# Esame di Ricerca Operativa - 24 giugno 2024

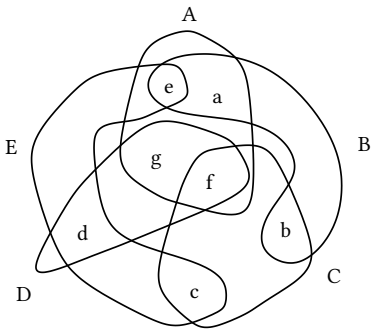
Facoltà di Scienze MM.FF.NN. - Verona

4 esercizi per 79 punti in palio (voto $\geq$ punti $-6$, $40 \longrightarrow 30$ *e lode*)

# - CORREZIONE -

**Esercizio 1 (con 12 richieste: 3+1+3+2+1+4+2+3+1+5+1+5 = 31 punti [modellazione/riduzioni]):**

Un ipergrafo è una coppia $H = (U, \mathcal{E})$ con $U$ un insieme finito di elementi chiamati *nodi* ed $\mathcal{E}$ una famiglia di sottoinsiemi non-vuoti di $U$ chiamati *iperarchi*. In pratica $H$ è una qualsiasi famiglia di insiemi non-vuoti, e viene chiamato anche col nome di *set system*. Un *set cover* di $H$ è un qualsiasi sottoinsieme $\mathcal{E}'$ di $\mathcal{E}$ tale che $\cup_{e \in \mathcal{E}'} e = U$.




**esempio.** i set covers dell'ipergrafo $\overline{H}$ in figura sono $\{A, B, E\}$, $\{A, C, D\}$, $\{A, C, E\}$, $\{B, D, E\}$ e tutti i loro sovrainsiemi. Non è invece set cover di $\overline{H}$ alcun sottoinsieme di $\{A, B, C\}$ (il nodo $d$ rimarrebbe scoperto), di $\{A, B, D\}$ (manca $c$), di $\{A, D, E\}$ (manca $b$), di $\{B, C, D\}$ (manca $e$), di $\{B, C, E\}$ (manca $g$), o di $\{C, D, E\}$ (manca $a$).

Min Set Cover è il problema di trovare un set cover di minima cardinalità per un generico ipergrafo $H$ dato in input.

**Richieste dell'Esercizio 1**

**1 .1** ( 3 pt, model via hypergraphs ) Un caporeparto ha preso atto che ci sono coppie di dipendenti che quando presenti entrambi passano tutto il giorno alla macchinetta del caffè a parlare male di tè o indugiano in altri comportamenti negativi non più tollerabili. Formula in termini di Min Set Cover il problema di chi licenziare assumendo di voler tenere più personale possibile visto che si è investito così tanto su di loro nei corsi di formazione per insegnargli a lavorare di più a meno paga.

**1 .2** ( 1 pt, forge hypergraph model ) Ad ogni dipendente hai associato un valore che esprime quanto ti costerebbe licenziarlo. Prova a definire un problema Min Set Cover Pesato che ti consenta di meglio rappresentare il problema di individuare chi licenziare volendo minimizzare i costi.

**1 .3** ( 3 pt, model as ILP ) Formula come un problema di Programmazione Lineare Intera (PLI) il problema Min Set Cover per la specifica istanza $\overline{H}$ in figura.

**1 .4** ( 2 pt, generalize ILP model ) Estendi la tua formulazione PLI a un generico ipergrafo $H = (U, \mathcal{E})$.

**1 .5** ( 1 pt, model as ILP ) Offri una formulazione PLI di Min Set Cover Pesato per istanza generica.

**1 .6** ( 4 pt, relax and yin yang ) Il rilassamento continuo di un problema di PLI è il problema di Programmazione Lineare (PL) che si ottiene facendo cadere i vincoli di interezza. Per la specifica istanza $\overline{H}$ in figura, scrivi il rilassamento continuo (1pt) della formulazione di PLI che hai dato, e il suo problema duale (2pt). Da quale teorema (1pt) puoi dedurre che se le soluzioni ottime di un problema di PLI restano ottime anche per il rilassamento continuo allora puoi certificarne l'ottimalità esibendo una soluzione ammissibile di pari valore nel duale del problema rilassato?

**1 .7** ( 2 pt, general dual ) Fornisci il duale del rilassamento continuo per ipergrafo $H$ generico.

**1 .8** ( 3 pt, fractional point ) Fornire un ipergrafo $H = (U, \mathcal{E})$ per il quale il rilassamento continuo non ha soluzioni ottime che siano intere. Si ottengono almeno 2pt se si fornisce una soluzione frazionaria ottima e si argomenta che ogni soluzione intera costerebbe di più, 3pt se $|U|$ è la più piccola possibile.

**1 .9** ( 1 pt, model via graphs ) Questa richiesta di un solo punto è un espediente che potrebbe facilitarti nell'intuire una riduzione da Min Node Cover a Max Independent Set, che è l'oggetto della più ambiziosa richiesta successiva. Un *node cover* di un grafo $G = (V, E)$ è un sottoinsieme di $V$ che contenga almeno uno dei due estremi per ciascun arco in $E$. Min Node Cover è il problema di trovare un node cover di minima cardinalità per un generico grafo $G$ dato in input. Domanda: Come potresti esprimere il tuo dilemma di quale personale licenziare e quale invece mantenere in termini di Min Node Cover?
**1 .10** ( 5 pt, NP-hardness proof 1 ) Min Node Cover è noto essere NP-hard. Sfrutta questo fatto per dimostrare che anche Min Set Cover è NP-hard.
**1 .11** ( 1 pt, NP-hardness proof 2 ) Deduci dal risultato sopra che anche Min Set Cover Pesato è NP-hard.
**1 .12** ( 5 pt, NP-hardness proof 3 ) SAT è noto essere NP-hard. Sfrutta questo fatto per dimostrare che anche Min Set Cover è NP-hard.

**Svolgimento esercizio 1 .**
**Richiesta 1.** Si consideri l'ipergrafo che ha per nodi le coppie di dipendenti incompatibili (quelle dove almeno uno dei due deve essere necessariamente licenziato) e dove per ogni dipendete $d$ si abbia un iperarco costituito precisamente da quelle coppie di dipendenti incompatibili che includano il dipendente $d$. Si noti che un sottoinsieme di questi iperarchi è un set cover in questo ipergrafo se e solo se essi corrispondono a dipendenti il cui licenziamento risolverebbe ogni problema in azienda. Pertanto, il problema di nostro interesse può essere visto come la ricerca di un set cover di cardinalità minima entro un ipergrafo dato.

**Richiesta 2.** Si noti che lo spazio delle scelte (ossia la specifica di quali dipendenti licenziare) non è cambiato per il fatto che ora guardiamo al costo economico diretto di ciascun licenziamento, l'impatto di questi in fondo concerne esclusivamente la formulazione della funzione obiettivo. Definiamo pertanto il problema Min Set Cover Pesato come la variante di Min Set Cover dove in input, oltre a $H = (U, \mathcal{E})$, mi viene inoltre specificata una funzione $c : \mathcal{E} \longrightarrow \mathbb{N}$, dove, per ogni iperarco $i \in \mathcal{E}$, il valore $c_i := c(v)$ riporta il costo di includere l'iperarco $i$ nella soluzione (ossia quanto mi costi licanziare il dipendente rappresentato da quell'iperarco). Il problema Min Set Cover Pesato chiede di trovare un set cover nell'ipergrafo in input che minimizzi la somma dei costi come presa sugli iperarchi inclusi. Chiameremo "ipergrafo pesato" una tale coppia $(H, c)$ che costituisce l'input del problema di ottimizzazione; volendo essere più specifici in questo caso potremmo parlare di "ipergrafo pesato sugli archi".

**Richiesta 3.** Introduciamo una variabile $x_i \in \{0, 1\}$ per ogni nodo iperarco $i \in \{A, B, C, D, E\}$, con l'idea che $x_i = 1$ significhi che l'iperarco $i$ vada incluso nel set cover mentre $x_i = 0$ significa che l'iperarco $i$ vada lasciato fuori. Queste 5 variabili binarie riescono quindi a catturare lo spazio delle possibili scelte, descrivendo compiutamente un qualsiasi sottoinsieme di iperarchi.

Volendo minimizzare la cardinalità del set cover selezionato, la funzione obbiettivo sarà:

$$\min x_A + x_B + x_C + x_D + x_E$$

Abbiamo un'unica famiglia di vincoli, che prevede precisamente un vincolo per ogni nodo:

**nodo *a*:** $x_A + x_B \geq 1$
**nodo *b*:** $x_B + x_C \geq 1$
**nodo *c*:** $x_C + x_E \geq 1$
**nodo *d*:** $x_D + x_E \geq 1$
**nodo *e*:** $x_A + x_B + x_E \geq 1$

**nodo $f$:** $x_A + x_C + x_D \geq 1$
**nodo $g$:** $x_A + x_D \geq 1$

Questi vincoli impongono che il sottoinsieme di iperarchi descritto dal vetore $x$ formi effettivamente un set cover.

**Richiesta 4.** Più in generale, introduciamo una variabile $x_i \in \{0, 1\}$ per ogni iperarco $i$ del generico ipergrafo $H = (U, \mathcal{E})$ che potremmo ricevere in input. Come sopra, $x_i = 1$ significa "iperarco $i$ incluso nel set cover soluzione" mentre $x_i = 0$ significa "iperarco $i$ NON incluso nel set cover soluzione".

La funzione obbiettivo sarà:

$$\min \sum_{i \in U} x_i$$

E la famiglia di vincoli è:

$$\sum_{i \ni u} x_i \geq 1 \quad \text{per ogni} \quad u \in U$$

Da tenere inoltre presenti i vincoli di non-negatività.

**Richiesta 5.**

Usiamo le stesse variabili $x_i \in \{0, 1\}$ di cui sopra (una per ogni iperarco $i$ del generico ipergrafo pesato $(H = (U, \mathcal{E}), c)$ che potremmo ricevere in input; e come sopra $x_i = 1$ significa che l'iperarco $i$ vada incluso).

La funzione obbiettivo sarà:

$$\min \sum_{i \in \mathcal{E}} c_i x_i$$

E la famiglia di vincoli rimane:

$$\sum_{i \ni u} x_i \geq 1 \quad \text{per ogni} \quad u \in U$$

**Richiesta 6.**

Il rilassamento continuo sarà:

$$\begin{aligned}
\min\ & x_A + x_B + x_C + x_D + x_E & & \\
& x_A + x_B & \geq 1 \quad & (\text{nodo } a) \\
& x_B + x_C & \geq 1 \quad & (\text{nodo } b) \\
& x_C + x_E & \geq 1 \quad & (\text{nodo } c) \\
& x_D + x_E & \geq 1 \quad & (\text{nodo } d) \\
& x_A + x_B + x_E & \geq 1 \quad & (\text{nodo } e) \\
& x_A + x_C + x_D & \geq 1 \quad & (\text{nodo } f) \\
& x_A + x_D & \geq 1 \quad & (\text{nodo } g) \\
& x_A, x_B, x_C, x_D, x_E \geq 0 & &
\end{aligned}$$

Il duale del rilassamento continuo sarà:

$$
\begin{aligned}
\max\ & y_a + y_b + y_c + y_d + y_e + y_f + y_g & & \\
& y_a & + y_e + y_f + y_g \leq 1 & \quad \text{(iperarco } A\text{)} \\
& y_a + y_b & + y_e \leq 1 & \quad \text{(iperarco } B\text{)} \\
& y_b + y_c & + y_f \leq 1 & \quad \text{(iperarco } C\text{)} \\
& y_d & + y_f + y_g \leq 1 & \quad \text{(iperarco } D\text{)} \\
& y_c + y_d + y_e & \leq 1 & \quad \text{(iperarco } E\text{)} \\
& y_a, y_b, y_c, y_d, y_e, y_f, y_g \geq 0 & &
\end{aligned}
$$

Quando una soluzione ammissibile del primale ed una soluzione ammissibile del duale presentano lo stesso valore nelle rispettive funzioni obiettivo, allora esse sono entrambe ottime come corollario del teorema della dualità debole. E ogni soluzione ottima del rilassamento continuo che rispetti anche i vincoli di interezza è ovviamente soluzione ottima anche per il problema non rilassato.

**Richiesta 7.**

Riferito ad un'istanza $H = (U, \mathcal{E})$ generica, il rilassamento continuo sarà:

$$
\begin{aligned}
\min\ & \sum_{i \in \mathcal{E}} x_i & & \\
& \sum_{i \ni u} x_i \geq 1 & \quad \text{per ogni } \ u \in U \\
& x_i \geq 0 \text{ per ogni } i \in \mathcal{E} & &
\end{aligned}
$$

L'interpretazione combinatorica generale del duale del rilassamento continuo sarà:

$$
\begin{aligned}
\max\ & \sum_{u \in U} y_u & & \\
& \sum_{u \in i} y_u \leq 1 & \quad \text{per ogni } \ i \in \mathcal{E} \\
& y_u \geq 0 \text{ per ogni } u \in U & &
\end{aligned}
$$

**Richiesta 8.** Sia $U = \{a, b, c\}$ ed $\mathcal{E} = \{\{a, b\}, \{b, c\}, \{a, c\}\}$. In pratica $H = (U, \mathcal{E})$ è il grafo chiamato "triangolo", e denotato $C_3$ (ciclo di 3 nodi) o $K_3$ (cricca di 3 nodi). Si noti (= può essere controllato da King Arthur, non serve farlo noi) che $x_{\{a,b\}} = \frac{1}{2}$, $x_{\{b,c\}} = \frac{1}{2}$, $x_{\{a,c\}} = \frac{1}{2}$ è soluzione ammissibile del rilassamento continuo, e ha costo $\frac{3}{2}$. (Non richiesto dall'esercizio, ma se voglio convincermi dell'ottimalità di tale soluzione posso considerare una soluzione di pari costo per il problema duale ($y_a = \frac{1}{2}, y_b = \frac{1}{2}, y_c = \frac{1}{2}$) e limitarmi a verificarne l'ammissibilità.) La soluzione intera ottima ha valore almeno 2 in quanto al massimo una delle tre variabili può essere settata a 0. Ad esempio se, senza perdita di generalità (per gli automorfismi del triangolo), assumo che $x_{\{a,b\}} = 0$, allora $x_{\{b,c\}} \geq 1$ disegue dal vincolo $x_{\{a,b\}} + x_{\{b,c\}} \geq 1$, e $x_{\{a,c\}} \geq 1$ disegue dal vincolo $x_{\{a,b\}} + x_{\{a,c\}} \geq 1$.

**Richiesta 9.** Si consideri il grafo che ha per nodi i dipendenti e dove due nodi sono adiacenti se tenerli entrambi darebbe problemi. Si noti che un sottoinsieme di dipendenti è un node cover in questo grafo se e solo se il loro licenziamento risolverebbe ogni problema in azienda. Pertanto, il problema di nostro interesse può essere visto come la ricerca di un node cover di cardinalità minima entro un grafo dato.

**Richiesta 10.** Data un'istanza $G = (V, E)$ di Min Node Cover consideriamo, ad istanza di Min Set Cover, l'ipergrafo $H = (E, \mathcal{E})$ che ha per nodi gli archi di $G$ (le coppie di dipendenti incompatibili) e dove per ogni dipendete $vinV$ si abbia un iperarco in $\mathcal{E}$ costituito precisamente da quegli archi in $E$ che includano il dipendente $v$. Si noti che un sottoinsieme di questi iperarchi è un set cover in questo

ipergrafo $H$ se e solo se essi corrispondono a nodi di $G$ che formano un node cover di $G$ (a dipendenti il cui licenziamento risolverebbe ogni problema in azienda).

La validità della riduzione poggia sulla seguente osservazione:

**Fact:** Un insieme $S \subseteq V$ è un node cover di $G$ se e solo se gli corrisponde un insieme di iperarchi di $H$ che costituiscono un set cover di $H$. Pertanto, in $G$ vi è un node cover $X$ di cardinalità $k$ se e solo se vi è un set cover di cardinalità $k$ in $H$.

Ne conseque l'NP-hardness del problema Min Set Cover.

**Richiesta 11.**

Questa riduzione è in discesa in quanto Min Set Cover può essere visto come il caso particolare di Min Set Cover Pesato ristretto a quelle sole istanze dove $c_i$ è sempre la funzione che associa il valore $c_i = 1$ ad ogni $i \in \mathcal{E}$.

**Richiesta 12.** Dobbiamo mostrare come, data una qualsiasi istanza di SAT, ossia una qualsiasi formula booleana $f$ in forma CNF, sia possibile costruire un'istanza di Min Set Cover in forma decisionale che la rappresenti. Infatti Min Set Cover è un problema di ottimizzazione mentre SAT è solo un problema di decisione, pertanto se vogliamo evitare di progettare una Turing reduction e limitarci a progettare una Karp reduction conviene concentraci su una versione di decisione di Min Set Cover che certo non potrà essere computazionalmente più difficile da risolvere che non la forma di ottimizzazione. Nella forma di decisione riceviamo in input non solo $H$ ma anche un budget di spesa $B$ e la domanda è se $H$ ammetta un set cover $\mathcal{E}'$ con $|\mathcal{E}'| \leq B$. Siano $x_1, ..., x_n$ le variabili booleane che occorrono in $f$; sia $X = \{x_1, ..., x_n\}$. Pertanto i $2n$ letterali che possono essere disposti in .OR. entro le clausole di $f$ sono le $n$ variabili $x_1, ..., x_n$ e le loro negazioni $\overline{x}_1, ..., \overline{x}_n$. Assumiamo che la CNF $f$ consti di $m$ clausole, ossia $f = \bigwedge_{j=1}^{m} C_j$. Ogni clausola $C_j$, $j = 1, ..., m$, consta di un sottoinsieme dei letterali che essa dispone in .OR. logico. Sia $\mathcal{C} = \{C_1, ..., C_m\}$. Come istanza della nostra forma decisionale di Min Set Cover si consideri un budget di $B \coloneqq n$ per l'ipergrafo $H_f$ con insieme dei nodi $V \coloneqq \mathcal{C} \cup X$ ed insieme degli iperarchi $\mathcal{E} \coloneqq \{\{x_i\} \cup \{C \in \mathcal{C} : C \ni x_i\} : x_i \in X\} \cup \{\{\overline{x}_i\} \cup \{C \in \mathcal{C} : C \ni \overline{x}_i\} : x_i \in X\}$.

La validità della riduzione poggia sui seguenti fatti:

**Fact [easy lemma]:** Sia $\varphi : X \longrightarrow \{\top, \bot\}$ un assegnamento di verità alle $n$ variabili che soddisfi $f$ (ossia, secondo il quale la formula $f$ valuti a vero, sia cioè soddisfatta). Allora un set cover $\mathcal{E}'$ di $H_f$ con $|\mathcal{E}'| = B$ può essere ottenuto mettendo in $\mathcal{E}'$ tutti gli insiemi in $\mathcal{E}$ della forma $\{x_i\} \cup \{C \in \mathcal{C} : C \ni x_i\}$ con $\varphi(x_i) = \top$ e tutti gli insiemi della forma $\{\overline{x}_i\} \cup \{C \in \mathcal{C} : C \ni \overline{x}_i\}$ con $\varphi(x_i) = \bot$.

**Fact [hard lemma]:** Sia $\mathcal{E}' \subseteq \mathcal{E}$ un insieme di iperarchi di $H_f$ che costituiscono un set cover di $H_f$ con $|\mathcal{E}'| \leq B$. Allora $|\mathcal{E}'| = B$ dato che $\mathcal{E}'$ deve coprire quantomeno $X \subseteq V$ e considerato che $|X| = n = B$ mentre ogni set in $\mathcal{E}$ contiene precisamente un solo elemento in $X$. In verità, per ogni $i = 1, ..., n$, sono precisamente due i set di $\mathcal{E}$ che contengono l'elemento $x_i$ di $X$. Pertanto $\mathcal{E}'$ contiene uno di questi due e non l'altro. Si consideri pertanto l'assegnamento di verità $\varphi : X \longrightarrow \{\top, \bot\}$ con $\varphi(x_i) = \top$ se e solo se $\mathcal{E}'$ contiene il set della forma $\{x_i\} \cup \{C \in \mathcal{C} : C \ni x_i\}$ piuttosto che non il set della forma $\{\overline{x}_i\} \cup \{C \in \mathcal{C} : C \ni \overline{x}_i\}$. Si noti che ogni clausola in $\mathcal{C}$ è soddisfatta da $\varphi$ in quanto $\mathcal{E}'$ deve coprire quantomeno $\mathcal{C} \subseteq V$.

**Fact [corollary of hard and easy lemma]:** Una CNF $f$ è soddisfacibile se e solo se l'ipergrafo $H_f$ ammette un set cover di cardinalità $B$.

Ne conseque l'NP-complessità della versione decisionale del problema Min Set Cover. Ne conseque l'NP-hardness del problema Min Set Cover.

**Esercizio 2 (con 12 richieste: 1+1+1+1+1+1+1+1+2+1+1+2 = 14 punti [programmazione dinamica]):**

La seguente tabella offre, nella sua seconda riga, una sequenza $S$ di numeri naturali (la prima riga, a caratteri in neretto, 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** | **26** | **27** |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 66 | 45 | 54 | 85 | 95 | 52 | 63 | 74 | 80 | 51 | 71 | 94 | 56 | 77 | 65 | 90 | 92 | 55 | 57 | 49 | 72 | 91 | 59 | 82 | 92 | 75 | 58 |

**Richieste dell'Esercizio 2**

**2 .1** ( 1 pt, DP: last_in_pos ) Alla tabella si aggiunga una riga che in ogni posizione $i$, con $1 \leq i \leq n$, riporti la massima lunghezza di una sottosequenza strettamente decrescente di $S$ che di $S$ prenda l'elemento in posizione $i$ come suo ultimo elemento.

**2 .2** ( 1 pt, DP: first_in_pos ) Si aggiunga una riga che in ogni posizione $i$, con $1 \leq i \leq n$, riporti la massima lunghezza di una sottosequenza strettamente decrescente di $S$ che di $S$ prenda l'elemento in posizione $i$ come suo primo elemento.

**2 .3** ( 1 pt, opt_sol: libera ) Trovare una sottosequenza strettamente decrescente di $S$ di massima lunghezza. Specificare quanto è lunga e fornirla.

**2 .4** ( 1 pt, certify opt ) Fornire un minimo numero di sottosequenze mai decrescenti tali che ogni elemento della sequenza originale in input ricada in almeno una di esse. Specificare quante sono e fornirle.

**2 .5** ( 1 pt, opt_sol: last ) Trovare una sottosequenza strettamente decrescente di $S$ di massima lunghezza tra quelle che terminano con l'elemento in posizione 19. Specificare quanto è lunga e fornirla.

**2 .6** ( 1 pt, opt_sol: left ) Trovare una sottosequenza strettamente decrescente di $S$ di massima lunghezza tra quelle che di $S$ non prendono alcun elemento di indice inferiore a 8. Specificare quanto è lunga e fornirla.

**2 .7** ( 1 pt, opt_sol: prende ) Trovare una sottosequenza strettamente decrescente di $S$ di massima lunghezza tra quelle che di $S$ prendono l'elemento in posizione 13. Specificare quanto è lunga e fornirla.

**2 .8** ( 1 pt, Z-sequenza ) Una sequenza è detta una Z-sequenza, o sequenza strettamente decrescente con al più un ripensamento, se esiste un indice $i$ tale che ciascuno degli elementi della sequenza, esclusi al più il primo e l'$i$-esimo, è strettamente minore dell'elemento che lo precede. Trovare la più lunga Z-sequenza che sia una sottosequenza della sequenza data. Specificare quanto è lunga e fornirla.

**2 .9** ( 2 pt, quante opt sol: libere ) Le sottosequenze di $S$ sono $2^n$, in corrispondenza biunivoca coi sottoinsiemi degli indici degli elementi di $S$ che includono. Stabilire quante siano le sottosequenze strettamente decrescenti di $S$ di massima lunghezza.

**2 .10** ( 1 pt, opt_sol: last ) Quante sono le sottosequenze strettamente decrescenti di $S$ di massima lunghezza tra quelle che prendono l'elemento in posizione 19 come loro ultimo elemento?

**2 .11** ( 1 pt, opt_sol: first ) Quante sono le sottosequenze strettamente decrescenti di $S$ di massima lunghezza tra quelle che prendono l'elemento in posizione 8 come loro primo elemento?

**2 .12** ( 2 pt, quante opt sol: con ) Quante sono le sottosequenze strettamente decrescenti di $S$ di massima lunghezza tra quelle che di $S$ includono l'elemento in posizione 13?

**Svolgimento esercizio 2 .**

**Richiesta 1.** Per rispondere alle prime richieste conviene compilare preventivamente un paio di tabelle di programmazione dinamica, di fatto quelle oggetto delle prime due richieste.

Ecco la prima tabella esplicitamente richiesta (`max_len_last_at`, compilata da sinistra):

| **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** | **26** | **27** |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 2 | 2 | 1 | 1 | 3 | 2 | 2 | 2 | 4 | 3 | 2 | 4 | 3 | 4 | 3 | 3 | 5 | 5 | 6 | 4 | 4 | 5 | 5 | 3 | 6 | 7 |
| 66 | 45 | 54 | 85 | 95 | 52 | 63 | 74 | 80 | 51 | 71 | 94 | 56 | 77 | 65 | 90 | 92 | 55 | 57 | 49 | 72 | 91 | 59 | 82 | 92 | 75 | 58 |

**Richiesta 2.** Ed ecco la seconda (`max_len_first_at`), compilata da destra (la nuova riga subito sotto a quella coi valori della sequenza in input):

| **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** | **26** | **27** |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 2 | 2 | 1 | 1 | 3 | 2 | 2 | 2 | 4 | 3 | 2 | 4 | 3 | 4 | 3 | 3 | 5 | 5 | 6 | 4 | 4 | 5 | 5 | 3 | 6 | 7 |
| 66 | 45 | 54 | 85 | 95 | 52 | 63 | 74 | 80 | 51 | 71 | 94 | 56 | 77 | 65 | 90 | 92 | 55 | 57 | 49 | 72 | 91 | 59 | 82 | 92 | 75 | 58 |
| 5 | 1 | 4 | 6 | 7 | 3 | 4 | 5 | 5 | 2 | 4 | 6 | 3 | 4 | 3 | 4 | 5 | 2 | 2 | 1 | 3 | 4 | 2 | 3 | 3 | 2 | 1 |

Il fatto che il massimo valore sia 7 sia nella riga prodotta per evadere la Richiesta 1 che nella riga ora inserita per evadere la Richiesta 2 viene a verifica parziale della correttezza dei conteggi.

**Richiesta 3.**

La massima lunghezza di una sottosequenza strettamente decrescente di $S$ è pertanto appunto **7**. Volendo ricostruire una sottosequenza strettamente decrescente di tale lunghezza si procede a ritroso rispetto all'ordine in cui i numeri nella riga (possiamo lavorare su quella prodotto nella Richiesta 1, oppure su quella prodotto nella Richiesta 2, è in differente) sono stati calcolati, ed utilizzarli come angioletti compagnetti (oracoli che ci anticipano le conseguenze di ogni nostra scelta, se includere o meno l'elemento nella posizione corrente del nostro esodo). Affidandoci a tali grilli parlanti diviene facile affrontare a colpo sicuro ogni scelta guidati dalla forza della promessa.

Le soluzioni ottime di un problema combinatorico come questo possono essere anche in numero esponenziale, ma eccone una:

| **pos** | **5** | **12** | **17** | **22** | **24** | **26** | **27** |
|---|---|---|---|---|---|---|---|
| $S[i]$ | 95 | 94 | 92 | 91 | 82 | 75 | 58 |

E' facile verificare l'ammissibilità di tale soluzione, ma come verificarne l'ottimalità?

**Richiesta 4.**

Poichè una sottosequenza strettamente decrescente ed una sottosequenza mai decrescente non possono condividere più di un singolo elemente della sequenza di riferimento, allora un set di $k$ sottosequenze mai decrescenti che coprano ogni elemente della sequenza di riferimento dimostrano che nessuna sottosequenza strettamente decrescente può contenere più di $k$ elementi (contenendone al più uno per ogni sottosequenza mai decrescente).

Ecco quindi il nostro certificato di ottimalità della soluzione fornita al punto precedente, nella forma di $k = 7$ sottosequenze mai decrescenti che coprano ogni elemente della sequenza $S$ di riferimento per questo esercizio:

| **pos** | **27** |
|---|---|
| $S[i]$ | 58 |

| **pos** | **26** |
|---|---|
| $S[i]$ | 75 |

| **pos** | **23** | **24** | **25** |
|---|---|---|---|
| $S[i]$ | 59 | 82 | 92 |

| **pos** | **18** | **19** | **20** | **21** | **22** |
|---|---|---|---|---|---|
| $S[i]$ | 55 | 57 | 49 | 72 | 91 |

| **pos** | **13** | **14** | **15** | **16** | **17** |
|---|---|---|---|---|---|
| $S[i]$ | 56 | 77 | 65 | 90 | 92 |

| **pos** | **6** | **7** | **8** | **9** | **10** | **11** | **12** |
|---|---|---|---|---|---|---|---|
| $S[i]$ | 52 | 63 | 74 | 80 | 51 | 71 | 94 |

| **pos** | **1** | **2** | **3** | **4** | **5** |
|---|---|---|---|---|---|
| $S[i]$ | 66 | 45 | 54 | 85 | 95 |

**Richiesta 5.** La massima lunghezza di una sottosequenza strettamente decrescente di $S$ tra quelle che terminano con l'elemento in posizione 19 è il **5** che si trova nella colonna 19 della riga di programmazione dinamica `last_in_pos` introdotta con la Richiesta 1. La tecnica/mantra per ricostruire una soluzione ottima è quella di consultare ad ogni scelta la tabella di programmazione dinamica, come spiegato sopra. La parola di Javè non è univoca ma potente in ricchezza generativa: le soluzioni ottime di un problema combinatorico come questo possono essere anche in numero esponenziale. Per soddisfare la richiesta ci basta esibirne una:

| **pos** | **5** | **12** | **14** | **15** | **19** |
|---|---|---|---|---|---|
| $S[i]$ | 95 | 94 | 77 | 65 | 57 |

**Richiesta 6.** La massima lunghezza di una sottosequenza strettamente decrescente di $S$ tra quelle che di $S$ non prendono alcun elemento di indice inferiore a 8 è **6** come si può evincere della riga di programmazione dinamica `first_in_pos` introdotta con la Richiesta 2. (Volendo rendere questa riga più leggibile, si potrebbe aggiungere come ulteriore riga di programmazione dinamica il calcolo del massimo valore che lei offre a destra di ciascuna posizione, poi consultare tale riga nella colonna 8.) La tecnica/mantra per ricostruire una soluzione ottima è sempre quella (anche per altri problemi di ottimizzazione risolti tramite programmazione dinamica). Per soddisfare la richiesta ci basta esibire una soluzione ottima:

| **pos** | **12** | **17** | **22** | **24** | **26** | **27** |
|---|---|---|---|---|---|---|
| $S[i]$ | 94 | 92 | 91 | 82 | 75 | 58 |

**Richiesta 7.** Combinando il dato in colonna 13 dalle due tabelle compilate per le Richieste 1 e 2 scopriamo che la massima lunghezza di una sottosequenza strettamente decrescente di $S$ che prenda l'elemento in posizione 13 è $\mathbf{4 + 3 - 1 = 6}$. Ricostruendo al rispettivo ritroso entrambe (verso destra oppure verso sinistra) le ali di una soluzione ottima si ottiene:

| **pos** | **5** | **9** | **11** | **13** | **18** | **20** |
|---|---|---|---|---|---|---|
| $S[i]$ | 95 | 80 | 71 | 56 | 55 | 49 |

**Richiesta 8.**

Per trovare un punto di ripensamento favorevole per la Z-sottosequenza di $S$ ho confrontato i valori nelle tabelle compilate per rispondere alle Richieste 1 e 2. Per semplificare tale confronto (da quadratico a lineare) ho preferito introdurre due ulteriori righe di PD (una, compilata da sinistra, riporta la massima lunghezza di una sottosequenza strettamente decrescente per ogni prefisso; l'altra, compilata da destra, riporta la massima lunghezza di una sottosequenza strettamente decrescente per ogni suffisso; sono ovvie le ricorrenze con cui calcolarle avvalendosi dei valori già prodotti per rispondere alle Richieste 1 e 2).

| **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** | **26** | **27** |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 2 | 2 | 2 | 2 | 3 | 3 | 3 | 3 | 4 | 4 | 4 | 4 | 4 | 4 | 4 | 4 | 5 | 5 | 6 | 6 | 6 | 6 | 6 | 6 | 6 | 7 |
| 1 | 2 | 2 | 1 | 1 | 3 | 2 | 2 | 2 | 4 | 3 | 2 | 4 | 3 | 4 | 3 | 3 | 5 | 5 | 6 | 4 | 4 | 5 | 5 | 3 | 6 | 7 |
| 66 | 45 | 54 | 85 | 95 | 52 | 63 | 74 | 80 | 51 | 71 | 94 | 56 | 77 | 65 | 90 | 92 | 55 | 57 | 49 | 72 | 91 | 59 | 82 | 92 | 75 | 58 |
| 5 | 1 | 4 | 6 | 7 | 3 | 4 | 5 | 5 | 2 | 4 | 6 | 3 | 4 | 3 | 4 | 5 | 2 | 2 | 1 | 3 | 4 | 2 | 3 | 3 | 2 | 1 |
| 7 | 7 | 7 | 7 | 7 | 6 | 6 | 6 | 6 | 6 | 6 | 6 | 5 | 5 | 5 | 5 | 5 | 4 | 4 | 4 | 4 | 4 | 3 | 3 | 3 | 2 | 1 |

In fondo ogni Z-sottosequenza può essere spezzata in due sottosequenze strettamente decrescenti, una sinistra ed una destra. La tabella quì sopra può essere compilata in tempo lineare e consente di stabilire in tempo lineare che possiamo far terminare in posizione 10 la sottosequenza sinistra e far partire in posizione 12 la sottosequenza destra.

Procedendo a ritroso sulle due ali come da competenza già esibita otteniamo la seguente Z-sequenza di massima lunghezza:

| **pos** | **1** | **3** | **6** | **10** | **12** | **17** | **22** | **24** | **26** | **27** |
|---|---|---|---|---|---|---|---|---|---|---|
| $S[i]$ | 66 | 54 | 52 | 51 | 94 | 92 | 91 | 82 | 75 | 58 |

Essa è lunga **10**.

**Richiesta 9.**

Dalla seguente tabella evinco che sono **1** le sottosequenze strettamente decrescenti di $S$ di massima lunghezza. La riga centrale della tabella riporta la sequenza in input, mentre le due righe collocate sopra (sotto) di essa sono compilate da sinistra (da destra). Le righe limitrofe alla sequenza in input (`max_len_last_at` e `max_len_first_at`) le abbiamo gia incontrate in precedenti tabelle ma vengono quì riprese in quanto ancillari al computo delle due ulteriori righe che ci servono ora (`num_opts_last_at` e `num_opts_first_at`). L'idea è che in ogni posizione $i$, con $1 \leq i \leq n$, la riga `num_opts_last_at` (`num_opts_first_at`) riporti il numero delle più lunghe sottosequenze strettamente decrescenti di $S$ che di $S$ prendano l'elemento in posizione $i$ come loro ultimo (primo) elemento.

| **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** | **26** | **27** |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 1 | 1 | 1 | 1 | 1 | 3 | 2 | 2 | 1 | 4 | 1 | 4 | 3 | 7 | 1 | 1 | 11 | 7 | 18 | 5 | 1 | 13 | 1 | 1 | 1 | 1 |
| 1 | 2 | 2 | 1 | 1 | 3 | 2 | 2 | 2 | 4 | 3 | 2 | 4 | 3 | 4 | 3 | 3 | 5 | 5 | 6 | 4 | 4 | 5 | 5 | 3 | 6 | 7 |
| 66 | 45 | 54 | 85 | 95 | 52 | 63 | 74 | 80 | 51 | 71 | 94 | 56 | 77 | 65 | 90 | 92 | 55 | 57 | 49 | 72 | 91 | 59 | 82 | 92 | 75 | 58 |
| 5 | 1 | 4 | 6 | 7 | 3 | 4 | 5 | 5 | 2 | 4 | 6 | 3 | 4 | 3 | 4 | 5 | 2 | 2 | 1 | 3 | 4 | 2 | 3 | 3 | 2 | 1 |
| 2 | 1 | 1 | 12 | 1 | 1 | 1 | 4 | 8 | 1 | 4 | 1 | 1 | 4 | 3 | 2 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

Anche quì ho preferito compilare sia delle righe di PD (programmazione dinamica) da sinistra (che rispondono a domande per ogni prefisso di $S$) che delle righe di PD da destra (che rispondono alle domande speculari per ogni suffisso di $S$) in modo da trovare verifica almeno parziale nel confronto dei valori ottenuti per i due (quello sinistro e quello destro) sotto-problemi più grossi, ultimi a cadere come

birilli sotto i colpi della ovvia ricorrenza che definisce la rispettiva famiglia. Per altro questo doppio conteggio mi serve se poi voglio rispondere in economia a tutte le richieste che seguono.

**Richiesta 10.** Sono **7** le sottosequenze strettamente decrescenti di $S$ di massima lunghezza tra quelle che prendono l'elemento in posizione 19 come loro ultimo elemento.

**Richiesta 11.**

Sono **4** le sottosequenze strettamente decrescenti di $S$ di massima lunghezza tra quelle che prendono l'elemento in posizione 8 come loro primo elemento.

**Richiesta 12.**

Sono $\mathbf{4 = 4 \times 1}$ le sottosequenze strettamente decrescenti di $S$ di massima lunghezza tra quelle che di $S$ includono l'elemento in posizione 13.

**Esercizio 3 (con 7 richieste: 3+4+4+4+2+5+4 = 26 punti [grafi]):**
Ogni richiesta che faremo può essere agevolmente evasa utilizzando la *rappresentazione a liste di adiacenza* del grafo in questione. Per ogni nodo, si fornisce la lista degli archi entranti e quella degli archi uscenti, entrambe terminate da ";" e quando una è vuota resta indicata da un trattino ("-"). Ogni arco ricompreso nelle due liste associate ad un *nodo* è una coppia (*altro nodo*, *costo*). Alcune delle richieste che seguono si riferiscono al grafo non-diretto e/o non-pesato ottenuto ignorando semplicemente l'orientazione o il peso dell'arco.

A: (F,4), (J,2); (U,3), (B,5);
B: (A,5), (S,5); (C,5), (U,4), (G,4);
C: (F,5), (H,5), (B,5); (K,4);
D: (F,2); (U,5), (W,3), (R,3);
E: (O,4), (T,4); (P,4);
F: (K,5); (C,5), (D,2), (I,3), (A,4);
G: (B,4); (H,4), (J,4);
H: (G,4); (C,5), (I,4);
I: (F,3), (H,4); (J,3);
J: (I,3), (G,4); (A,2);
K: (C,4), (S,5); (F,5);
L: (M,1), (N,1); -;
M: (S,2); (L,1);
N: (S,2); (L,1);
O: -; (P,4), (E,4);
P: (O,4), (E,4); (Q,3);
Q: (P,3); (T,3);
R: (D,3), (X,5), (Z,4); -;
S: -; (B,5), (M,2), (N,2), (K,5);
T: (Q,3); (E,4);
U: (D,5), (A,3), (B,4); (V,5), (Z,3);
V: (U,5); (W,5), (Y,3);
W: (D,3), (V,5); (X,4);
X: (W,4); (Y,4), (R,5);
Y: (V,3), (X,4); (Z,5);
Z: (U,3), (Y,5); (R,4);

Di massima, i vari elementi in risposta ad una richiesta potranno essere forniti compilando una diversa riga di una tabella la cui prima riga contiene i nomi dei nodi in ordine alfabetico. Se preferisci tenere separare le tabelle delle risposte per le varie richieste, assicurati che ciascuna abbia tali intestazioni di colonna per non complicare gli atti di verifica.

Ad esempio, la prima richiesta chiede di individuare le componenti connesse del grafo (non-diretto e non-pesato), e sia la BFS che la DFS (puoi scegliere) sono due semplici ed efficaci (lineari) algoritmi cui basta la rappresentazione a liste di adiacenza per produrre ogni elemento di risposta richiesto.

**Richiesta di studiare le componenti connesse.** In un grafo non-diretto $G = (V, E)$ due nodi si dicono *collegati* se e solo se tra di essi vi è un cammino. Essere collegati è una relazione di equivalenza le cui classi sono chiamate le *componenti connesse* di $G$. Di fatto, un $V' \subseteq V$ e il sottografo indotto $G[V']$ sono entrambi chiamati componenti connesse di $G$ se $V'$ è massimale con la proprietà che ogni suoi due nodi siano collegati. Non ti serve disegnare $G$ per esprimere efficacemente la partizione di $V$ in componenti connesse: in una tabella delle risposte che ha per etichette di colonna i nomi dei nodi presi in ordine alfabetico, e scelto come rappresentante di ciascuna classe il suo primo nodo in ordine alfabetico, la partizione in componenti connesse può essere espressa con una sola riga (1pt) della tabella che per ogni nodo $v$ in $V$ riporta il nome del rappresentante della classe di $v$. Analizziamo l'efficacia di tale codifica: per verificare che due nodi cui hai attribuito rappresentanti diversi appartengano

effettivamente a classi diverse basterà controllare che i due estremi di ogni arco dichiarino lo stesso rappresentante. Pertanto, su questo versante, non serve aggiungere un certificato. Ma come verificare che tutti i nodi cui sia stato assegnato uno stesso rappresentante $x$ siano effettivamente collegati ad $x$? Sia io (Arturo) che tù (Merlino) vogliamo assicurarci che la vera partizione di $V$ in componenti connesse non sia in realtà una sotto-partizione di quella che tu consegni. Collaborare è una risorsa. L'accordo è che per ogni componente connessa $V_x \subseteq V$ Merlino fornisca inoltre un albero ricoprente del sottografo indotto $G[V_x]$. Un modo pratico per entrambi di passarsi questo albero è di orientarlo in modo che abbia radice in $x$, in questo modo ogni nodo $v \neq x$ avrà un unico padre. Pertanto, in una seconda riga (1pt) della tabella, ogni nodo riporterà l'identità del proprio padre (le radici, ossia i rappresentanti delle rispettive classi di equivalenza, riporteranno semplicemente sè stesse). Come ulteriore garbo ad Arturo (1pt), potresti fornire una terza informazione certificante (dell'aciclicità dei puntatori al padre) specificando, per ogni nodo, la sua distanza generazionale dalla rispettiva radice (ossia la lunghezza del cammino che si otterrebbe risalendo alla radice seguendo i padri).

**Richieste dell'Esercizio 3**

**3 .1** ( 3 pt, componenti connesse ) Identificazione delle componenti connesse (1 punto per ogni elemento o co-elemento certificante, ossia per ogni riga della tabella delle risposte come precisata sopra).
**3 .2** ( 4 pt, 2-colorability ) Garantiamo che ogni componente connessa possa essere resa bipartita con la rimozione di al più 1 arco (fù mio errore, in realtà erano 2). Per le componenti connesse non-bipartite fornire un ciclo dispari (1pt) e indicazione dell'arco da rimuovere (1pt). Fornire, in una riga della tabella delle risposte, una 2-colorazione $c : V \longrightarrow \{0, 1\}$ valida a valle della rimozione degli archi indicati (2pt se colori tutti i nodi, 1 se colori almeno tutti quelli delle componenti già in partenza bipartite).
**3 .3** ( 4 pt, cammini minimi ) In una riga della tabella si riporti la distanza (1pt) di ogni nodo da $S$. Si fornisca un albero dei cammini minimi dal nodo $S$ ai nodi raggiungibili da $S$ (per ogni nodo il padre, (1pt)). Fuori dalla tabella, si riporti la lista delle scelte alternative per quei nodi che potrebbero optare per un altro padre (1pt), e dire quanti sono i diversi alberi dei cammini minimi (1pt).
**3 .4** ( 4 pt, cammini minimi diretti ) La stessa consegna di cui al punto precedente, e con lo stesso schema attributivo dei punti, ma ora con riferimento al grafo diretto. Per ogni lista che hai ricevuto in input, prima del ";" hai gli archi entranti nel nodo e dopo il ";" quelli uscenti.
**3 .5** ( 2 pt, MST: algoritmo di Prim ) Sia $\overline{C}$ la componente connessa di $G$ contenente il nodo $S$. Si fornisca un albero ricoprente di peso minimo per $\overline{C}$, orientato con radice in $S$ (1pt per la riga dei padri, 1pt per la distanza generazionale da $S$).
**3 .6** ( 5 pt, MST: cicli e tagli, 5=1+1+1+2 ) Per ciascuno degli archi $IH$, $IJ$ e $ZR$, dire (1pt se ogni arco è classificato correttamente) se esso appartenga a tutti, oppure a nessuno, oppure a qualcuno ma non tutti gli alberi ricoprenti di peso minimo di $\overline{C}$, fornendo i cicli e/o tagli certificati (1pt per ogni certificato non-ridondante). Organizzarsi per fornire in modo chiaro ogni elemento di risposta (non si prestano ad essere espressi con una riga di tabella delle risposte come per altre richieste).
**3 .7** ( 4 pt, DAG recognition ) Con riferimento al grafo diretto, garantiamo che ogni componente connessa $V_x$ possa essere resa aciclica rimuovendo al più un arco. Per ogni componente connessa che contenga cicli diretti se ne fornisca uno (1pt) e si dia indicazione dell'arco da rimuovere (1pt) per renderla aciclica. In una riga della tabella si fornisca un ordinamento topologico del grafo ottenuto con la rimozione degli archi indicati, ossia una numerazione dei nodi tale che per ogni arco il numero assegnato al nodo testa superi il numero assegnato al nodo coda. (2pt se lo fai per tutte le componenti, 1 se gestisci solo quelle che erano dei DAG già in partenza).

**Svolgimento esercizio 3 .**
**richiesta 1 (componenti connesse).** Per esaudire la prima richiesta usiamo la BFS multi-start. Essa

simula la propagazione del fuoco, che raggiunge tutti i nodi per i quali esista un cammino che parte dal nodo dove il nodo è stato appiccato. La lanciamo dapprima dal nodo $A$ e una seconda volta dal nodo $X$ che non è stato raggiunto dal fuoco appiccato in $A$. A valle di questi due avvi della BFS, tutti i nodi sono stati raggiunti dal fuoco avviato da $A$ oppure da quello avviato da $X$. Questo significa che ci sono due componenti connesse, quella dei nodi raggiunti dal fuoco $A$ è quella dei nodi raggiunti dal fuoco $X$. Merlino deve solo verificare che ogni arco di $G$ ha entrambi gli estremi colorati $A$ oppure entrambi gli estremi colorati $X$. E poi verificherà le altre due righe prodotte facendo accorto book-keeping di quanto avviene durante la BFS: il vettore dei padri che per ogni nodo dice chi gli ha trasmesso il fuoco, e quello che per ogni nodo dice quanto lui disti dalla radice entro l'albero definito dal vettore dei padri (la sua distanza sarà pari a quella del padre incrementata di 1).

| A | B | C | D | E | F | G | H | I | J | K | L | M | N | O | P | Q | R | S | T | U | V | W | X | Y | Z | node |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| A | A | A | A | E | A | A | A | A | A | A | A | A | A | E | E | E | A | A | E | A | A | A | A | A | A | class_rep |
| A | S | F | U | E | D | B | G | H | I | C | M | S | L | E | O | P | X | K | Q | A | W | D | Y | V | R | daddy |

**richiesta 2 (2-colorabilità).**

Per rendere bipartito il grafo basta rimuovere l'arco $EO$ dalla componente connessa $\{E, T, O, P, Q\}$ e i due archi $BL$ e $FK$ dalla componente connessa di rappresentante $A$. Che dalla prima componente serva rimuovere almeno un arco è certificato dalla presenza del triangolo $EOP$. Per la seconda componente esibiamo i triangoli $F\mathbb{C}K$ e $BAL$ ed osserviamo che non hanno archi in comune. Un certificato di 2-colorabilità del grafo così ottenuto è dato in tabella come meglio conviene (garbo a King Arthur controllore).

| A | B | C | D | E | F | G | H | I | J | K | L | M | N | O | P | Q | R | S | T | U | V | W | X | Y | Z | node |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| A | S | F | U | E | D | B | G | H | I | C | M | S | L | P | Q | T | X | K | E | A | W | D | Y | V | R | dad |

**richiesta 3 (cammini minimi nel grafo non diretto).**

Un'albero dei cammini minimi dal nodo $S$ è rappresentato in tabella, dove per ogni nodo si è avuto cura di riportare la distanza dal nodo $S$.

| A | B | C | D | E | F | G | H | I | J | K | L | M | N | O | P | Q | R | S | T | U | V | W | X | Y | Z | node |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 10 | 5 | 9 | 12 | - | 10 | 9 | 13 | 13 | 12 | 5 | 3 | 2 | 2 | - | - | - | 15 | 0 | - | 9 | 14 | 15 | 19 | 17 | 12 | dist |
| B | S | K | F | - | K | B | G | F | A | S | M | S | S | - | - | - | D | S | - | B | U | D | W | Z | U | dad |

| A | B | C | D | E | F | G | H | I | J | K | L | M | N | O | P | Q | R | S | T | U | V | W | X | Y | Z | node |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 10 | 5 | 9 | 12 | - | 10 | 9 | 13 | 13 | 12 | 5 | 3 | 2 | 2 | - | - | - | 15 | 0 | - | 9 | 14 | 15 | 19 | 17 | 12 | dist |
| B | S | K | F | - | K | B | G | F | A | S | M;N | S | S | - | - | - | D | S | - | B | U | D | W | Z;V | U | dads |

Per verificare l'ottimalità dell'albero basta pertanto controllare che nessun arco del coalbero è più corto delle differenza delle distanze in coppa ai suoi estremi.

Con questa verifica è anche possibile identificare quali nodi abbiano alternative (e quali) per la scelta del padre nell'albero dei cammini ottimi. In questo caso vediamo che sono presenti precisamente $2 \times 2 = 4$ alberi dei cammini minimi in quanto il nodo $L$ ha precisamente 2 opzioni per il padre (il nodo $M$ oppure il nodo $N$) come anche il nodo $Y$ ha precisamente 2 opzioni (il nodo $Z$ oppure il nodo $V$), mentre nessun altro nodo ha scelta.

**richiesta 4 (cammini minimi nel grafo diretto).**

Un'albero dei cammini minimi dal nodo $S$ è rappresentato in tabella, dove per ogni nodo si è avuto cura di riportare la distanza dal nodo $S$.

| A | B | C | D | E | F | G | H | I | J | K | L | M | N | O | P | Q | R | S | T | U | V | W | X | Y | Z | node |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 14 | 5 | 10 | 12 | - | 10 | 9 | 13 | 13 | 13 | 5 | 3 | 2 | 2 | - | - | - | 15 | 0 | - | 9 | 14 | 15 | 19 | 17 | 12 | dist |
| F | S | B | F | - | K | B | G | F | G | S | M | S | S | - | - | - | D | S | - | B | U | D | W | V | U | dad |

| A | B | C | D | E | F | G | H | I | J | K | L | M | N | O | P | Q | R | S | T | U | V | W | X | Y | Z | node |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 14 | 5 | 10 | 12 | - | 10 | 9 | 13 | 13 | 13 | 5 | 3 | 2 | 2 | - | - | - | 15 | 0 | - | 9 | 14 | 15 | 19 | 17 | 12 | dist |
| F | S | B | F | - | K | B | G | F | G | S | M;N | S | S | - | - | - | D | S | - | B | U | D | W | V | U | dads |

Per verificare l'ottimalità dell'albero basta pertanto controllare che nessun arco del coalbero è più corto delle differenza delle distanze in coppa ai suoi estremi.

Con questa verifica è anche possibile identificare quali nodi abbiano alternative (e quali) per la scelta del padre nell'albero dei cammini ottimi. In questo caso vediamo che sono presenti precisamente 2 alberi dei cammini minimi in quanto il nodo $L$ ha precisamente 2 opzioni per il padre (il nodo $M$ oppure il nodo $N$) mentre nessun altro nodo ha scelta.

**Una nota:** Succede spesso che per questo esercizio lo studente consegna una reportistica degli aggiornamenti alle distanze dei nodi come avvengono durante l'esecuzione di un algoritmo (tipicamente un Dijkstra). Questo va bene per quelli che io tendo a classificarmi come "i fogli della brutta", da cui cerco di recuperare dei punti se vedo che lo studente era riuscito comunque ad ottenere in essi qualcosa di concludente o quantomeno significativo, ma non è il modo più opportuno nè il più conveniente per rispondere alla consegna. La giusta forma per "la bella" è qualcosa come il disegno quì sopra (tutto questo documento di correzione, come i suoi fratelli, è prioritariamente inteso ad esemplificare come si debba rispondere alle consegne), e cerco di spiegarlo: di fronte ad una consegna dovete consegnare un prodotto finito, e in una forma conveniente per il committente (ben approssimato dal verificatore King Arthur). Se in uno stesso disegno del grafo, il più semplice e leggibile possibile, riportate sia le lunghezze degli archi che le distanze che col vostro lavoro avete computato per ciascun nodo, allora quella figura racconta già un sacco di cose sulla famiglia di tutti gli alberi dei cammini ottimi. Evidenziati inoltre gli archi di una particolare soluzione (di un particolare albero), King Arthur potrà verificare che:

1. per ogni arco di albero la sua lunghezza sia pari alla differenza tra i potenziali (le distanze) riportati in coppa ai nodi
2. per ogni arco di coalbero la sua lunghezza sia non-inferiore alla differenza tra i potenziali (le distanze) riportati in coppa ai nodi

Un punto chiave, a valore metodologico generale, è questo: il garbo verso King Arthur non solo è funzionale ad ottemperare congruamente una consegna, ma vi consente di evitare di consegnare soluzioni che contengano errori. Ben presto diventa quindi strumento dialettico di crescita, di problem solving, e di ricerca.

Tornando all'esame:

0. Se tutte le verifiche tornano, i punti sono in cassaforte.
1. Se uno degli atti di verifica 1 riscontra un errore, potrete correggerlo in modo immediato ripropagando i potenziali lungo l'albero.

2. Se uno degli atti di verifica 2 riscontra un errore, potrete individuare dei cammini migliori, aggiornando la vostra soluzione fino ad ora solo ammissibile per un albero dei cammini minimi.

E se nella verifica 2 per degli archi ottenente uguaglianza? Beh, state sensando tutti quegli archi che appartengono a qualche albero dei cammini minimi.

Cercate di fare tesoro generale di questa metodologia che ci arriva in parte dalla Ricerca Operativa ed in parte dalla Complessità Computazionale.

**richiesta 5 (Minimum Spanning Trees).**

Ecco gli archi di un MST (minimum spanning tree):

| L | L | A | D | M | A | D | D | F | I | P | Q | U | V | B | B | C | E | E | G | W | X | B | B | u |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| M | N | J | F | S | U | W | R | I | J | Q | T | Z | Y | U | G | K | O | T | H | X | Y | C | S | v |
| 1 | 1 | 2 | 2 | 2 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 4 | 4 | 4 | 4 | 4 | 4 | 4 | 4 | 5 | 5 | weight |

Il costo totale dei suoi archi è 77.

Ecco gli archi che devono appartenere ad ogni MST:

| A | A | C | D | D | D | F | I | L | L | P | Q | U | V | W | X | u |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| J | U | K | F | R | W | I | J | M | N | Q | T | Z | Y | X | Y | v |
| 2 | 3 | 4 | 2 | 3 | 3 | 3 | 3 | 1 | 1 | 3 | 3 | 3 | 3 | 4 | 4 | weight |

Ecco gli archi che non appartengono ad alcun MST:

| A | A | D | R | R | U | V | Y | u |
|---|---|---|---|---|---|---|---|---|
| B | F | U | X | Z | V | W | Z | v |
| 5 | 4 | 5 | 5 | 4 | 5 | 5 | 5 | weight |

Ecco gli archi che appartengono ad alcuni ma non a tutti gli MST:

| B | B | B | B | C | C | E | E | E | F | G | G | H | K | M | N | O | u |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| C | G | S | U | F | H | O | P | T | K | H | J | I | S | S | S | P | v |
| 5 | 4 | 5 | 4 | 5 | 5 | 4 | 4 | 4 | 5 | 4 | 4 | 4 | 5 | 2 | 2 | 4 | weight |

L'arco $HI$ è di peso minimo tra quelli con un estremo in $H$ e pertanto appartiene a qualche MST, ma non a tutti poichè di peso massimo nel ciclo $HI, IF, FD, DR, RZ, ZL, LB, BG, GH$.

L'arco $IJ$ appartiene a tutti gli MST in quanto arco di peso strettamente minimo del taglio che separa i nodi $J, A, L, Z$ da tutti gli altri nodi del grafo.

L'arco $ZR$ non appartiene ad alcun MST in quanto arco di peso strettamente massimo del ciclo $ZR, RD, DF, FI, IJ, JA, AL, LZ$.

**richiesta 7 (riconoscimento di DAG).**

Nella seguente tabella diamo la partizione dei nodi nelle componenti fortemente connesse del grafo (due nodi appartengono alla stessa componente se e solo se la seconda riga della tabella associa loro lo stesso numero):

| A | B | C | D | E | F | G | H | I | J | K | L | M | N | O | P | Q | R | S | T | U | V | W | X | Y | Z | node |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 7 | 7 | 7 | 8 | 6 | 7 | 7 | 7 | 7 | 7 | 7 | 5 | 4 | 3 | 2 | 6 | 6 | 15 | 1 | 6 | 9 | 10 | 11 | 12 | 13 | 14 | comp |

Il ciclo $(P,Q),(Q,T),(T,E),(E,P)$ ci dice che i nodi $P,Q,T,E$ sono tutti contenuti in una stessa componente fortemente connessa. Poichè gli archi $(O,P)$ e $(O,E)$ sono gli unici altri archi con un estremo in $\{P,Q,T,E\}$, ed entrambi sono diretti verso $\{P,Q,T,E\}$, allora $\{P,Q,T,E\}$ è una componente fortemente connessa (di testa, ossia un pozzo nel DAG delle componenti fortemente connesse).

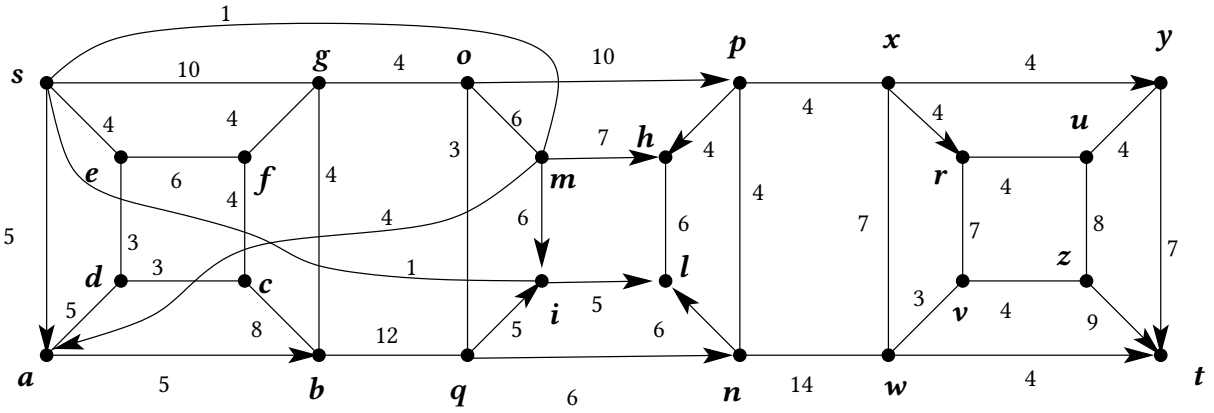
Il ciclo $(F,\mathbb{C}),(\mathbb{C},K),(K,F)$ ci dice che i nodi $F,\mathbb{C},K$ sono tutti contenuti in una stessa componente fortemente connessa. A questo punto, il cammino $(F,A),(A,B),(B,\mathbb{C})$ aggiunge a tale componente i nodi $A$ e $B$, in quanto parte dal nodo $F$ e termina nel nodo $\mathbb{C}$, entrambi già contenuti nella componente. In modo analogo, il cammino $(B,G),(G,J),(J,A)$ aggiunge a tale componente i nodi $G$ ed $J$. Infine i cammini $(G,C),(C,\mathbb{C})$ e $(F,I),(I,J)$ aggiungono i nodi $C$ e $I$. A questo punto anche questa seconda componente fortemente connessa è completa, mentre ogni altra componente è formata da un solo nodo e non serve pertanto fornire certificato della mutua raggiungibilità tra i suoi nodi.

Invece, come certificato del fatto che queste componenti non collassino ulteriormente, forniamo il seguente topological sort del DAG (Directed Acyclic Graph) delle componenti fortemente connesse (i nodi sono stati riordinati e per ogni nodo si riporta la posizione della componente connessa cui appartiene, come numerate secondo un ordine topologico):

| S | O | N | M | L | P | Q | T | E | B | G | H | C | K | F | I | J | A | D | U | V | W | X | Y | Z | R | node |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 2 | 3 | 4 | 5 | 6 | 6 | 6 | 6 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | pos |

A King Arthur basta verificare che, per ogni arco diretto $(u,v)$, la coda $u$ dell'arco precede la testa $v$ nell'ordinamento proposto (tranne che per quegli archi con entrambi i nodi contenuti entro una stessa componente fortemente connessa).

**Esercizio 4 (con 3 richieste: 3+2+3 = 8 punti [grafi]):**




**Richieste dell'Esercizio 4**

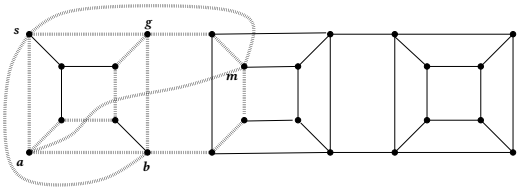
**4 .1** ( 3 pt, recognize planarity ) Dire, certificandolo, se siano planari o meno il grafo $G$ e il grafo $G'$ ottenuto da $G$ sostituendo l'arco $si$ con un arco $so$. (2 punti per il certificato di non-planarità, 1 per quello di planarità)

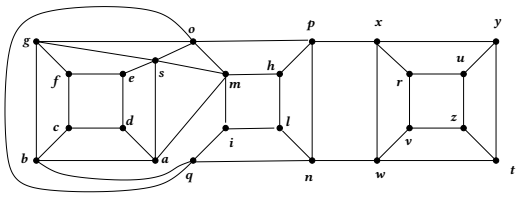
**4 .2** ( 2 pt, max flow ) In $G$, trovare un massimo flusso dal nodo $s$ al nodo $t$.

**4 .3** ( 3 pt, min cut ) In $G$, trovare un $s,t$-taglio minimo.

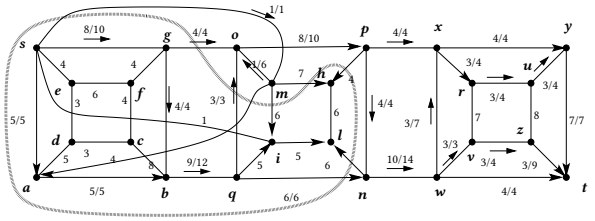
**Svolgimento esercizio 4 .**

La non-planarità di $G$ è certificata dalla $K_5$ subdivision in figura, sulla sinistra. Sulla destra, un planar embedding di $G'$ ne certifica la planarità.





Un flusso massimo ed un taglio minimo che ne dimostra l'ottimalità (non esibisco i passaggi spesi per ottenerli). Il flusso ha valore 13 e satura l'insieme degli archi che attraversano la curva tratteggiata portandosi dal lato di $s$ al lato di $t$. Questi 4 archi costituiscono pertanto un minimo $s,t$-taglio, anch'esso di valore 13. Esso certifica pertanto l'ottimalità del flusso proposto.



## CONSIGLI SU COME PREPARARSI ALL'ESAME

Per conseguire un voto per l'insegnamento di Ricerca Operativa devi partecipare ad un appello di esame. Il primo appello d'esame di ogni anno accademico ha luogo a giugno, dopo la conclusione del corso. Per gli appelli estivi in aula delta, non abbiamo controllo dell'aria condizionata e l'ambiente potrà risultarvi troppo freddo. Data la durata dell'appello consiglio di portarsi golfini, snack, acqua e matite o pennarelli colorati. Potete portarvi materiali cartacei ma non è consentita alcuna strumentazione elettronica. Dovete portare il tesserino col vostro numero di matricola.

Durante l'esame, dovrete lavore per almeno 4 ore a quella che definisco "una prova di cromatografia su carta". Serve per riconoscervi con ragionevole confidenza quanto avete lavorato, appreso, sedimentato. E trasformare questo in una proposta di voto la più congrua possibile. La logica dello svolgimento dell'esame deve essere quella di dimostrare al meglio le competenze acquisite andando con efficienza a raccogliere, dei molti punti messi in palio a vario titolo: cercate e concretizzate quelli che più vi convengono, non impegolatevi a dimostrare quello che non sapete o dove incontrate incertezze. Contano le risposte corrette, fornite in chiarezza, ed i certificati (in questo la struttura dell'esame ribadisce il ruolo metodologico ubiquito dei concetti di complessità computazionale propagandati nel corso). Tutto il resto (incluse le castronerie colossali ma anche le doppie risposte discordanti) non verrà conteggiato. Ricordate che in buona sostanza il voto corrisponderà al punteggio positivamente raccolto. I punti messi in palio ad ogni tema eccedono significativamente quanto necessario al raggiungimento dei pieni voti, gestitevi quindi per dimostrare le competenze che avete, senza impelagarvi dove avete invece delle lacune. Non ci interessano le vostre mancanze o lacune quanto piuttosto quello che dimostrate di saper fare.

L'esame presenta diverse tipologie di esercizi e domande su vari aspetti di quanto esposto a lezione. Nel prepararti all'esame, prendi a riferimento i testi e le correzioni dei temi precedenti che trovi al sito del corso:

```
http://profs.sci.univr.it/~rrizzi/classes/RO/index.html
```

Ogni esercizio è anche un'opportunità di apprendimento e di allenamento, sfruttalo al meglio senza sprecarlo. Una prima utilità è quella di testare la tua preparazione all'esame. Dopo aver letto il testo, consigliamo pertanto di svolgere l'esercizio quantomeno nella propria mente. Ma, in sufficiente numero di esemplari, poi anche materialmente, prestando attenzione ai tempi impiegati ed ai punti conseguiti. Solo a valle di un'esperienza almeno parziale con l'esercizio, passa alla lettura del documento di correzione. Se non sai come affrontare l'esercizio, sbircia sì la correzione, ma cercando di utilizzarla solo come suggerimento, cercando di riacquisire quanto prima autonomia nella conduzione dell'esercizio.

E se invece ti sembra di saper risolvere del tutto l'esercizio? Beh, a questo punto vale il converso: controlla che quanto hai in mente come soluzione corrisponda a quanto considerato e proposto come svolgimento opportuno. Nel confronto con la correzione proposta, presta attenzione non solo alle risposte in sè, ma anche a come esse vadano efficacemente offerte all'esaminatore/verificatore, ossia alla qualità dei tuoi certificati, alla precisione della tua dialettica, a come ottemperi il contratto implicito nella soluzione di un problema ben caratterizzato. In un certo senso, questo ti consentirà di raggiungere pragmaticamente quella qualità che in molti chiamano impropriamente ordine'', che ha valore e giustamente finisce, volenti o nolenti, per essere riconosciuta in ogni esame della vita. Ordine, ma noi preferiamo chiamarlo saper rispondere in chiarezza alla consegna'' non significa bella calligrafia o descrizioni prolisse, ma cogliere tempestivamente gli elementi salienti, quelli richiesti da contratto più o meno implicito. In questo le competenze che abbiamo messo al centro di questo insegnamento di ricerca operativa potranno renderti più consapevolmente ordinato. Lo scopo del documento di correzione non è tanto quello di spiegare come l'esercizio vada risolto ma piuttosto come le risposte vadano adeguatamente esibite pena il mancato conseguimento dei punti ad esse associati. Aggiungo che per le tipologie di esercizio classiche, descrizioni curate dei più noti algoritmi risolutori possono essere facilmente reperite altrove (perchè non collaborare a raccogliere una ricca collezione di link a tali sorgenti?).