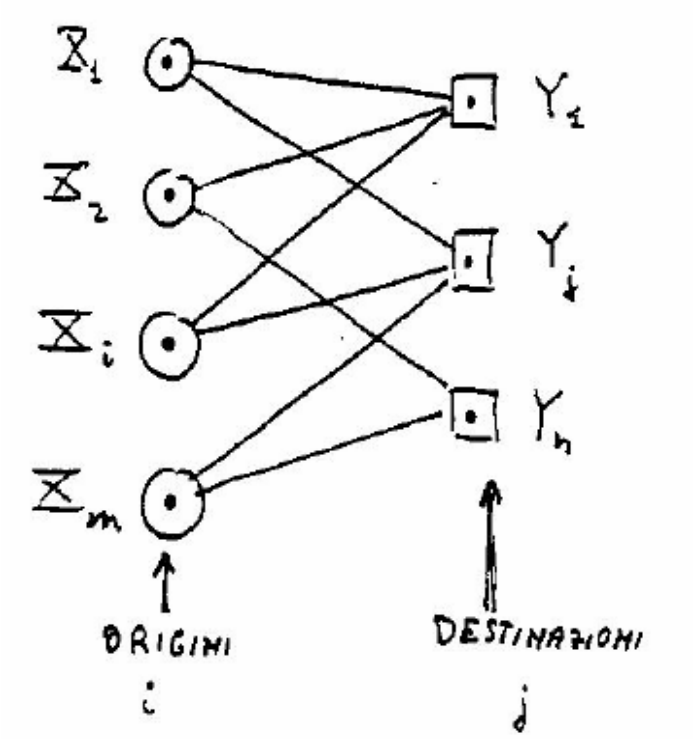
Esiste però un algoritmo ad hoc, dovuto a Ford-Fulkerson. Questo algoritmo è basato sull'osservazione che il massimo flusso uguaglia la somma delle capacità di una "sezione" (taglio) della rete (il cosiddetto collo di bottiglia).



Max [flusso] = min [taglio]

Nel caso di figura il minimo taglio è T_2 e quindi il flusso massimo da A a G è pari a 7.

Trasporto: algoritmo di PL



m origini n destinazioni

X_i = quantità prodotta in i

Y_i = quantità richiesta in j

Il problema può essere formulato come LP

$$\min\left[\sum_i\sum_j c_{ij}x_{ij}\right] \longleftarrow \min.\ \text{costo trasporto}$$

$$\sum_j x_{ij} \geq X_i \qquad i = 1,\ldots,m$$

$$\sum_i x_{ij} \leq Y_j \qquad j = 1,\ldots,n$$

$$x_{ij} \geq 0$$

- Anche per questo problema esiste un algoritmo ad hoc
- Il problema con n = m e $X_i = Y_j = 1$ è il problema dell'assegnamento