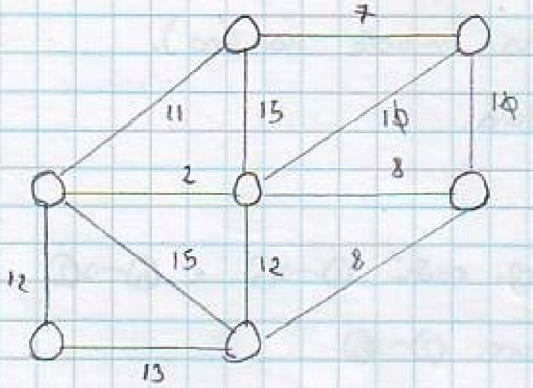
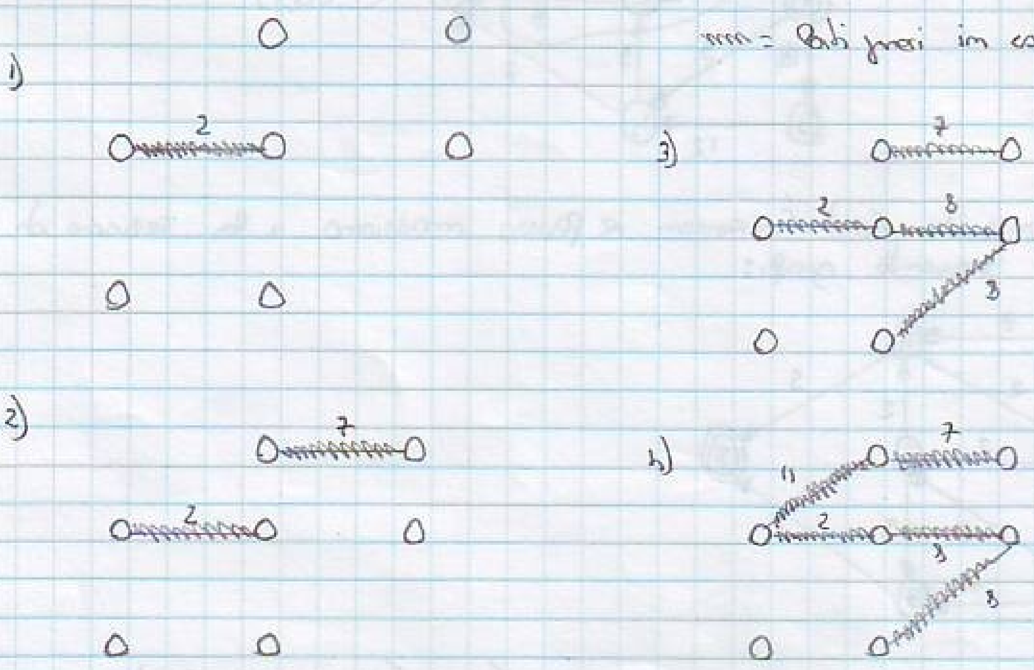
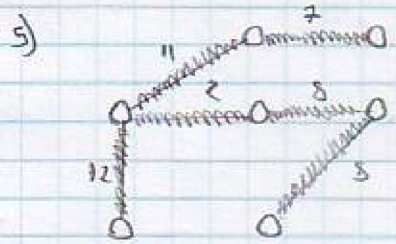


1) Innanzitutto consideriamo il lato che costa meno e via via consideriamo i lati che costano di più stando attenti a non formare cicli.

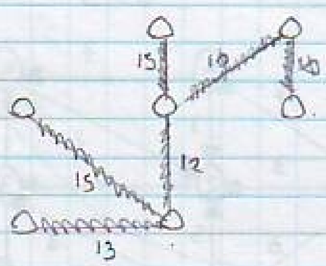
~~~ = lati presi in considerazione.



Attenzione: i lati 10 non vengono presi in considerazione perché insieme formano un ciclo.

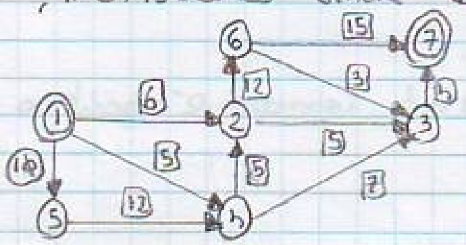


→ Albero ricoprente di costo minimo. L'albero ricoprente di costo massimo è il complementare di quello di costo minimo. Quindi:



ALBERO RICOPRENTE DI COSTO MASSIMO.

2) Individuare con l'algoritmo di FORD-FULKERSON il flusso massimo nel grafo seguente, individuando anche la sezione di capacità minima.


~~~

2) Possibili cammini di flusso (ipotizzando che il flusso iniziale sia zero).

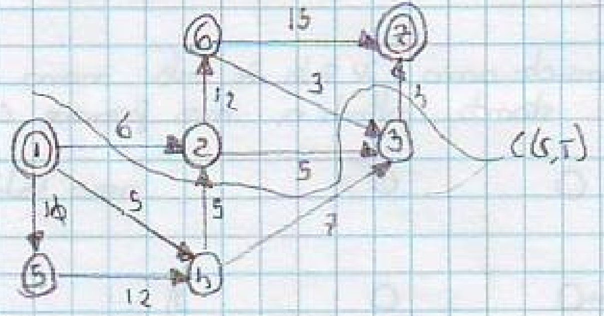
1) 1 $\rightarrow$ 2 $\rightarrow$ 3 $\rightarrow$ 7 $\delta = 4$ $\Rightarrow$ 2 $\rightarrow$ 3 diventa satura.

2) 1 $\rightarrow$ 2 $\rightarrow$ 6 $\rightarrow$ 7 $\delta = 2$ $\Rightarrow$ 1 $\rightarrow$ 2 si satura

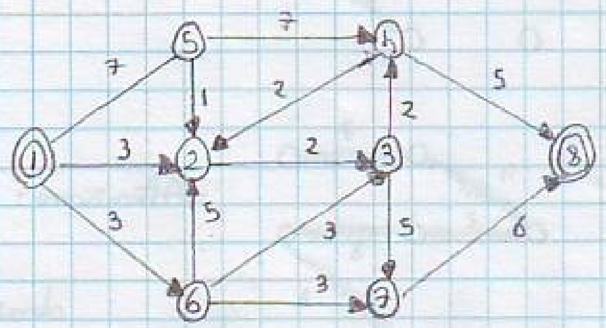
3) 1 $\rightarrow$ 4 $\rightarrow$ 2 $\rightarrow$ 6 $\rightarrow$ 7 $\delta = 5$ e si saturano gli archi 1 $\rightarrow$ 4 e 4 $\rightarrow$ 2

4) 1 $\rightarrow$ 5 $\rightarrow$ 4 $\rightarrow$ 3 $\rightarrow$ 2 $\rightarrow$ 6 $\rightarrow$ 7 $\delta = 4$ e si satura 2 $\rightarrow$ 6

Quindi $\delta_{TOT} = 4+2+5+4 = 15$ $\Rightarrow$



3) Individuare con l'algoritmo di Ford-Fulkerson il flusso massimo e la sezione di capacità minima del seguente grafo:



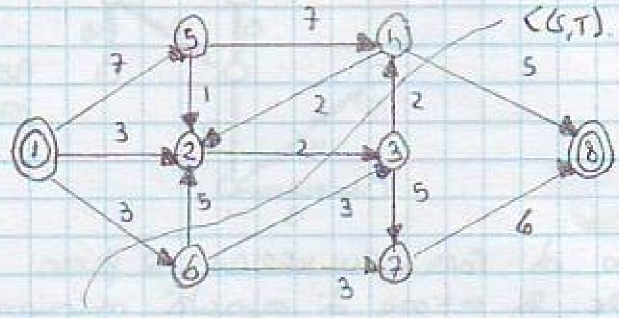
3) cammini possibili: $\delta = 0$ (FLUSSO INIZIALE)

1) 1 $\rightarrow$ 2 $\rightarrow$ 3 $\rightarrow$ 7 $\rightarrow$ 8 $\delta = 2$ e 2 $\rightarrow$ 3 saturo

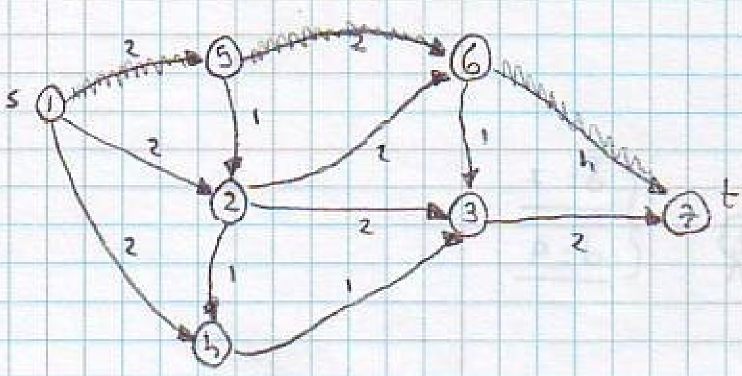
2) 1 $\rightarrow$ 6 $\rightarrow$ 7 $\rightarrow$ 8 $\delta = 3$ e 1 $\rightarrow$ 6 $\rightarrow$ 7 saturi.

3) 1 $\rightarrow$ 5 $\rightarrow$ 4 $\rightarrow$ 8 $\delta = 5$ e 4 $\rightarrow$ 8 saturo

Quindi: $\delta_{TOT} = 10$.

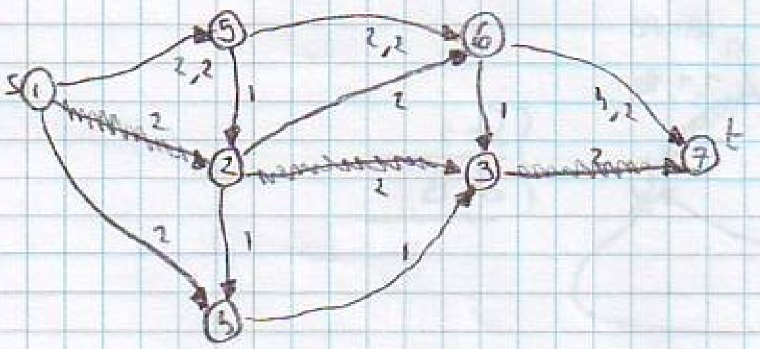


4) Risolvere il problema del massimo flusso da s a t usando l'algoritmo dei cammini aumentanti.

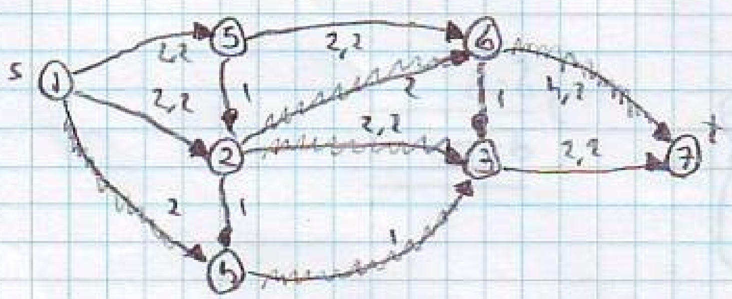


$X_{iniz} = \emptyset$

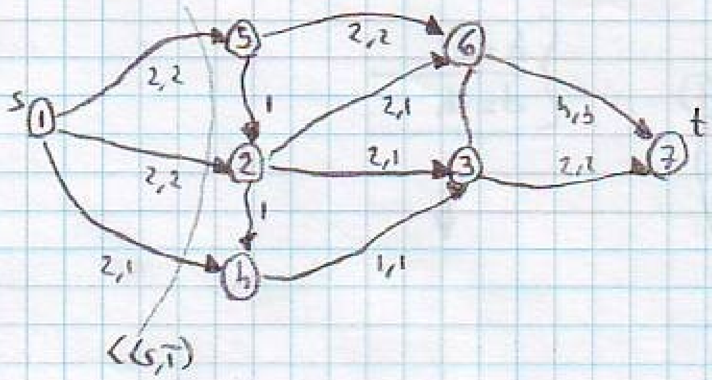
$\underline{\delta = 2}$



$\underline{\delta = 2}$

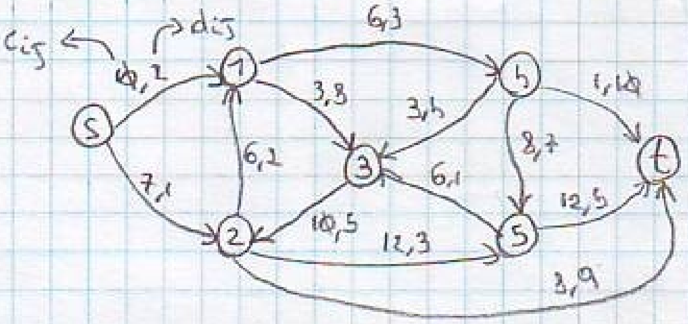


$\underline{\delta = 1}$



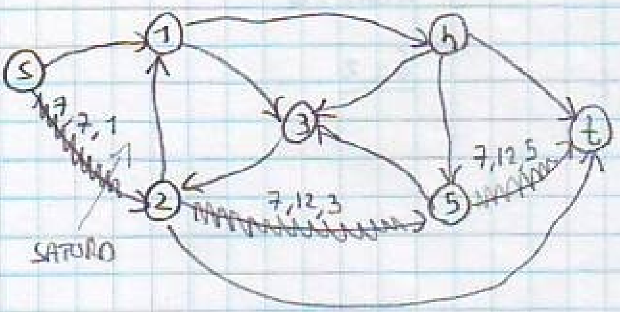
$X_{TOT} = \underline{2 + 2 + 1 = 5}$

5) Determinare flusso massimo di costo minimo nella seguente rete:

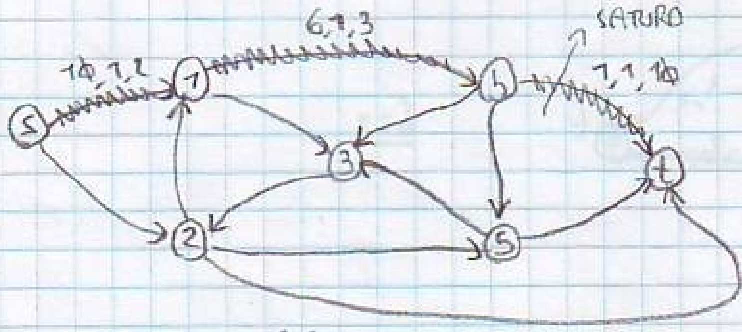


74

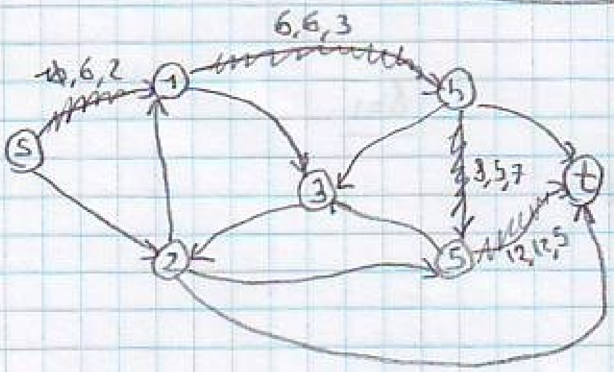
5)



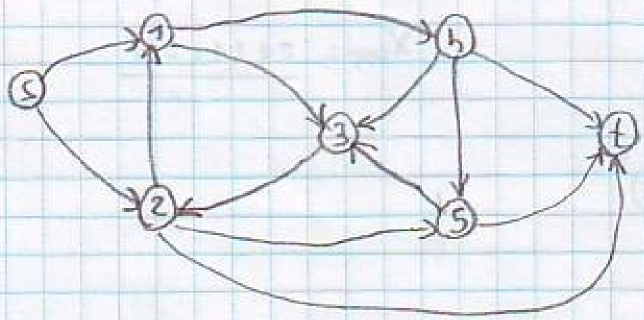
$$\begin{cases} \delta = 7 \\ d = 9 \end{cases}$$



$$\begin{cases} \delta = 1 \\ d = 15 \end{cases}$$

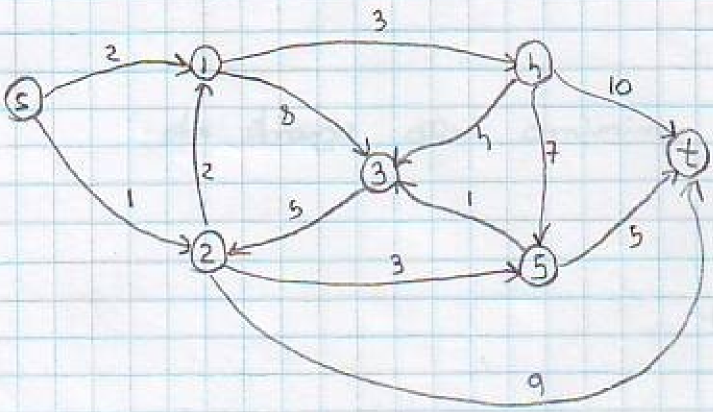


$$\begin{cases} \delta = 5 \\ d = 17 \end{cases}$$



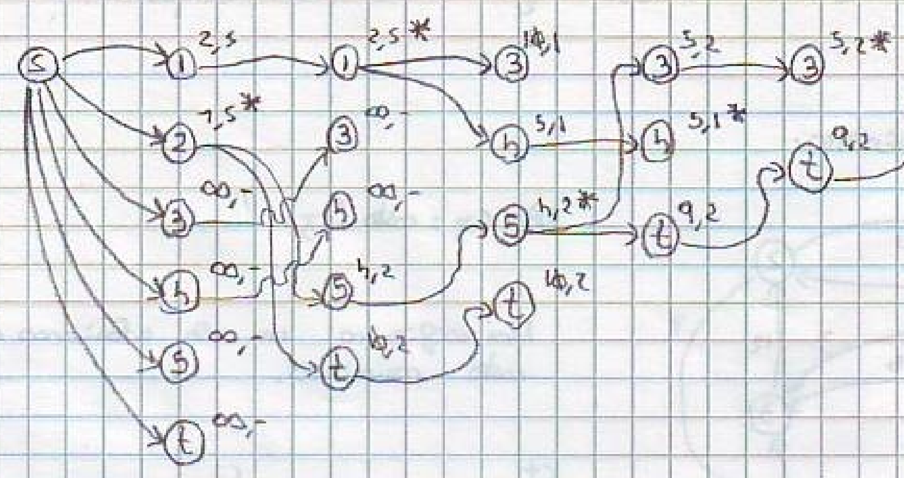
$$\begin{cases} \delta = 3 \\ d = 24 \end{cases}$$

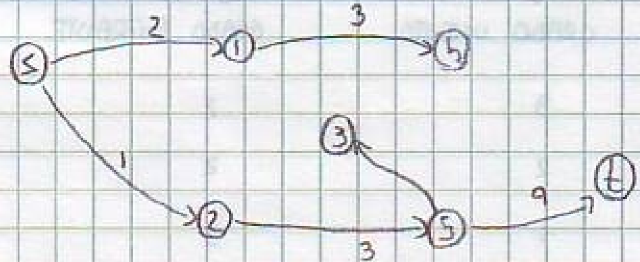
6)



Usando Dijkstra, trovare arborescenza minima.

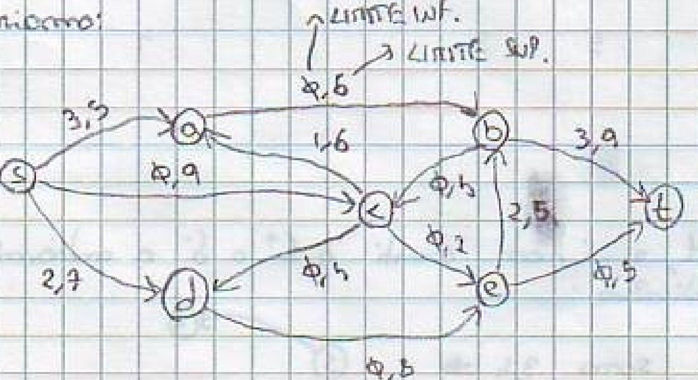
Dijkstra: (i) costo, origine





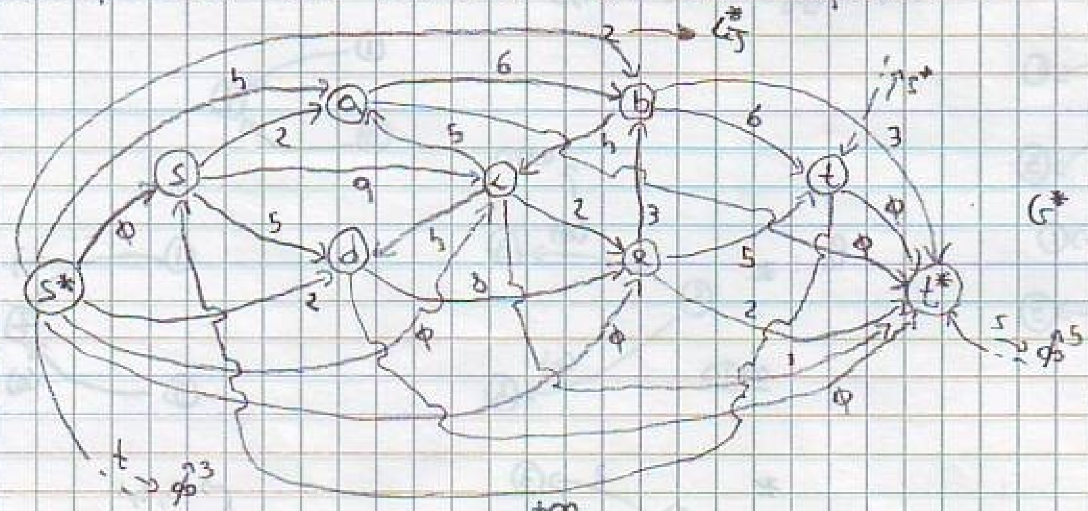
ARBORESCENZA DI COSTO MINIMO.

7) Consideriamo:



Trovare flusso di costo minimo con vincoli inferiori.

7)



$v^* = 11 = \sum e_j$

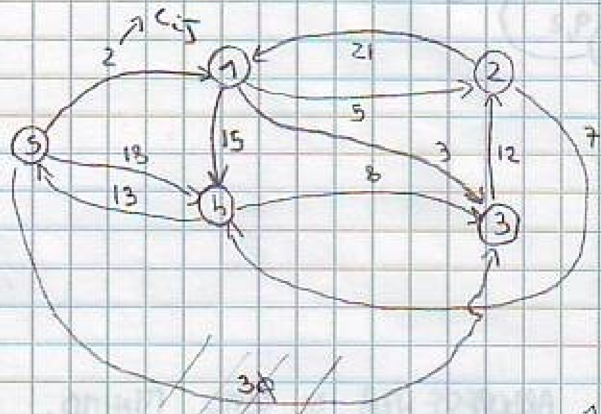
FLUSSI:

1) $s^*, b, t^* \rightarrow \underline{\delta = 2}$
2) $s^*, a, b, t^* \rightarrow \underline{\delta = 1}$
3) s^*, a, b, t, s, t^* $\underline{\delta = 3}$
4) s^*, d, e, t^* $\underline{\delta = 2}$
5) s^*, t, s, t^* $\underline{\delta = 2}$
6) s^*, t, s, c, t^* $\underline{\delta = 1}$

Vediamo ora se è minimo con il metodo degli archi non conformi.

...

3) Problema del POSTINO CINESE ORIENTATO:

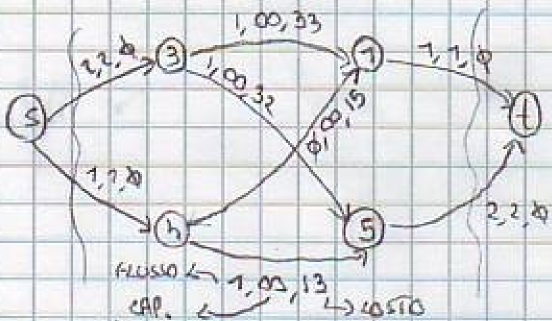


NB: c_{ij} = costo (i,j).

Noi vogliamo un ciclo euleriano di costo minimo.

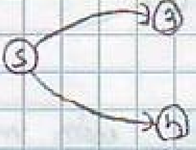
	δ^+ GRADO USCENTE	δ^- GRADO ENTRANTE
1	3	2
2	2	2
3	1	3
4	2	3
5	3	1

A questo punto scriviamo il grafo ausiliario:



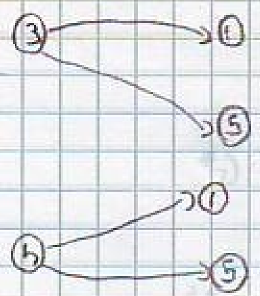
Praticamente consideriamo solo s,t, e i nodi aventi o δ^+ o δ^- o entrambi dispari. Inoltre si considera se $\delta^- \lessgtr \delta^+$. In particolare:

δ^+ maggiori sono 3,4 ⇒

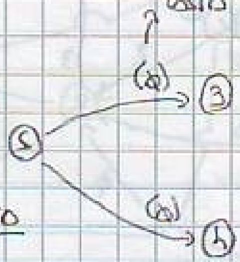


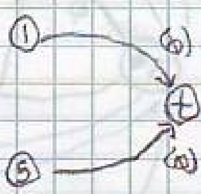
δ^+ maggiori sono 1,5

Poi

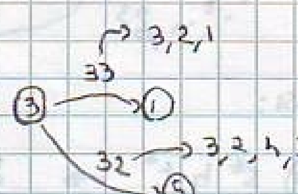


Poi: *





*



Inoltre: $c_{s,i} = \delta^- - \delta^+$

$c_{i,t} = \delta^+ - \delta^-$

* FLUSSO: $x_{ij} = c_{ij}$ per s, t. Per i nodi intermedi si segue il principio di solenoidalità.

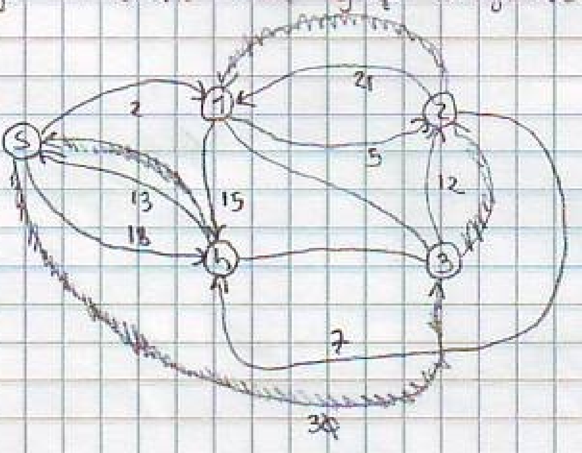
A questo punto si svolgono le stesse azioni relative al problema del postino cinese non orientato. In particolare posso usare Busacker & Gowen e quindi:

1) s, 4, 5, t $\delta = 1$, $c = 13$

2) s, 3, 5, t $\delta = 1$, $c = 32$ con $c = \sum \delta_c = 78$.

3) s, 3, 1, t $\delta = 1$, $c = 33$

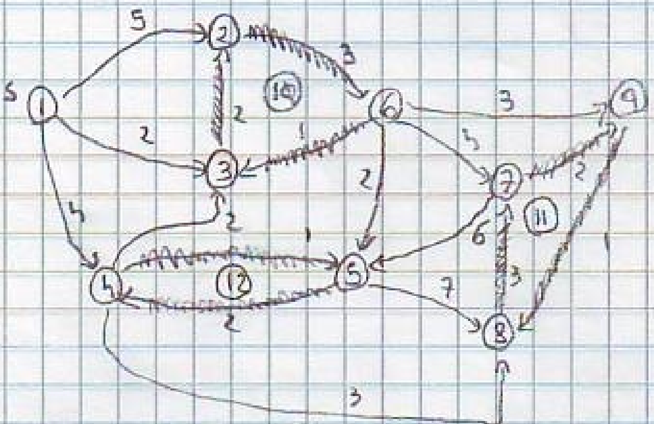
Quindi aggiungo i archi nel grafo originale. Perciò:



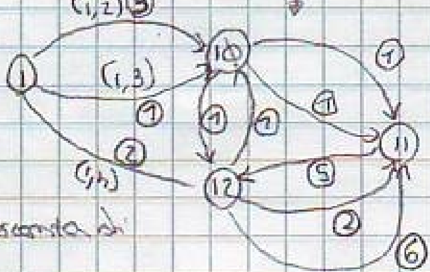
= ARCO AGGIUNTO

Su questo grafo si trova il ciclo euleriano

9) ARBORESCENZA A COSTO MINIMO con il metodo di Edmonds:



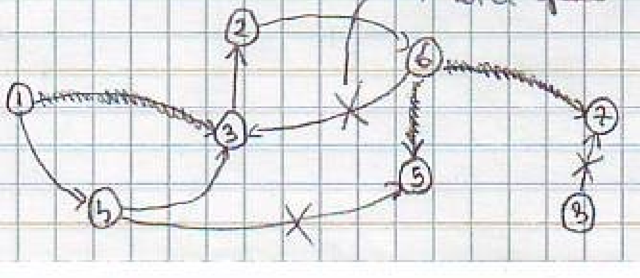
Considero l'arco minimo che entra in ogni nodo.
Formo in questo caso 3 cicli e ottengo un MULTIGRAFO in questo modo



NB: Ogni ciclo viene etichettato.

NB: PESO = PESO ORIGINALE − PESO ARCO CHE VA A SPASSO.

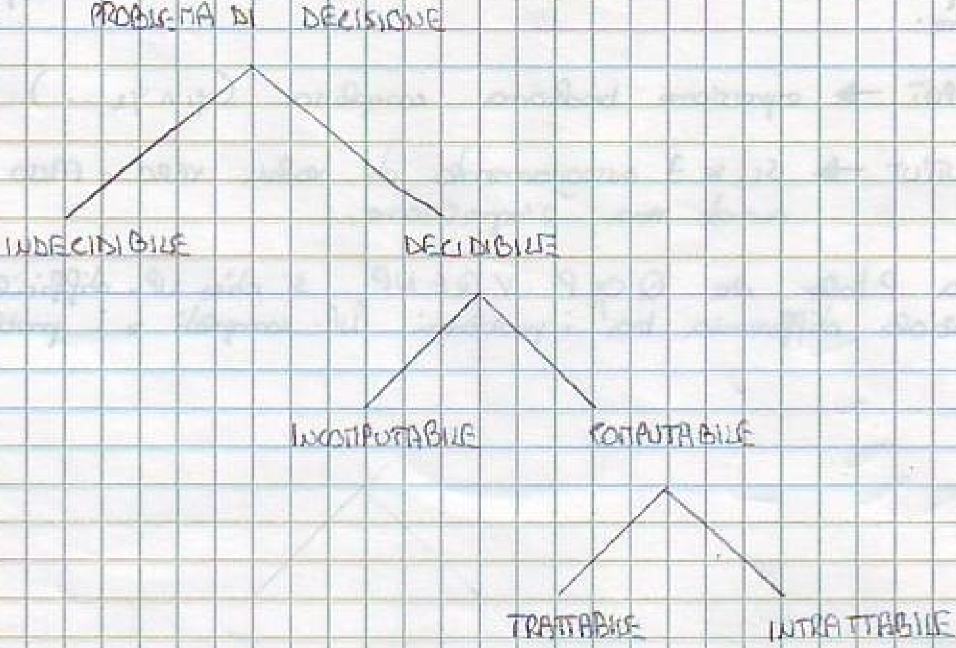
Alla fine quando non ho più cicli ho trovato l'arborescenza di costo minimo:



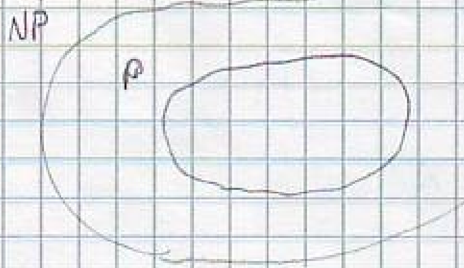
1, 3, 2, 6, 7, 5, 9, 8, 4.

PARTE SULLA CLASSIFICAZIONE DEI PROBLEMI.

Un problema di decisione può essere così classificato:



Si ricordi che un generico problema si dice trattabile se la soluzione può essere trovata usando algoritmi la cui complessità è polinomiale. Indichiamo con P la classe di problemi di decisione trattabili. Questi problemi possono essere risolti rispondendo semplicemente con un SI o con un NO. Rientrano nella classe NP quei problemi per i quali rispondere con un sì costa una complessità polinomiale, mentre rientrano nella classe co-NP quei problemi la cui soluzione rispondendo con un NO è un problema trattabile. Quindi co-NP è il complementare di NP. Graficamente:



Siano ora dati due problemi P_1 e P_2. Se esiste un algoritmo A tale che ogni istanza di P_2 ottiene un'istanza di P_1 in modo che risolvendo quest'ultima ci consente di risolvere anche l'istanza di P_2, allora P_2 è RIDUCIBILE a P_1. Si scrive:

$\underline{P \propto Q}$ P è riducibile a Q.

Chiaramente se si ha $P \propto Q$, e Q è difficile da risolvere non si guadagna niente, ma se Q è facile anche P lo è. I problemi più difficili della classe NP sono i problemi NP-completi. Un generico problema B è NP-completo quando:

1) $B \in NP$

2) $\forall L \in NP$ è riducibile a B in tempo polinomiale.

20

Stephen Cook dimostrò che ogni problema $\in NP$ è riducibile in tempo polinomiale in un problema di SODDISFACIBILITÀ (SAT). Un problema di questo tipo è un problema dove:

- INPUT → espressione booleana complessa $(y_1 \wedge y_2 \ldots)$
- OUTPUT → SI, se $\exists$ assegnamento di valori VERO o FALSO alle variabili che rende vera l'espressione.

Un problema P tale che $Q \propto P \ \forall Q \in NP$, si dice NP-difficile ($P \in NP$?). Questa è la sostanziale differenza tra i problemi NP-completi e i problemi NP-difficili.