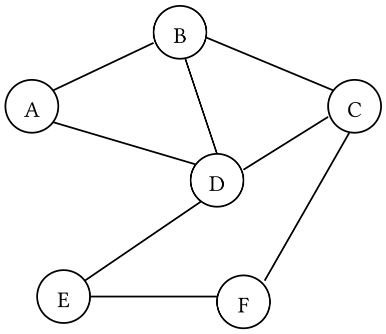
Esame di Ricerca Operativa - 23 giugno 2023

Facoltà di Scienze MM.FF.NN. - Verona

4 esercizi per 71 punti in palio (voto $\geq$ punti, 35$\longrightarrow$ 30 *e lode*, esercizio 4 aggiunto successivamente)

CORREZIONE -

Esercizio 1 (con 7 richieste: 2+3+2+1+4+3+6 = 21 punti [modellazione/riduzioni]):
Un *independent set* in un grafo $G = (V, E)$ è un sottoinsieme di nodi $S \subseteq V$ tale che, per ogni arco $e \in E$, $e =$ uv, al più uno tra u e v è contenuto in S. Un independent set S è detto *massimale* se non esiste alcun independent set $S' \supsetneq S$.



esempio. $\{A, C, F\}$ non è un independent set per via dell'arco CF. Invece $\{B, F\}$, $\{D, F\}$ e $\{A, E\}$ sono independent sets ma $\{A, E\}$ non è massimale in quanto $\{A, E\} \subsetneq \{A, C, E\}$.

Max Independent Set è il problema di trovare un'independent set di massima cardinalità per un grafo G dato in input.

Min Maximal Independent Set è il problema di trovare un maximal independent set di minima cardinalità per un grafo G dato in input.

Richieste dell'Esercizio 1

1.1 (2 pt, model via graphs) Vorresti invitare un massimo numero di amici alla tua festa di compleanno. Purtroppo su alcune coppie di amici serpeggia inimicizia. Formula in termini di Max Independent Set il tuo problema di invitare il maggior numero possibile di amici evitando però di rischiare situazioni spiacevoli. Spiega come definiresti il tuo grafo partendo dalla situazione che rilevi sul campo.

1.2 (3 pt, model as ILP) Formula come un problema di Programmazione Lineare Intera (PLI) il problema Max Independent Set per la specifica istanza G in figura.

1.3 (2 pt, generalize model) Estendi la tua formulazione PLI di Max Independent Set a un generico grafo $G = (V, E)$.

1.4 (1 pt, graph models) A corto di soldi, vorresti minimizzare gli inviti ma senza che alcuno abbia a lamentarsene. Sai che nessun escluso si lamenterà se alla festa sarà presente almeno una persona con cui è in inimicizia. Puoi formulare in termini di Min Maximal Independent Set il problema di invitare il minor numero possibile di amici evitando però che alcuno abbia rimostranze? Commenta.

1.5 (4 pt, model as ILP) Formulare nella PLI il problema Min Maximal Independent Set per il grafo G in figura.

1.6 (3 pt, generalize model) Formulare nella PLI il problema Min Maximal Independent Set per un grafo $G = (V, E)$ generico.

1.7 (6 pt, NP-hardness proof) Max Independent Set è noto essere NP-hard. Sfrutta questo fatto per dimostrare che anche Min Maximal Independent Set è NP-hard.

svolgimento esercizio 1.

Richiesta 1. Si consideri il grafo che ha per nodi i tuoi amici e dove due nodi sono adiacenti se e solo se non è possibile invitarli entrambi alla festa. Si noti che un sottoinsieme degli amici è un independent set di questo grafo se e solo se tra di loro non vi è alcuna inimicizia. Pertanto, il problema di nostro interesse può essere visto come la ricerca di un independent set di massima cardinalità entro un grafo dato.

Richiesta 2. Introduciamo una variabile $x_v \in \{0,1\}$ per ogni nodo $v \in \{A,B,C,D,E,F\}$, con l'idea che $x_v = 1$ significa "nodo v incluso nell'independent set soluzione" mentre $x_v = 0$ significa "nodo v NON incluso nell'independent set soluzione". Queste 6 variabili riescono quindi a catturare lo spazio delle possibili scelte, descrivendo compiutamente un qualsiasi sottoinsieme dei nodi.

Volendo massimizzare la cardinalità dell'independent set, la funzione obbiettivo sarà:

$$\max x_A + x_B + x_C + x_D + x_E + x_F$$

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

arco AB: $x_A + x_B \leq 1$
arco AD: $x_A + x_D \leq 1$
arco BD: $x_B + x_D \leq 1$
arco BC: $x_B + x_C \leq 1$
arco CD: $x_C + x_D \leq 1$
arco CF: $x_C + x_F \leq 1$
arco DE: $x_D + x_E \leq 1$
arco EF: $x_E + x_F \leq 1$

Questi vincoli impongono che il sottoinsieme di nodi descritto dal vetore x sia effettivamente un independent set.

Richiesta 3. Più in generale, introduciamo una variabile $x_v \in \{0,1\}$ per ogni nodo v del generico grafo $G = (V,E)$ che potremmo ricevere in input. Come sopra, $x_v = 1$ significa "nodo v incluso nell'independent set soluzione" mentre $x_v = 0$ significa "nodo v NON incluso nell'independent set soluzione".

La funzione obbiettivo sarà:

$$\max \sum_{v \in V} x_v$$

E la famiglia di vincoli è:

$$x_u + x_v \leq 1 \quad \text{per ogni arco } uv \in E$$

Richiesta 4. Per modellare questa seconda situazione ci si riferisca ora al MIN MAXIMAL INDEPENDENT SET ma cosiderando lo stesso identico grafo che avevamo definito ed introdotto per rispondere alla *richiesta 1*.

Richiesta 5. Si richiede ora di concentrarsi su quei soli independent sets che siano massimali. Poichè nella *richiesta 2* abbiamo già saputo catturare gli independet sets, basterebbe ora trovare come esprimere dei vincoli che filtrino via gli independent sets che non sono massimali. Introdotta pertanto una variabile $x_v \in \{0,1\}$ per ogni nodo $v \in \{A,B,C,D,E,F\}$, come fatto sopra, e introdotta la famiglia di vincoli che tra tutti i possibili sottoinsiemi di nodi concentrano l'attenzione sui soli independent set (gli stessi vincoli di cui sopra vanno bene), dobbiamo ora aggiungere una famiglia di vincoli che riesca ad esprimere la condizione di massimalità sugli independent set. Partiamo con l'esprimere tale condizione di massimalità in modo naturale, come da linguaggio umano, e procediamo per gradi a rimaneggiarla verso espressioni maggiormente operative:

con l'aggiunta di un qualsiasi nodo l'insieme cessa di essere un independent set

)

ovvero

ogni nodo escluso è adiacente ad almeno un nodo incluso

)

ovvero

per ogni nodo, esso è incluso oppure almeno uno dei suoi vicini è incluso

)

Va quindi aggiunta una seconda famiglia di vincoli, eccoli riportati uno per uno nello specifico:

nodo A**:** $x_A + x_B + x_D \geq 1$
nodo B**:** $x_B + x_A + x_C + x_D \geq 1$
nodo C**:** $x_C + x_B + x_D + x_F \geq 1$
nodo D**:** $x_D + x_A + x_B + x_C + x_E \geq 1$
nodo E**:** $x_E + x_D + x_F \geq 1$
nodo F**:** $x_F + x_E + x_C \geq 1$

Ovviamente dobbiamo invertire il senso dell'ottimizzazione, pur conservando lo stesso funzionale lineare per la funzione obiettivo:

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

Richiesta 6. Più in generale, introdotta una variabile $x_v \in \{0, 1\}$ per ogni nodo v, come visto sopra, e assunta la funzione obiettivo:

$$\min \sum_{v \in V} x_v$$

dobbiamo considerare due famiglie di vincoli:

indipendenza:

$$x_u + x_v \leq 1 \quad \text{per ogni arco } uv \in E$$

massimalità:

$$x_v + \sum_{u \in V(v)} x_u \geq 1 \quad \text{per ogni nodo } v \in V, \text{dove } V(v) \text{ indica l'insieme dei nodi adiacenti a } v.$$

Richiesta 7. Partendo da un generico grafo $G = (V, E)$ assegnatoci come istanza di Max Independent Set, costruiamo il grafo $G' = (V', E')$ dove $V' = \{v, v_a, v_b : v \in V\}$ e $E' = E \cup \{vv_a, vv_b : v \in V\}$. In pratica, il grafo G viene ottenuto da G introducendo due cloni v_a e v_b per ogni nodo v di G e rendendo v_a e v_b entrambi adiacenti al solo nodo v.

Lasciamo come esercizio di dimostrare i seguenti due lemmi che illustano l'idea della riduzione e ne dimostrano la correttezza.

Lemma easy: Se G ammette un independent set S con $|S| \geq k$, allora G' ammette un independent set massimale S' con $| \ S' \ | \leq k + 2(|V| - k)$.

Lemma hard: Se G' ammette un independent set massimale S' con $| \ S' \ | \leq k + 2(|V| - k)$ allora G ammette un independent set S con $|S| \geq k$.

Esercizio 2 (con 10 richieste: 1+1+1+1+1+1+1+4+5+3+2 = 20 punti [grafi]):
Un grafo $G = (V, E)$ è *bipartito* se e solo se è 2-colorabile, ossia se esiste una colorazione $c : V \longrightarrow$

$\{A, B\}$ tale che $c(u) \neq c(v)$ per ogni arco $\{u, v\} \in E$. Per $i = 1, 2, 3$, si consideri il grafo $G_i = (V, E_i)$, dove $V = \{A, B..., T\}$, ed E_1 è specificato in tabella.

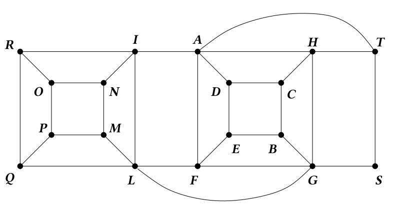
u	H	I	I	A	D	A	A	B	C	F	B	A	F	G	G	M	M	L	G	L	N	B	C	O	A	E	I	O	P	Q	S
v	T	L	N	H	E	I	T	C	H	L	E	D	G	H	S	N	P	M	L	Q	O	G	D	P	F	F	R	R	Q	R	T
	4	5	3	5	3	2	5	4	5	3	4	3	5	5	5	5	4	5	4	3	4	4	4	4	4	2	3	5	5	4	5

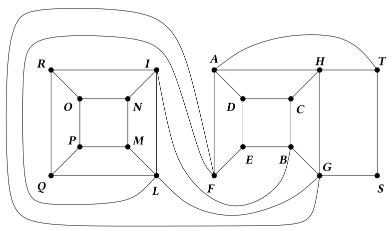
Invece, E_2 differisce da E_1 per la presenza dell'arco $\{B, I\}$ al posto dell'arco $\{A, I\}$, e E_3 differisce sempre da E_1 per la presenza dell'arco $\{C, I\}$, sempre al posto dell'arco $\{A, I\}$.

Richieste dell'Esercizio 2

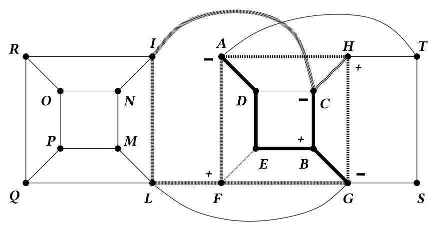
2.1 (1 pt, recognize planarity) Dire se G_1 sia planare o meno, fornendo certificato.
2.2 (1 pt, recognize planarity) Dire se G_2 sia planare o meno, fornendo certificato.
2.3 (1 pt, recognize planarity) Dire se G_3 sia planare o meno, fornendo certificato.
2.4 (1 pt, recognize 2-colorability) Dire se G_1 sia bipartito o meno, fornendo certificato.
2.5 (1 pt, 2-colorability, feasibility) Dire quale sia il minimo numero di archi la cui rimozione renda G_2 bipartito, fornendo certificato che rimuovere quel numero di archi basti.
2.6 (1 pt, 2-colorability, optimality) Dire quale sia il minimo numero di archi la cui rimozione renda G_2 bipartito, fornendo certificato che sia necessario rimuovere almeno quel numero di archi.
2.7 (4 pt, MST: cicli e tagli, 4=1+1+2) Per ciascuno degli archi DC, DE e QR, dire se esso appartenza a tutti, oppure a nessuno, oppure a qualcuno ma non tutti gli alberi ricoprenti di peso minimo di G_1, fornendo i certificati del caso.
2.8 (5 pt, MST: struttura famiglia) Si descriva la struttura della famiglia degli alberi ricoprenti di peso minimo per il grafo G_1, e si dica quanti essi siano.
2.9 (3 pt, cammini minimi, 3=1+1+1) Si evidenzino, in un disegno del grafo G_1, gli archi di un albero dei cammini minimi dal nodo S (si scriva la distanza da S in coppa ad ogni nodo, in modo che sia facile verificare l'ottimalità dell'albero fornito). Si descriva la struttura della famiglia degli alberi dei cammini minimi da S e si dica quanti sono.
2.10 (2 pt, DAG recognition) Si condiderino gli archi come diretti da u (nodo nella prima riga della tabella) a v (nodo nella seconda riga della tabella). Trovare un ciclo diretto nel grafo G_1 e fornirlo, oppure fornire un ordinamento topologico dei nodi di G_1.
svolgimento esercizio 2.
Planarità. Per rispondere alla prima richiesta conviene compilare prima un paio di tabelle di programmazione dinamica.

Il fatto che G_1 e G_2 siano planari è messo in evidenza dai rispettivi planar embedding in figura.





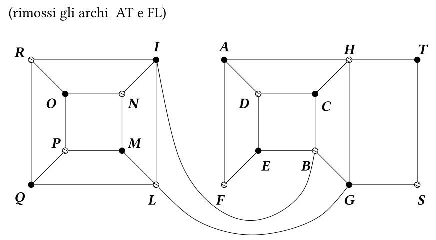
Il certificato di non planarità di G_3 è invece offerto evidenziando una suddivisione di $K_3, 3$ entro G_3.



Nello svolgimento di alcuni dei successivi punti (quelli dove i certificati sono più efficamente espressi con un disegno) converrà riferirsi al planar drawing del grafo G_1 fornito sopra.

2-**colorabilità.** Il grafo G_1 non è bipartito: si controlli la presenza del triangolo AHT.

Per rendere bipartito il grafo G_2 basta rimuovere gli archi AT e FL; un certificato di 2-colorabilità del grafo così ottenuto è dato in figura come meglio conviene (garbo a King Arthur controllore, che poi sono mè).



Il fatto che per rendere bipartito G_2 serva rimuovere almeno due archi è certificato dai cicli dispari AHT e $FLIBE$: si riscontri che essi non hanno archi in comune, e il verificatore deve poi solo controllare che ciascuno dei loro $3 + 5 = 8$ archi è effettivamente un arco di G_2. (In effetti, se all'esame avevo a disposizione un colore, oppure in questo .pdf potevo ispessire il tratto, mi conveniva evidenziare questi stessi due cicli dentro la stessa figura in cui ho dato la 2-colorazione; i 2 archi rimossi avrei potuto offrirli in riga tratteggiata. In questo modo avrei evitato il passaggio di codifica attraverso i nomi dei nodi a carico di King Arthur, e soprattutto quello di decodifica a carico di Mago Merlino, che mi sono da interfaccia nel dialogo tra mè e mè e nel dialogo con l'altro. Sarebbe inoltre stata più evidente la complementarietà tra i due certificati spesi per l'acquisizione degli 1+1=2 punti. Ovviamente il modo di esibire e di comunicare dentro e fuori non è unico, ma spero passi il messaggio che la cura dei certificati offre via pragmatica che può portare lontano, in matematica e non solo.)

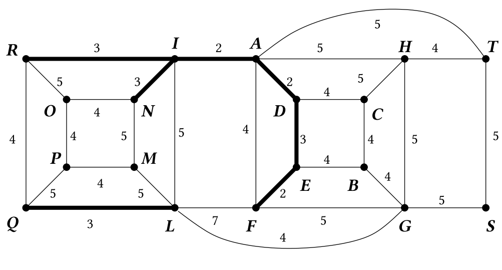
Minimum Spanning Trees.

L'arco DC, appartiene a degli MST ma non a tutti. Appartiene ad alcuni in quanto arco di peso minimo del taglio che separa la parte sinistra dalla parte destra della figura presa a riferimento (ossia quello che separa i nodi nodi Q, P, M, L, F, E, D, A, I, N, O, R dagli altri nodi e che consta degli archi AT, AH, DC, EB FG LG. Non appartiene a tutti in quanto arco di peso massimo del ciclo $DCBE$.

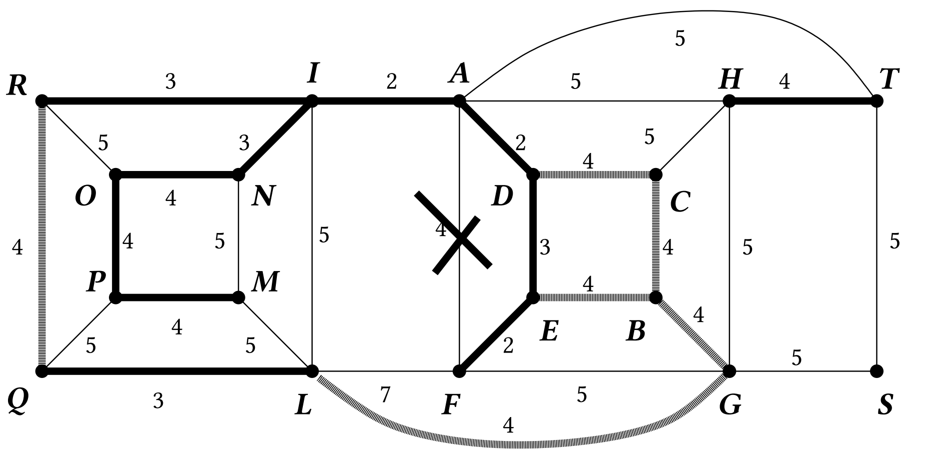
L'arco DE appartiene a tutti gli MST in quanto arco di peso strettamente minimo del taglio che separa la parte alta dalla parte bassa della figura presa a riferimento (ossia quello che separa i nodi nodi Q, P, M, L, F, E, B, G, S dagli altri nodi e che consta degli archi QR, PO, MN, LI FA, ED, BC, GH, ST.

L'arco QR non appartiene ad alcun MST in quanto arco di peso strettamente massimo del ciclo $QRIADEFL$.

Questa figura mostra come il sottografo costituito dai soli archi di peso al più 3 sia una foresta (aciclico). Questo significa che tutti questi archi devono necessariamente essere inclusi in ogni MST. E ogni arco che dovesse chiudere ciclo con essi (come l'arco AF) è necessariamente escluso.



Nella seguente figura ci siamo limitati a evidenziare (inspessendoli) i soli archi di peso al più 4.



,

Analizzando i cicli contenuti in questo sottografo degli archi di peso al più 4 possiamo comprendere diverse cose:

- dal ciclo $AFED$, dove l'arco AF è l'unico arco di peso massimo, potremmo concludere (in realtà lo avevamo già fatto) che l'arco AF non appartiene ad alcun MST e può pertanto essere rimosso (edge deletion) per meglio proseguire nella nostra analisi.
- tutti gli archi inspessiti che non formano alcun ciclo con altri archi inspessiti appartengono necessariamente ad ogni MST. Degli archi evidenziati, quelli che restano in bilico sono i 6 archi DC, CB, BE, BG, GL, e QR in linea tratteggiata.
- il sottografo degli archi di peso al più 4 presente 3 componenti connesse: $\{S\}$, $\{H, T\}$, e l'insieme di tutti gli altri nodi. Contraendo (edge contraction degli archi di peso al più 4) ciascuna di queste componenti connesse in un singolo nodo (potremmo chiamare S', H' e A' questi supernodi collassati) otteniamo un grafo sui 3 supernodi A', H' e S', con un solo arco di estremi S' e H', un solo arco di estremi S' e A', e 4 archi paralleli di estremi A' e H' (essi corrispondono agli archi HA, HC, HG e AT). Contiamo gli spanning tree di questo grafo (tutti MST dato che tutti gli archi hanno peso 5). Ce ne sono 4 che non usano l'arco $S'A'$ più 4 che non usano l'arco $S'H'$ più 1 solo non usa nessuno dei 4 archi tra A' e H'. Quindi l'MST del grafo originale ha $4 + 4 + 1 = 9$ diverse scelte su quali archi di peso 5 includere.

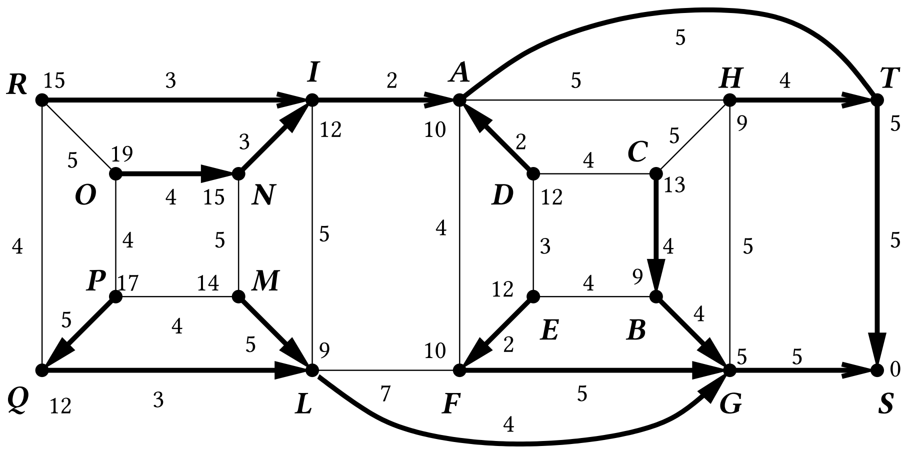
Per stabilire quante siano le scelte sugli archi di peso 4 potremmo allo stesso modo concentrarci sul conteggio degli spanning tree nel grafo ottenuto contraendo tutti gli archi di peso al più 3 e rimuovendo tutti gli archi di peso almeno 5. Ma in questo documento scelgo di evitarmi di costruire tale grafo limitandomi piuttosto a raccogliere gli elementi già accessibili dalla figura sopra.

Osservo che gli archi RQ, LG e BG sono in serie tra di loro e con il parallelo tra l'arco EB e la serie $BC - CD$. Se prendiamo tutti e 3 gli archi RQ, LG e BG allora, pena chiudere ciclo, non possiamo prendere l'arco EB e dobbiamo prendere uno solo tra gli archi BC e CD (uno dobbiamo prenderlo per non lasciare isolato il nodo C). Quindi le possibilità sono solo 2 (prendere BC oppure CD). Altrimenti abbiamo $3 \times 3 = 9$ modi, dove sono 3 modi di scegliere quale dei 3 archi RQ, LG e BG non includere e 3 sono i modi di scegliere quale dei 3 archi EB, BC e CD non includere. In totale abbiamo $2 + 9 = 11$ scelte sugli archi di peso 4.

In definitiva ci sono $11 \times 9 = 99$ alberi ricoprenti di perso minimo. Ciascuno di essi include tutti gli archi di peso al più 3, precisamente 2 archi di peso 5 e precisamente 4 archi di peso 4.

Cammini minimi.

Un'albero dei cammini minimi dal nodo S è rappresentato in figura, dove per ogni nodo si è avuto cura di riportare la distanza dal nodo S. 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 nessun nodo ha scelte alternative e quindi l'albero fornito in figura è l'unico albero dei cammini minimi.

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.

Riconoscimento di DAG.

Quando ogni arco uv è orientato da u a v come da tabella fornita in input all'inizio dell'esercizio, il grafo diretto è un Directed Acyclic Graph (DAG). Certificato: il seguente topological sort:

B, A, C, D, E, F, G, I, L, M, N, O, P, Q, R, H, S, T

A King Arthur basta verificare che, per ogni arco diretto (u, v), la coda u dell'arco precede la testa v nell'ordinamento proposto.

Esercizio 3 (con 7 richieste: 8+2+4+1+2+1+2 = 20 punti [programmazione dinamica]):
Il tempo è stato suddiviso in 30 slot di 1 minuto ciascuno. Per $i = 1, 2, ..., 30$, il trailer $r_i = (V_i, A_i, D_i)$ è caratterizzato da 3 valori:

V_i è il ***valore*** che saremmo disposti a pagare per poter vedere il trailer r_i

D_i è la ***durata*** del trailer r_i espressa in timeslots

A_i è il timeslot all'inizio del quale prende ***avvio*** lo streaming del trailer r_i

i	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	28	29	30
V	4	3	7	3	4	5	1	4	5	2	7	8	3	1	1	4	1	1	5	3	1	7	6	8	4	7	1	5	3	6
D	4	2	9	3	6	6	1	4	2	5	2	4	2	3	5	1	4	3	5	1	1	3	4	2	3	1	2	3	1	9
A	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	28	29	30

Un fatto importante, si noti, è che $A_i = i$ per ogni $i = 1, 2, ..., 30$. Si noti che un trailer r_i finisce per occupare i $D[i]$ timeslots nell'intervallo $[i, i + D[i])$. Ad esempio, il timeslot r_6 ha durata sei e quindi occupa i timeslots in $[6, 12) = [6, 11]$.

Purtroppo non posso visionare sia il trailer r_i che il trailer r_j se si sovrappongono anche solo per un singolo timeslot, ossia se $i < j$ e $A_j \leq A_i + (D_i - 1)$.

Decidi quali trailer visionare con l'obiettivo di massimizzarne il valore complessivo.

Richieste dell'Esercizio 3
3.1 (8 pt, pensa programmazione dinamica) Ti chiediamo di lavorare con metodo: di fronte al caso generale con n trailers, poniti prima il solo obiettivo di come determinare il valore ottimo, ossia di rispondere alla domanda di quale sia il massimo valore complessivo per un insieme di trailer tutti visualizzabili (penserai solo successivamente a come costruirlo). Inventati una famiglia di domande che includa la domanda di interesse come un suo membro particolare, e che sia chiusa rispetto ad induzione. Questi punti vengono attribuiti se prima (1) definisci la famiglia e poi (2) offri la regola uniforme per il computo dei sui membri. Per questa richiesta ti aggiudichi:

3 punti: se il numero di domande è esponenziale in n (una qualsiasi espressione ricorsiva, anche inefficiente)

4 punti: se il numero di domande è super-lineare in n

5 punti: se è lineare in n

≥ 6 punti: se è n o $n+1$, ma i punti salgono a ...

8 punti: se per rispondere ad una domanda ti basta sbirciare la risposta a due sola altre domande (per sotto-istanze più piccole).

3.2 (2 pt, valore ottimo) quale è il massimo valore complessivo ottenibile per il problema in tabella?

3.3 (4 pt, soluzione ottima) fornire una soluzione ottima (quali trailer vanno scelti?)

3.4 (1 pt, opt val -tardi 6) ti sei svegliato tardi, perdendoti i primi 5 timeslot; quale è il massimo valore complessivo cui ambire ora?

3.5 (2 pt, opt sol -tardi 6) fornire una soluzione di massimo valore quando non si possa includere alcuno dei primi 5 trailer

3.6 (1 pt, opt val -tardi 16) quale è il massimo valore complessivo per una soluzione che non includa nessuno dei primi 15 trailer?

3.7 (2 pt, opt sol -tardi 16) fornire una soluzione di massimo valore quando non si possa includere alcuno dei primi 15 trailer

svolgimento esercizio 3.

Richiesta 1. Per ogni $i = 1, 2, ..., 31$, ci inventiamo la seguente domanda/sottoproblema:

quale è il massimo valore complessivo Q_i che è possibile raccogliere se ci si sveglia tardi, al timeslot i, e resta pertanto possibile visionare solo quei trailers r_j con $j \geq i$?

Questa famiglia di $n+1$ domande include la domanda originale sotto le vesti di Q_1. Inoltre, come caso base, $Q_{31} = 0$ poichè a quel punto non resta più alcun trailer disponibile.

Quando ci si sveglia tardi al timeslot i, la prima scelta che abbiamo di fornte sarà quella se visionare o meno r_i; dobbiamo prendere la migliore tra queste due opzioni. Non è difficile convincersi che se si decide di visionare un trailer r_i con $A_i + D_i \geq 31$ allora r_i sarà necessariamente l'ultimo e quindi l'unico trailer visionato.

Pertanto, se $A_i + D_i \geq 31$ allora $Q_i = \max(V_i, Q_{i+1})$.

La ricorrenza non è poi molto più complicata nel caso in cui $A_i + D_i < 31$: basterà adeguare il valore della prima opzione considerando che dopo aver visionato il trailer r_i ci si ritroverà sostanzialmente nella situazione di essersi svegliati all'istante $A_i + D_i$, con la differenza di avere già intascato V_i.

Pertanto, se $A_i + D_i \leq 30$ allora $Q_i = \max\left(V_i + Q_{i+D_i}, Q_{i+1}\right)$.

Richieste 2-7. Compiliamo il vettore di programmazione dinamica Q progettato al punto precedente.

Ecco la tabella ricevuta in input dopo aver sostituito la riga contenete in vettore A (per altro identica con la prima riga con gli indici dei trailers) con la riga che offre il vettore Q di programmazione dinamica contenete i valori ottimi per ogni sotto-problema nella famiglia considerata:

i	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	28	29	30
V	4	3	7	3	4	5	1	4	5	2	7	8	3	1	1	4	1	1	5	3	1	7	6	8	4	7	1	5	3	6
D	4	2	9	3	6	6	1	4	2	5	2	4	2	3	5	1	4	3	5	1	1	3	4	2	3	1	2	3	1	9
Q	56	56	53	53	50	50	50	49	49	44	44	42	37	34	34	34	30	30	30	29	26	25	25	25	17	17	10	9	9	6

Richiesta 2. Il massimo valore complessivo ottenibile per il problema in input è $Q[1] = 56$.

Richiesta 3. Per ricostruire una soluzione ottima procedo a ritroso, ossia ora da sinistra verso destra (mentre il computo dei valori in Q è avvenuto da destra verso sinistra) affrontando ogni scelta con la forza delle promesse di Abramo contenute in Q.

Una soluzione ottima per il problema in input consiste nel visionare i trailers: `(2, 4, 7, 9, 11, 13, 16, 19, 24, 26, 27, 29, 30)`

Richiesta 4. Il massimo valore complessivo ottenibile qualora si siano persi i primi 5 timeslots e si sia in gioco solo a partire dal timeslot 6 è $Q[1] = 50$.

Richiesta 5. Una soluzione ottima in tale situazione consiste nel visionare i trailers: `(7, 9, 11, 13, 16, 19, 24, 26, 27, 29, 30)`

Richiesta 6. Il massimo valore complessivo ottenibile qualora si siano persi i primi 15 timeslots e si sia in gioco solo a partire dal timeslot 16 è $Q[1] = 34$.

Richiesta 7. Una soluzione ottima in tale situazione consiste nel visionare i trailers: `(16, 19, 24, 26, 27, 29, 30)`

Esercizio 4 (con 3 richieste: 5+2+3 = 10 punti [programmazione dinamica]):

\	A	B	C	D	E	F	G
1	0	1	3	2	1	2	2
2	1	*	2	1	*	1	1
3	1	1	*	2	1	*	1
4	2	1	2	2	1	2	1
5	2	1	1	1	2	*	1
6	1	*	1	*	1	1	1
7	1	2	2	2	1	1	0

Un *cammino* nella griglia 7 x 7 in figura è una sequenza di celle $C_1, C_2, \ldots, C_{13}$ tale che:

1. nessuna cella attraversata contiene una mina
2. per ogni $i = 1, \ldots, 7 + 7 - 2$, la cella C_{i+1} è la cella immediatamente sotto oppure all'immediata destra della cella C_i.

Il *premio* di un cammino è pari alla somma dei numeri riportati nelle celle che visita (incluse la prima e l'ultima).

Richieste dell'Esercizio 4

4.1 (5 pt, numero cammini) Stabilire il numero di cammini dalla cella A1 alla cella G7 (1 punto), dalla cella A1 alla cella E6 (1 punto), dalla cella C2 alla cella G7 (1 punto), dalla cella A1 alla cella G7 passanti per D5 (2 punti)
4.2 (2 pt, max val cammino) Calcolare il massimo premio di un cammino dalla cella A1 alla cella G7 (1 punto), dalla cella A1 alla cella E6 (1 punto)
4.3 (3 pt, cammino ottimo) Fornire un cammino di massimo premio tra quelli dalla cella A1 alla cella G7 (2 punti), dalla cella A1 alla cella E6 (1 punto)
svolgimento esercizio 4.
Richiesta 1. Per rispondere alla prima richiesta conviene compilare prima un paio di tabelle di programmazione dinamica.

tabella 1 (`num_paths_to`): per $i \in \{0, ..., 7\}$ e $X \in \{z, A, B, ..., G\}$ la cella "Xi" riporta il numero di cammini che nel campo minato vanno dalla cella $A1$ alla cella Xi. Chiaramente, se $i = 0$ o $X = z$, ma non entrambi, il numero di tali cammini è zero in quanto a un cammino non è mai concesso di salire o di portarsi a sinistra. Se invece $i = 0$ e $X = z$, allora il valore di risposta alla domanda è 1, e farà da big-bang per la generazione dei valori in questa tabella. Queste celle sentinella si prestano quindi molto bene come casi base dell'induzione in quanto possono essere riempite con regola uniforme (settate tutte a 0). Un'altra regola uniforme si occuperà di riempire tutte le rimanenti celle (nelle posizioni di mina si colloca uno 0, nelle altre la somma del valore sopra e del valore a sinistra).

\	z	A	B	C	D	E	F	G
0	0	0	0	0	0	0	0	0
1	0	1	1	1	1	1	1	1
2	0	1	0	1	2	0	1	2
3	0	1	1	0	2	2	0	2
4	0	1	2	2	4	6	6	8
5	0	1	3	5	9	15	0	8
6	0	1	0	5	0	15	15	23
7	0	1	1	6	6	21	36	59

tabella 2 (`num_paths_from`): per $i \in \{1, ..., 8\}$ e $X \in \{A, B, ..., G, H\}$ la cella "Xi" riporta il numero di cammini che nel campo minato portano da "Xi" alla cella $G7$. Chiaramente, se $i = 8$ o $x = H$, ma non entrambi, il numero di tali cammini è zero in quanto a un cammino non è mai concesso di salire o di portarsi a sinistra. Se invece $i = 8$ e $x = H$, allora il valore di risposta alla domanda è 1, e farà da big-bang per la generazione dei valori in questa tabella. Queste celle sentinella si prestano quindi molto bene come casi base dell'induzione in quanto possono essere riempite con regola uniforme (settate tutte a 0). Un'altra regola uniforme si occuperà di riempire tutte le rimanenti celle (nelle posizioni di mina si colloca uno 0, nelle altre la somma del valore sotto e del valore a destra).

\	A	B	C	D	E	F	G	H
1	59	24	24	13	2	2	1	0
2	35	0	11	11	0	1	1	0
3	35	15	0	11	4	0	1	0
4	20	15	11	7	4	1	1	0
5	5	4	4	3	3	0	1	0
6	1	0	1	0	3	2	1	0
7	1	1	1	1	1	1	1	0
8	0	0	0	0	0	0	0	0

Come si vede le due tabelle concordano nel dire che il numero di cammini dalla cella A1 alla cella G7 è 59, ciò vale a verifica parziale dei valori nelle due tabelle.

(1 pt) il numero di cammini dalla cella A1 alla cella G7 è 59 (ultimo valore calcolato in una qualsiasi delle due tabelle)

(1 pt) il numero di cammini dalla cella A1 alla cella E6 è num_paths_to[E6] = 15

(1 pt) il numero di cammini dalla cella C2 alla cella G7 è num_paths_from[C2] = 11

(2 pt) il numero di cammini dalla cella A1 alla cella G7 passanti per D5 è num_paths_to[D5] x num_paths_from[D5] = 9 x 3 = 27.

Per rispondere alle richieste relative ai cammini di massimo premio, riempiamo un'ultima tabella di programmazione dinamica:

tabella 3 (`max_val_paths_to`): per $i \in \{0, ..., 7\}$ e $X \in \{z, A, B, ..., G\}$ la cella "Xi" riporta il massimo premio di un cammino dalla cella $A1$ alla cella Xi. Se $i = 0$ o $X = z$ l'insieme di tali cammini è vuoto (e quindi il massimo premio è $-\infty$) eccetto che per la cella A1 dove il premio dell'unico cammino è 0 essendo nullo il valore di tale cella. Terremo queste celle come casi base dell'induzione per le stesse ragioni di cui sopra. La regola uniforme che riempie tutte le rimanenti celle colloca $-\infty$ nelle celle di mina, mentre nelle altre celle pone la somma del premio della cella e del massimo tra il valore subito sopra e il valore subito a sinistra).

\	z	A	B	C	D	E	F	G
0	-∞	-∞	-∞	-∞	-∞	-∞	-∞	-∞
1	-∞	0	1	4	6	7	9	11
2	-∞	1	-∞	6	7	-∞	10	12
3	-∞	2	3	-∞	9	10	-∞	13
4	-∞	4	5	7	11	12	14	15
5	-∞	6	7	8	12	14	-∞	16
6	-∞	7	-∞	9	-∞	15	16	17
7	-∞	8	10	12	14	16	17	17

Di nuovo, ora è solo una questione di saper leggere i valori nella tabella:

(1 pt) il massimo premio di un cammino dalla cella A1 alla cella G7 è max_val_paths_to[G7] = 17

(1 pt) il massimo premio di un cammino dalla cella A1 alla cella E6 è max_val_paths_to[E6] = 15

Per ricostruire dei cammini ottimi procedo a ritroso affrontando ogni scelta con la forza delle promesse di Abramo contenute nella tavola 3.

(2 pt) un cammino di massimo premio dalla cella A1 alla cella G7 è A1-B1-C1-C2-D2-D3-D4-D5-E5-E6-E7-F7-G7

(1 pt) un cammino di massimo premio dalla cella A1 alla cella E6 è A1-B1-C1-C2-D2-D3-D4-D5-E5-E6-E7-F7-G7 (questa è discesa facile dalla precedente perchè prefisso di cammino ottimo è ottimo, proprietà tipica per un problema che si presta ad essere risolto tramite la programmazione dinamica).

La programmazione dinamica è per altro una tecnica abbastanza robusta, potrebbe ad esempio risolverci anche questioni del tipo: quanti sono i cammini ottimi? Quanti sono i cammini ammissibili se ad un cammino è consentito visitare al più tre mine?

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?).