Esame di Ricerca Operativa - 3 luglio 2014
Facoltà di Scienze MM.FF.NN. - Verona
- CORREZIONE -

Problema 1 (3+1+2+1=7 punti):

Riforniamo $n_C = 3$ clienti di acqua da $n_S = 2$ sorgenti. Le richieste dei 3 clienti, riportate in ettolitri nella seguente tabella, devono essere assolutamente evase:

Cliente 1	Cliente 2	Cliente 3
150	80	210

Le disponibilità (in ettolitri) dalle 2 sorgenti sono le seguenti:

Sorgente 1	Sorgente 2
250	300

Si noti che $250 + 300 = 550 > 440 = 150 + 80 + 210$, ed è quindi lecita l'aspettativa di evadere ogni richiesta. I costi unitari di trasporto, come da seguente tabella, sono a carico dei clienti.

	Cliente 1	Cliente 2	Cliente 3
Sorgente 1	10	15	20
Sorgente 2	8	14	7

Poiché non vogliamo disaffezionare nessun cliente, vogliamo organizzare il trasporto in modo che il massimo costo di trasporto unitario sostenuto da un singolo cliente nel suo complesso, risulti il piú basso possibile.

((**3pt**)) Fornire un modello di PL per tale problema specifico.

((**1pt**)) Fornire un modello di PL generale che si riferisca ad un numero n_C arbitrario di clienti ed ad un numero n_S arbitrario di sorgenti. Si continui ad assumere che la portata complessiva delle sorgenti sia sufficiente ad evadere tutte le richieste.

((**2pt**)) Esisterà sempre una soluzione ottima che è anche intera? Fornire argomentazione a supporto (dimostrazione) oppure controesempio.

((**1pt**)) Proporre un modello generale che valga nel caso in cui la disponibilità alle sorgenti non basti ad evadere tutta la richiesta. In questo caso, vorremmo collocare la massima quantità d'acqua possibile, e, soggetto a ciò, vorremmo poi minimizzare il massimo costo complessivo di trasporto sostenuto da un singolo cliente.

svolgimento.

((**3pt**)) Le primissime variabili di decisione che ci è naturale introdurre sono la quantità d'acqua da inviare da ciascuna sorgente a ciascun cliente, come espressamente definite nella seguente tabella.

	Cliente 1	Cliente 2	Cliente 3
Sorgente 1	$x_{1,1}$	$x_{1,2}$	$x_{1,3}$
Sorgente 2	$x_{2,1}$	$x_{2,2}$	$x_{2,3}$

Resta tuttavia da introdurre ancora qualche variabile di decisione. Esprimere la funzione obiettivo di questo problema come una funzione lineare, richiedendo esso di minimizzare un massimo, ci obbliga infatti ad introdurre una ulteriore variabile di decisione z per il valore dell'ottimo.

Il problema è quindi quello di minimizzare z imponendo che esso non sia inferiore nè al costo sostenuto dal Cliente 1, nè a quello sostenuto dal Cliente 2, nè a quello sostenuto dal Cliente 3. Pertanto,

$$\min z\,,$$

nel rispetto dei seguenti vincoli:

$$\begin{aligned} 150\,z &\geq 10\,x_{1,1} + 8\,x_{2,1}\,, \\ 80\,z &\geq 15\,x_{1,2} + 14\,x_{2,2}\,, \\ 210\,z &\geq 20\,x_{1,3} + 7\,x_{2,3}\,, \end{aligned}$$

Cui vanno poi ad affiancarsi i vincoli che a questo punto diseguono naturalmente dalla formulazione impostata.

non negativitá

$$x_{1,1}, x_{1,2}, x_{1,3}, x_{2,1}, x_{2,2}, x_{2,3} \geq 0.$$

portata delle sorgenti

$$\begin{aligned} x_{1,1} + x_{1,2} + x_{1,3} &\leq 250. \\ x_{2,1} + x_{2,2} + x_{2,3} &\leq 300. \end{aligned}$$

richieste dei clienti

$$\begin{aligned} x_{1,1} + x_{2,1} &= 150. \\ x_{1,2} + x_{2,2} &= 80. \\ x_{1,3} + x_{2,3} &= 210. \end{aligned}$$

(**(1pt)**) Con n_S sorgenti e n_C clienti, le variabili di decisione sono, oltre la z che stabilisce il valore all'ottimo, le $n_S n_C$ variabili $x_{i,j}$, $i = 1, \ldots, n_S$, $j = 1, \ldots, n_C$ definite da

$x_{i,j}$ = la quantità d'acqua da inviare direttamente dalla Sorgente i al Cliente j.

Dove si assuma di ricevere in input, o di altrimenti prestabilire, una matrice $n_S \times n_C$ di costi di trasporto C, con

$C_{i,j}$ = costo del trasportare un ettolitro d'acqua direttamente dalla Sorgente i al Cliente j,

il problema è quello di minimizzare il massimo costo totale di trasporto in cui vada ad incorrere un singolo cliente.

$$\min z\,,$$

nel rispetto dei seguenti vincoli:

$$C_j z \quad \geq \quad \sum_{i=1}^{n_S} C_{i,j} x_{i,j} \text{ per ogni } j = 1, \ldots, n_C,$$

dove C_j esprime il fabbisogno del Cliente j e compare in questo vincolo per esprimere il costo di trasporto unitario sostenuto dal Cliente j complessivamente. Avremo inoltre i seguenti vincoli.

vincoli di non negativitá

$$x_{i,j} \geq 0 \text{ per ogni } i = 1, \ldots, n_S,\ j = 1, \ldots, n_C.$$

portata delle sorgenti

$$\sum_{j=1}^{n_C} x_{i,j} \leq S_i \text{ per ogni } i = 1, \ldots, n_S,$$

dove S_i é la disponibilitá d'acqua alla Sorgente i.

richieste dei clienti

$$\sum_{i=1}^{n_S} x_{i,j} = C_j \text{ per ogni } j = 1, \ldots, n_C,$$

dove C_j esprime il fabbisogno del Cliente j.

(**(2pt)**) Ecco un controesempio con $n_S = n_C = 2$, $S_1 = S_2 = 1$, $C_1 = C_2 = 1$ e

	Cliente 1	Cliente 2
Sorgente 1	2	2
Sorgente 2	0	0

Dove l'ottimo si ha inviando $\frac{1}{2}$ ettolitro lungo ognuna delle 4 tratte per un costo di 1, mentre ciascuna delle (solo 2) soluzioni ottime intere vede uno dei due clienti rifornirsi interamente alla sorgente costosa ed incorrere nell'intero costo di 2.

(**(1pt)**) Puó forse risultare controintuitivo dapprima, ma vi è una simmetria perfetta tra le sorgenti ed i clienti:

> se l'acqua non basta a coprire le richieste, allora le richieste bastano ad accogliere l'intera offerta d'acqua.

Quindi, nel caso in cui non vi sia abbastanza acqua al soddisfare tutte le richieste adotteremo allora il seguente modello:

$$\min z\,,$$

nel rispetto dei seguenti vincoli:

$$z \;\geq\; \sum_{i=1}^{n_S} C_{i,j}x_{i,j} \text{ per ogni } j = 1,\ldots,n_C,$$

Cui vanno poi naturalmente ad affiancarsi i seguenti vincoli.

vincoli di non negativitá

$$x_{i,j} \geq 0 \text{ per ogni } i = 1,\ldots,n_S,\ j = 1,\ldots,n_C.$$

portata e pieno utilizzo delle sorgenti

$$\sum_{j=1}^{n_C} x_{i,j} = S_i \text{ per ogni } i = 1,\ldots,n_S,$$

dove S_i é la disponibilitá d'acqua alla Sorgente i.

richieste dei clienti

$$\sum_{i=1}^{n_S} x_{i,j} \leq C_j \text{ per ogni } j = 1,\ldots,n_C,$$

dove C_j esprime il fabbisogno del Cliente j.

Problema 2 (3+1+3=7 punti):
Ad ogni casella (i,j), $i,j = 0,\ldots,n-1$ di una scacchiera $n \times n$ é associato un costo $c_{i,j} \in \mathbf{N}$. Quando sulla casella (i,j) viene collocata una torre, si incorre nel costo $c_{i,j}$ ad essa associato. Vogliamo collocare delle torri sulla scacchiera in modo da minimizzare la somma dei costi ma sotto il vincolo che ogni cella risulti sotto il controllo di almeno una torre. Una torre controlla una cella se la occupa oppure se è collocata su una casella della stessa riga o colonna.

(**(3pt)**) Si formuli questo problema di ottimizzazione come un problema di programmazione lineare intera (PLI).

(**(1pt)**) Si consideri la versione di questo problema dove in input viene inoltre fornito un numero naturale k e viene richiesto di collocare esattamente k torri. Si mostri che la versione originale di questo problema non potrà certo rivelarsi piú difficile di quella con la prescrizione sul numero esatto di torri da impiegare.

(**(2+1pt)**) Si descriva un algoritmo che restituisca sempre una soluzione ammissibile ottima per la versione originale di questo problema. Dimostra che la soluzione restituita dal

tuo metodo è sempre ottima.

svolgimento.

(**(2pt)**) Le variabili di decisione sono le n^2 variabili booleane $x_{i,j}$, $i, j = 1, \ldots, n$, una per ogni casella, e definite da

$$x_{i,j} = \begin{cases} 1 & \text{se nella casella } (i,j) \text{ viene collocata una torre.} \\ 0 & \text{altrimenti.} \end{cases}$$

Il problema è quindi quello di minimizzare i costi.

$$\min \sum_{i,j} c_{i,j} x_{i,j} \,,$$

nel rispetto dei seguenti vincoli:

vincoli di interezza e dominio

$$x_{i,j} \in \{0,1\} \text{ per ogni } i, j = 1, \ldots, n.$$

vincoli di copertura di ogni cella

$$x_{i,j} + \sum_{\delta=1}^{n-1} x_{(i+\delta).mod.n,j} + \sum_{\delta=1}^{n-1} x_{i,(j+\delta).mod.n} \geq 1 \text{ per ogni } i, j = 1, \ldots, n.$$

(**(1pt)**) Data un'istanza che non preveda una prescrizione preassegnata sul numero delle torri da impiegare, essa potrà essere risolta con al più n chiamate ad un'ipotetica funzione capace di risolvere istanze con specifica sul numero k di torri da impiegarsi. Basterá provare (in gergo si dice *guessare*) tutti i possibili valori di k da 1 ad n^2. Questa proposta sopra viene chiamata una *Turing reduction* in quanto ci si avvale di un numero polinomiale di chiamate. In realtà in questo caso possiamo proporre una semplice *Karp reduction*, ossia ridurci ad una sola chiamata alla ipotetica funzione (chiamata *oracolo* in gergo). È infatti possibile dimostrare che, per ogni istanza del problema originale senza prescrizione sul numero delle torri da impiegarsi, esiste sempre una soluzione ottima che impiega esattamente n torri. È infatti facile convincersi che se una soluzione ottima non contiene una torre su ogni riga, ossia lascia vuota almeno una riga, allora deve prevedere almeno una torre su ciascuna colonna; a questo punto sarà possibile rinunciare ad una delle torri collocate su una quasiasi colonna che ne contiene almeno 2 (una tale colonna deve esistere se ho collocato più di n torri) senza perdere in ammissibilità e potendo solo migliorare in termini di funzione obiettivo. Interessante osservare come questi ragionamenti che hanno consentito di migliorare la riduzione secondo criteri oggettivi abbiano rappresentato un'utile occasione per migliorare la nostra comprensione del problema. Non è quindi un caso che ragionamenti analoghi a questi risultimo poi centrali nel progetto di un algoritmo risolutore quale quello descritto al punto seguente.

(**(1+1pt)**) L'algoritmo procede come segue: con riferimento ad una soluzione ottima *Opt*, non nota ma fissata, vi sono due sole ipotesi alternative:

o la soluzione *Opt* **colloca almeno una torre su ciascuna delle righe** nel qual caso si considera che porre una torre per riga, sulla cella di costo minimo della riga, produce sempre una soluzione ammissibile, e che il costo di una tale soluzione (ovviamente facile da produrre) non può certo eccedere quello di *Opt*;

o la soluzione *Opt* **colloca almeno una torre su ciascuna delle colonne** nel qual caso si considera che porre una torre per colonna, sulla cella di costo minimo della colonna, produce sempre una soluzione ammissibile, e che il costo di una tale soluzione (ovviamente facile da produrre) non può certo eccedere quello di *Opt*.

Non è possibile infatti che *Opt* non collochi alcuna torre nè su alcuna cella della riga r nè su alcuna cella della colonna c, altrimenti la cella (r, c) non sarebbe sotto il controllo di alcuna torre.

Un modo (=algoritmo) per produrre una soluzone ottima consiste pertanto di costruirsi una soluzione S_r che prenda la casella di costo minimo di ciascuna riga ed una soluzione S_c che prenda la casella di costo minimo di ciascuna colonna, confrontarne i costi e, delle due, prendere quella piú economica con scelta arbitraria in caso di pareggio.

Problema 3 (8 punti):

$$\begin{array}{l}\max\ 2 + 22x_1 - 10x_2 - 12x_3 \\ \left\{\begin{array}{rrrcr} 10x_1 - & x_2 + & 4x_3 & \leq & 8 \\ -10x_1 + & 5x_2 - & 2x_3 & \geq & 10 \\ x_1, x_3 \geq 0 & & & & \end{array}\right.\end{array}$$

3.1(1pt) Portare il problema in forma standard.

3.2(1pt) Impostare il problema ausiliario.

3.3(1pt) Risolvere il problema ausiliario.

3.4(1pt) Scrivere il tableau per una soluzione ammissibile di base al problema originario.

3.5(1pt) Risolvere il problema originario all'ottimo.

3.6(1pt) Quanto si sarebbe disposti a pagare per ogni unità di incremento per l'availability nei due vincoli? (Per piccole variazioni.)

3.7(1pt) Fornire una soluzione primale, parametrizzata negli incrementi, che evidenzi la nostra disponibilitá a pagare tale prezzo.

3.8(1pt) Fino a dove si sarebbe disposti a pagare tale prezzo?

svolgimento.

(3.1) Portando il problema in forma standard otteniamo:

$$\begin{array}{l}\max\ 2 + 22x_1 - 10x_2^+ + 10x_2^- - 12x_3 \\ \left\{\begin{array}{rrrrcr} 10x_1 - & x_2^+ + & x_2^- + & 4x_3 & \leq & 8 \\ 10x_1 - & 5x_2^+ + & 5x_2^- + & 2x_3 & \leq & -10 \\ x_1, x_2^+, x_2^-, x_3 \geq 0 & & & & & \end{array}\right.\end{array}$$

(3.2) Il problema ausiliario è sempre ammissibile ed è ottenuto introducendo una variabile "di colla" x_0. Del problema originario ci interessa solamente investigare l'ammissibilità, e quindi viene gettata a mare la funzione obiettivo originaria e ci si prefigge invece di minimizzare la quantità di colla necessaria all'ottenimento dell'ammissibilità.

$$\begin{array}{l}\max \ -x_0 \\ \left\{\begin{array}{rrrrrrrrr} 10x_1 - & x_2^+ + & x_2^- + & 4x_3 - & x_0 & \leq & 8 \\ 10x_1 - & 5x_2^+ + & 5x_2^- + & 2x_3 - & x_0 & \leq & -10 \\ x_0, x_1, x_2^+, x_2^-, x_3 \geq 0 & & & & & & \end{array}\right.\end{array}$$

Si ha che il problema originario era ammissibile se e solo se il problema ausiliario ammette una soluzione ammissibile con $x_0 = 0$.

(3.3) Introduciamo le variabili di slack come segue.

$$\begin{array}{l}\max \ -x_0 \\ \left\{\begin{array}{rcrrrrrrr} w_1 & = & 8 - & 10x_1 + & x_2^+ - & x_2^- - & 4x_3 + & x_0 \\ w_2 & = & -10 - & 10x_1 + & 5x_2^+ - & 5x_2^- - & 2x_3 + & x_0 \\ x_0, x_1, x_2^+, x_2^-, x_3, w_1, w_2 \geq 0 & & & & & & & \end{array}\right.\end{array}$$

Tecnicamente, anche il problema ausiliario non è ad origine ammissibile, ma riusciamo facilmente a procurarci una soluzione di base ammissibile in un singolo pivot: facciamo entrare x_0 in base settandone il valore a 10 (si guarda al vincolo con termine noto più negativo) e facciamo uscire di base la variabile di slack per quel vincolo.

$$\begin{array}{l}\max \ -10 - 10x_1 + 5x_2^+ - 5x_2^- - 2x_3 - w_2 \\ \left\{\begin{array}{rcrrrrrrr} w_1 & = & 18 & - & 4x_2^+ + & 4x_2^- - & 2x_3 + & w_2 \\ x_0 & = & 10 + & 10x_1 - & 5x_2^+ + & 5x_2^- + & 2x_3 + & w_2 \\ x_0, x_1, x_2^+, x_2^-, x_3, w_1, w_2 \geq 0 & & & & & & & \end{array}\right.\end{array}$$

La soluzione di base attuale non è ancora ottima: il coefficiente della x_2^+ nella funzione obiettivo vale $5 > 0$, quindi portiamo la x_2^+ in base. A farle posto è la x_0 che si annulla, quindi il problema originario era ammissibile (basta zero colla). Effettuiamo questo ultimo pivot per il problema ausiliario avendo cura di portare la x_0 fuori base non appena essa si annulla (in caso di dizionario degenere potrei anche decidere di portare fuori base un'altra variabile, ma non sarebbe una buona idea ...).

$$\begin{array}{l}\max \ -x_0 \\ \left\{\begin{array}{rcrrrrrrr} w_1 & = & 10 - & 8x_1 & - & \frac{18}{5}x_3 + & \frac{1}{5}w_2 + & \frac{4}{5}x_0 \\ x_2^+ & = & 2 + & 2x_1 + & x_2^- + & \frac{2}{5}x_3 + & \frac{1}{5}w_2 - & \frac{1}{5}x_0 \\ x_0, x_1, x_2^+, x_2^-, x_3, w_1, w_2 \geq 0 & & & & & & & \end{array}\right.\end{array}$$

(3.4) Ora che x_0 è fuori base ci basta rimuovere la colonna relativa alla x_0 per ottenere un primo dizionario con soluzione di base associata ammissibile per il problema originario. In tale dizionario, la scrittura per la funzione obiettivo è stata ottenuta partendo dalla funzione obiettivo originaria ed utilizzando le equazioni del dizionario per svendere fuori le variabili di base in termini delle variabili non di base.

$$\begin{array}{l}\max \ 2 + 22x_1 - 10x_2 - 12x_3 = -18 + 2x_1 - 16x_3 - 2w_2 \\ \left\{\begin{array}{rcrrrrrrr} w_1 & = & 10 - & 8x_1 & - & \frac{18}{5}x_3 + & \frac{1}{5}w_2 \\ x_2^+ & = & 2 + & 2x_1 + & x_2^- + & \frac{2}{5}x_3 + & \frac{1}{5}w_2 \\ x_1, x_2^+, x_2^-, x_3, w_1, w_2 \geq 0 & & & & & & \end{array}\right.\end{array}$$

(3.5) La soluzione di base associata a questo dizionario non è ancora ottima visto che il coefficiente della x_1 nella funzione obiettivo è positivo. Portano in base x_1 esce w_1 ed otteniamo il seguente dizionario.

$$\begin{array}{l}\max\ -\frac{31}{2}-\frac{1}{4}w_1-\frac{169}{10}x_3-\frac{39}{20}w_2\\ \left\{\begin{array}{rcrcrcr} x_1 & = & \frac{5}{4} & - & \frac{1}{8}w_1 & - & \frac{9}{20}x_3 + \frac{1}{40}w_2\\ x_2^+ & = & \frac{9}{2} & - & \frac{2}{8}w_1 & - & \frac{1}{2}x_3 + \frac{1}{4}w_2\end{array}\right.\\ x_1, x_2^+, x_2^-, x_3, w_1, w_2 \geq 0\end{array}$$

Si noti come la soluzione di base associata al dizionario ottenuto sia ora ottima (tutti i coefficienti della funzione obiettivo sono non-positivi) e quindi in questo caso non sono necessari ulteriori passi di pivot.

In termini delle variabili di decisione originarie la soluzione ottima è data da $x_1 = \frac{5}{4}$, $x_2 = \frac{9}{2}$, $x_3 = 0$ cui corrisponde un valore di $-\frac{31}{2}$ per la funzione obiettivo.

(3.6) Questa soluzione di base non risulta essere degenere poichè $\frac{5}{4} > 0$ e $\frac{9}{2} > 0$. Pertanto, per ogni unità di incremento del termine noto del primo vincolo saremmo disposti a pagare $\frac{1}{4}$ (almeno per piccoli incrementi) e per ogni unità di incremento del termine noto del secondo vincolo saremmo disposti a pagare $\frac{39}{20}$ (almeno per piccoli incrementi).

(3.7) Lo studio di cosa succede a seguito di variazioni nei termini noti dei vincoli porta a considerare la seguente generalizzazione del problema originale:

$$\begin{array}{l}\max\ 2+22x_1-10x_2-12x_3\\ \left\{\begin{array}{rcrcrcr} 10x_1 & - & x_2 & + & 2x_3 & \leq & 8+t_1\\ 10x_1 & - & 5x_2 & + & x_3 & \leq & -10+t_2\end{array}\right.\\ x_1, x_3 \geq 0\end{array}$$

Il tableau per la soluzione di base di questo problema caratterizzata dalla medesima partizione (in base/fuori base) delle variabili che nella soluzione ottima riscontrata per il problema originario differirà dal tableau di detta soluzione del problema originario solo per la colonna dei termini noti, la quale può essere facilmente ricostruita avvalendosi della prova del nove per il tableau. Poichè $(x_1, x_2^+, x_2^-, x_3, w_1, w_2, z) = (0, 0, 0, 0, 8+t_1, -10+t_2, 0)$ soddisfaceva al primissimo tableau (dizionario) essa dovrá soddisfare anche all'ultimo, e queste 3 condizioni ci consentono di ricostruire le 3 entries nella colonna dei termini noti. Con i conseguenti conteggi otteniamo il seguente tableau:

$$\begin{array}{l}\max\ -\frac{31}{2}+\frac{1}{4}t_1+\frac{39}{20}t_2-\frac{1}{4}w_1-\frac{169}{10}x_3-\frac{39}{20}w_2\\ \left\{\begin{array}{rcrcrcr} x_1 & = & \frac{5}{4}+\frac{1}{8}t_1-\frac{1}{40}t_2 & - & \frac{1}{8}w_1 & - & \frac{9}{20}x_3 + \frac{1}{40}w_2\\ x_2^+ & = & \frac{9}{2}+\frac{1}{4}t_1-\frac{1}{4}t_2 & - & \frac{1}{4}w_1 & - & \frac{1}{2}x_3 + \frac{1}{4}w_2\end{array}\right.\\ x_1, x_2, x_3, w_1, w_2 \geq 0\end{array}$$

Si noti come questo dizionario generalizzi effettivamente il dizionario da cui è stato ottenuto (riscontrabile per $t_1 = t_2 = 0$). La soluzione di base associata a questo dizionario, ossia $(x_1, x_2^+, x_2^-, x_3, w_1, w_2, z) = (\frac{5}{4}+\frac{1}{8}t_1-\frac{1}{40}t_2, \frac{9}{2}+\frac{1}{4}t_1-\frac{1}{4}t_2, 0, 0, 0, 0, -\frac{31}{2}+\frac{1}{4}t_1+\frac{39}{20}t_2)$ evidenzia la nostra disponibilità a pagare i prezzi ombra, come appaiono nell'espressione della coordinata z (valore di funzione obiettivo).

(3.8) Tale soluzione rimane indefinitivamente ammissibile al crescere di t_1, e pertanto non vi è limite alla nostra propensione a pagare quel prezzo sul primo vincolo. La non-negatività della x_2^+ suggerisce però che il prezzo ombra per la seconda risorsa perda il suo significato

per $t_2 > 18$. Quando $t_2 = 18$ la soluzione ottima è degenere e per $t_2 > 18$ dobbiamo rivedere le nostre strategie.

Problema 4 (6 punti):

Si consideri la seguente sequenza di numeri naturali (la prima riga serve solo ad indicizzarla).

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25
66	58	56	51	59	48	37	31	60	40	14	55	34	46	21	19	57	54	62	39	20	52	36	27	53

4.1(1pt) trovare una sottosequenza decrescente che sia la più lunga possibile. Specificare quanto è lunga e fornirla.

4.2(1pt) una sequenza è detta una Z-sequenza, o sequenza decrescente con un possibile ripensamento, se esiste un indice i tale cha ciascuno degli elementi della sequenza esclusi al più il primo e l'i-esimo sono strettamente minori dell'elemento che immediatamente li precede nella sequenza. Trovare la più lunga Z-sequenza che sia una sottosequenza della sequenza data. Specificare quanto è lunga e fornirla.

4.3(1pt) trovare la più lunga sottosequenza decrescente che includa l'elemento di valore 60. Specificare quanto è lunga e fornirla.

4.4(1pt) trovare una sottosequenza decrescente che sia la più lunga possibile ma eviti di utilizzare i primi 4 elementi. Specificare quanto è lunga e fornirla.

4.5(1pt) trovare una sottosequenza decrescente che sia la più lunga possibile ma eviti di utilizzare gli elementi dal 13-esimo a 16-esimo. Specificare quanto è lunga e fornirla.

4.6(1pt) fornire un minimo numero di sottosequenze non-decrescenti tali che ogni elemento della sequenza originale in input ricada in almeno una di esse. Specificare quante sono e fornirle.

svolgimento. Dapprima compilo la seguente tabella di programmazione dinamica.

DECRESCENTE

⇒	⇒	⇒	⇒	⇒	⇒	⇒	⇒	⇒	⇒	⇒	⇒	⇒	⇒	⇒	⇒	⇒	⇒	⇒	⇒	⇒	⇒	⇒	⇒	⇒
9	8	7	6	6	5	4	3	6	4	1	5	3	4	2	1	5	4	4	3	1	3	2	1	1
66	58	56	51	59	48	37	31	60	40	14	55	34	46	21	19	57	54	62	39	20	52	36	27	53
1	2	3	4	2	5	6	7	2	6	8	4	7	6	8	9	3	5	2	7	9	6	8	9	8
⇐	⇐	⇐	⇐	⇐	⇐	⇐	⇐	⇐	⇐	⇐	⇐	⇐	⇐	⇐	⇐	⇐	⇐	⇐	⇐	⇐	⇐	⇐	⇐	⇐

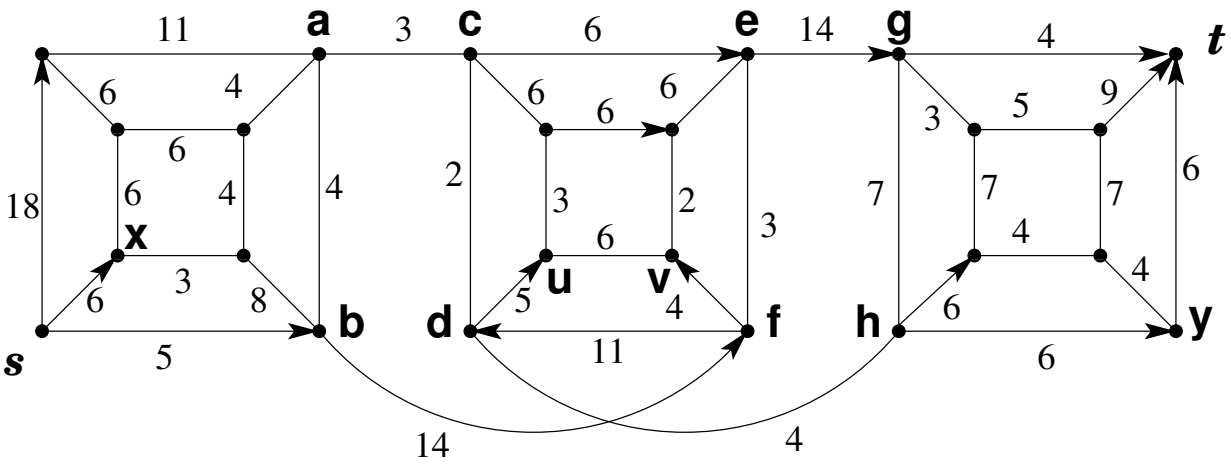
DECRESCENTE

Possiamo ora fornire le seguenti risposte.

tipo sottosequenza	opt val	soluzione ottima
decrescente	9	66, 58, 56, 51, 48, 37, 31, 21, 19
Z-sequenza	14	66, 58, 56, 51, 48, 37, 31, 21, 19, 57, 54, 39, 36, 27
decrescente con 60	7	66, 60, 55, 46, 39, 36, 27
evita i primi 4	6	59, 48, 37, 31, 21, 19
evita da 13-mo a 16-mo	9	66, 58, 56, 51, 48, 40, 39, 36, 27
minima copertura	9	$\underbrace{66}_{1}; \underbrace{58, 59, 60, 62}_{2}; \underbrace{56, 57}_{3}; \underbrace{51, 55}_{4}; \underbrace{48, 54}_{5}; \underbrace{37, 40, 46, 52}_{6}; \underbrace{31, 34, 39}_{7}; \underbrace{14, 21, 36, 53}_{8}; \underbrace{19, 20, 27}_{9}$

Dove per il penultimo punto (4.5) si é osservato dalla tabella di DP (ultima riga) che:
per raccogliere 8 elementi sul solo lato sinistro, l'ultimo deve valere 14 al piú ,
per raccogliere 7 elementi sul solo lato sinistro, l'ultimo deve valere al piú 31,
per raccogliere 6 elementi sul solo lato sinistro, l'ultimo deve valere al piú 30,
per raccogliere 5 elementi sul solo lato sinistro, l'ultimo deve valere al piú 48,
per raccogliere 4 elementi sul solo lato sinistro, l'ultimo deve valere al piú 55,
per raccogliere 3 elementi sul solo lato sinistro, l'ultimo deve valere al piú 56,
per raccogliere 2 elementi sul solo lato sinistro, l'ultimo deve valere al piú 60,
per raccogliere 1 elemento sul solo lato sinistro, esso deve valere al piú 66,
e si sono poi ordinatamente combinate queste osservazioni con analoghe osservazioni concernenti le migliori (non-dominate) scelte relative al come giocarsi il lato destro, sempre come lette dalla tabella (prima riga).
Infine, per l'ultimo punto (4.6) ho costruito la sequenza non-decrescente i-esima collocando in essa tutti quei numeri della sequenza in input tali che la massima lunghezza di una sequenza decrescente terminante in essi, come calcolata nell'ultima riga della tabella di PD, era precisamente i.

Problema 5 (12 punti):
Si consideri il grafo G, con pesi sugli archi, riportato in figura.



5.1.**(1pt)** Dire, certificandolo, se il grafo è planare oppure no.

5.2.**(1pt)** Trovare un albero ricoprente di peso minimo.

5.3.**(1pt)** Trovare tutti gli alberi ricoprenti di peso minimo. (Dire quanti sono e specificare con precisione come generarli).

5.4.**(1pt)** Per ciascuno dei seguenti archi dire, certificandolo, se esso appartenga a (tutte / a nessuna / a qualcuna ma non a tutte) le soluzioni ottime: sx, uv, yt.

5.5.**(1pt)** Trovare un albero dei cammini minimi da s e determinare le distanze di tutti i nodi da s.

5.6.**(1pt)** Trovare tutti gli alberi dei cammini minimi da s. (Dire quanti sono e specificare con precisione come generarli).

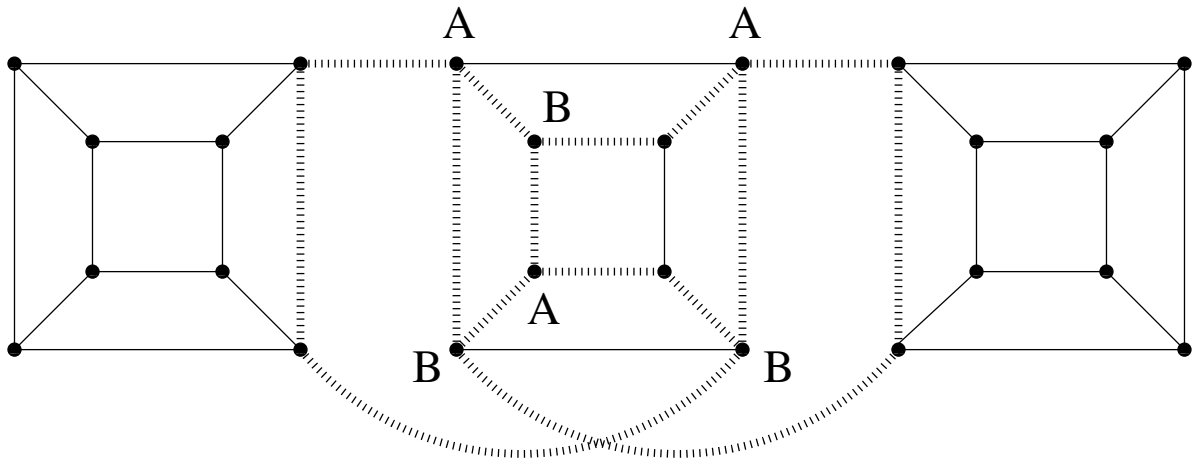
5.7.**(2pt)** Trovare un massimo flusso dal nodo s al nodo t.

5.8.**(2pt)** Certificare l'ottimalità del flusso massimo dal nodo s al nodo t.

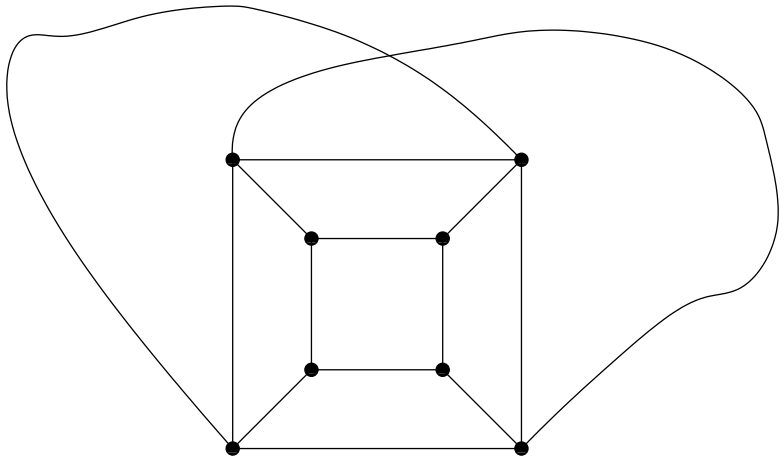
5.9.**(2pt)** Dire quale sia il minimo numero di archi la cui rimozione renda il grafo bipartito fornendo sia certificato (1pt) del fatto che il grafo ottenuto a seguito della rimozione è bipartito sia certificato (1pt) del fatto che la rimozione di un numero minore di archi non poteva bastare.

risposte.

Il fatto che G non sia planare può essere messo in evidenza esibendo la suddivisione di $K_{3,3}$ in figura.

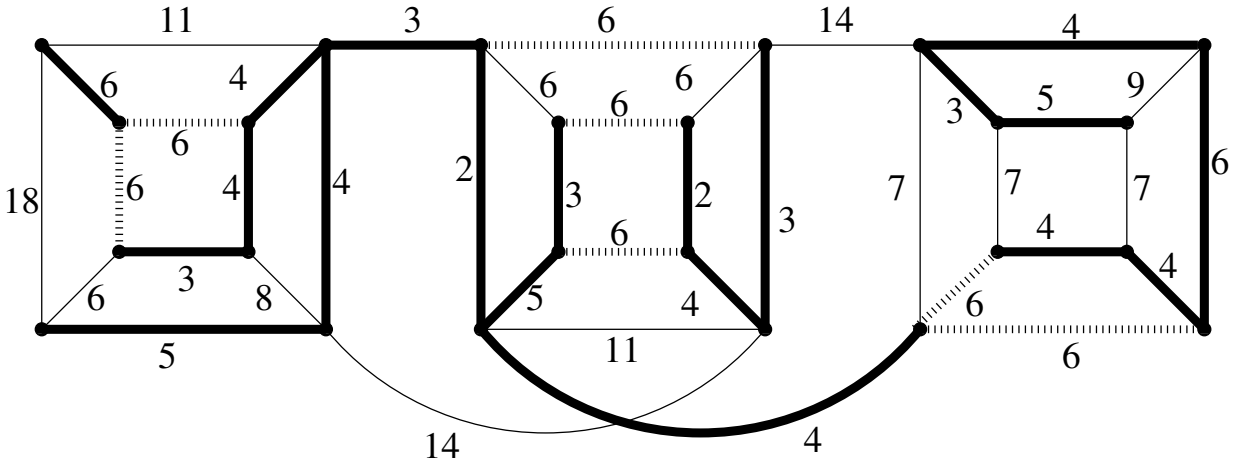


Nella ricerca di tale certificato (o di un planar embedding), poteva sicuramente aiutare la considerazione che G è planare se e solo se lo è anche il seguente grafo G'.

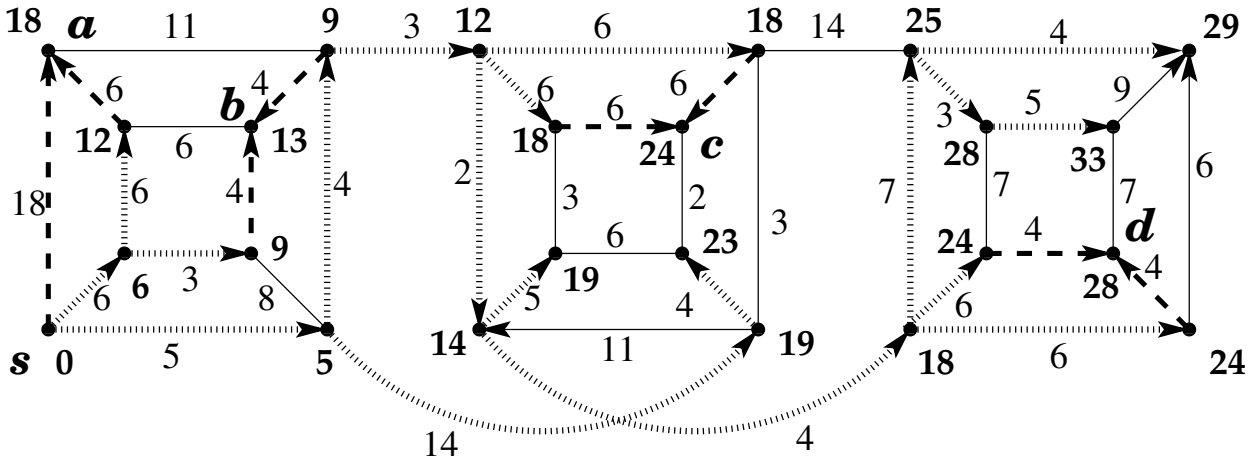


Tale grafo non sembra planare, e tuttavia non può contenere una suddivisione di K_5 visto che ha solo 4 nodi di grado almeno 4. Si era quindi indotti alla ricerca di una suddivisione di $K_{3,3}$. Una volta trovata una suddivisione di $K_{3,3}$ in G' era facile derivarne una suddivisione di $K_{3,3}$ in G.

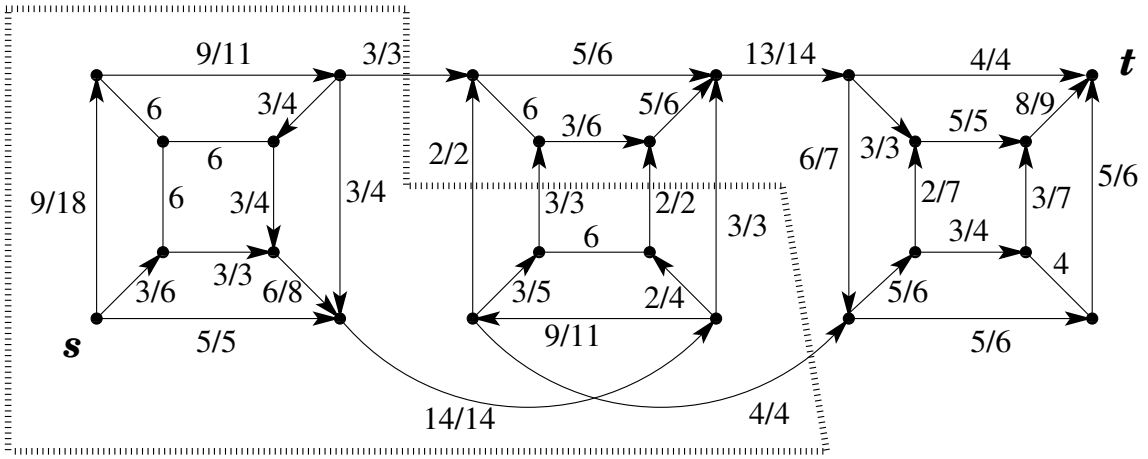
La seguente figura esprime la famiglia degli alberi ricoprenti di peso minimo. Ci sono 12 alberi ricoprenti di perso minimo e ciascuno di essi include i 20 archi in linea spessa, più uno qualsiasi dei 2 archi di peso 6 in linea sfumata spessa presenti nella zona a sinistra, più uno qualsiasi dei 3 archi di peso 6 in linea sfumata spessa presenti nella zona centrale, più uno qualsiasi dei 2 archi di peso 6 in linea sfumata spessa presenti nella zona a destra.



La seguente figura esprime la famiglia degli alberi dei cammini minimi dal nodo s. Ci sono $2^4 = 16$ alberi dei cammini minimi dal nodo s e ciascuno di essi include i 19 archi in linea spessa, più uno qualsiasi dei 2 archi tratteggiati entranti nel nodo a, uno qualsiasi dei 2 archi tratteggiati entranti nel nodo b, uno qualsiasi dei 2 archi tratteggiati entranti nel nodo c e uno qualsiasi dei 2 archi tratteggiati entranti nel nodo d.

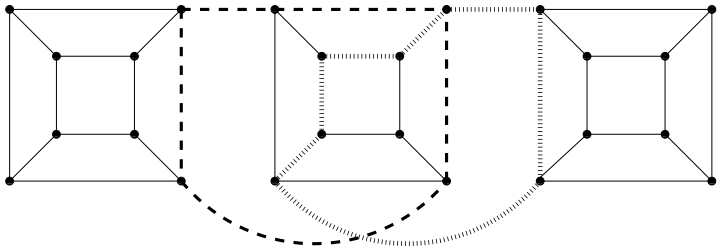


La seguente figura esibisce un flusso massimo (non esibisco tutti i passaggi che ho dovuto compiere per ottenerlo) ed un taglio (minimo) che ne dimostra l'ottimalità.

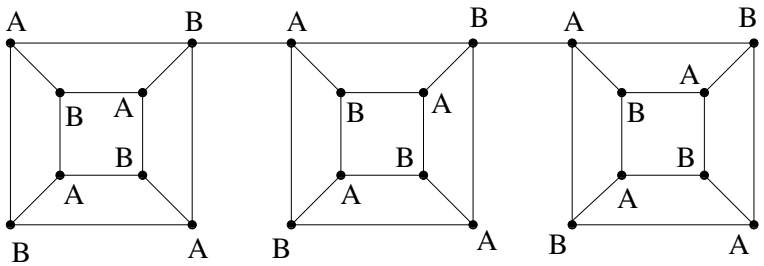


Il flusso ha valore 17 e satura l'insieme degli archi che attraversano la curva tratteggiata portandosi dal lato di s al lato di t. Questi 6 archi costituiscono pertanto un minimo s,t-taglio, anch'esso di valore 17 e che certifica pertanto l'ottimalità del flusso proposto.

Occorre rimuovere almeno 2 archi per rendere G bipartito dato che esso contiene 2 circuiti dispari disgiunti sugli archi come evidenziato in figura.



Di converso, con la rimozione di 2 soli archi possiamo rendere G bipartito come evidenziato in figura.



Problema 6 (6 punti):

Si ricerchino soluzioni algoritmiche per il seguente modello della Ricerca Operativa.

KNAPSACK' variante del KNAPSACK classico con vincolo sul massimo numero di oggetti presi.

INPUT: Tre numeri naturali n, k, B ed un insieme di n oggetti descritti ciascuno da una coppia valore/peso, (v_i, p_i) per ogni $i = 1, \ldots, n$.

OUTPUT: Trovare un sottoinsieme di al massimo k degli oggetti assegnati in input, a somma dei pesi non eccedente il budget assegnato B, e massimizzando il valore totale raccolto.

((**1pt**)) Si osservi come sia possibile ridurre il KNAPSACK classico alla versione KNAPSACK' di attuale interesse.

((**1pt**)) Se ne deduca che KNAPSACK' è NP-hard in senso debole.

((**1pt**)) Definire una famiglia di (al più un numero pseudo-polinomiale di) sottoproblemi chiusa rispetto ad induzione ed atta a risolvere KNAPSACK'.

((**1pt**)) Fornire una ricorrenza risolutiva per i sottoproblemi della famiglia proposta.

((**1pt**)) Trattare i casi base.

((**1pt**)) Specificare come vada letto dalla tabella il valore della soluzione ottima e come essa possa poi essere ricostruita.

svolgimento.

((**1pt**)) Si osservi come sia possibile ridurre il KNAPSACK classico alla versione KNAPSACK' di attuale interesse: il vincolo sul numero massimo di oggetti può sempre essere disabilitato semplicemente ponendo $k = n$.

((**1pt**)) Sappiamo che ogni problema in NP pu'øessere ridotto al KNAPSACK classico (che è noto essere NP-hard in senso debole) ed abbiamo visto che il KNAPSACK classico può essere ridotto alla sua variante KNAPSACK' da noi considerata. Componendo le due riduzioni se ne deduce che ogni problema in NP pu'øessere ridotto a KNAPSACK' e quindi KNAPSACK' è NP-hard in senso debole.

((**1pt**)) per $n' = 0, 1, \ldots, n$, $B' = 0, 1, \ldots, B$, e $k' = 0, 1, \ldots, k$, si definisca $opt[n', B', k']$ il valore della soluzione ottima per l'istanza modificata considerando solo i primi n' oggetti e ponendo $B := B'$ e $k := k'$.

((**3pt**)) Definire la famiglia corrisponde ad infilare la chiave di volta. Collocata questa pietra angolare nel punto sopra, riesci ora ad aggiudicarti gli altri tre punti?