Problem Set 3

Algoritmi e Strutture Dati – a.a. 2024/2025

Università di Roma "Tor Vergata"

Prof. Luciano Gualà

Importanti avvertenze preliminari:

1. Il gruppo deve essere formato da almeno tre persone!
2. Per lo svolgimento si consiglia fortemente l'utilizzo del template LaTeX disponibile nella pagina web del corso.

Linee guida per la consegna. Gli elaborati dovranno essere consegnati entro domenica 01/06/2025 alle ore 23:59. Si prega di seguire le seguenti indicazioni:

1. le soluzioni dovranno essere consegnate via email a entrambi gli indirizzi `guala@mat.uniroma2.it` e `alessandrostr95@gmail.com`;
2. l'oggetto dell'email dovrà essere "`Consegna problem set 3 - 2024/2025`";
3. il file dovrà essere in formato pdf;
4. il nome del file dovrà essere "`PS_3_XXX.pdf`" dove `XXX` è il primo cognome (in lower case) in ordine alfabetico dei membri del gruppo;
5. l'email dovrà essere inviata tramite l'indirizzo dello studente `XXX` presente nel nome del file;
6. inserire nel file nome, cognome, indirizzo email e matricola di tutti i membri del gruppo (meglio ancora se tutti i membri del gruppo sono in indirizzo nell'email della consegna).

Buon lavoro 🔥 😎!

Problema 1 *(Algoritmista giardiniere)*
Durante le vacanze ti sei dato al giardinaggio, ed hai deciso di iniziare sistemando il tuo giardino, che oramai è in condizioni pietose. Hai un grosso e vecchio albero al centro del tuo giardino che vorresti rimuovere. Alessandro, il dottorando che fa il dottorato di ricerca sotto la tua supervisione, si è detto disponibile a portare via il legno con la sua macchina che può però trasportare un massimo di $W \in \mathbb{N}$ chilogrammi per viaggio.

Purtroppo quel vecchio albero è troppo pesante per essere trasportato in una sola volta, e quindi devi tagliarlo in pezzi più piccoli e leggeri. Ora, del numero di viaggi che dovrà fare Alessandro con la sua macchina non ti interessa, però tagliare l'albero è faticoso e vorresti effettuare il minimo numero di tagli.

Dato che sei un informatico, modelli il problema nel seguente modo. Rappresenti il tuo albero in giardino come un albero radicato e pesato $T = (V, E, w : E \to \mathbb{N}^+)$ di n nodi, dove il peso $w(e)$ associato all'arco e è il peso effettivo del ramo corrispondente.

Tu puoi effettuare un *taglio* su un nodo interno v dell'albero, ottenendo $c(v)+1$ nuovi sottoalberi $T_0, T_1, \ldots, T_{c(v)}$, dove $c(v)$ è il numero di figli $u_1, \ldots, u_{c(v)}$ di v. In particolare, T_0 è l'albero ottenuto rimuovendo da T tutti i sottoalberi radicati nei figli di v, mentre T_i è il sottoalbero che ha v come radice e include tutti i discendenti di u_i, per ogni $i = 1, \ldots, c(v)$. Si veda la figura per un esempio.

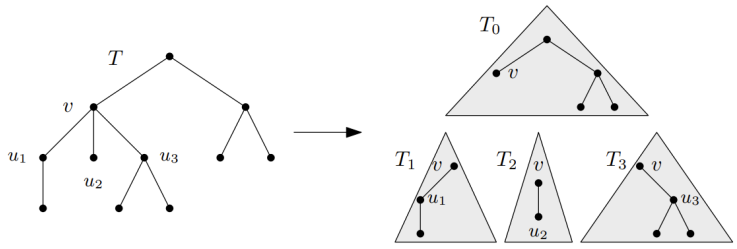

Figure 1: Esempio di taglio sul nodo v.

Vuoi quindi progettare un algoritmo che calcoli il minimo numero di tagli che devi effettuare su T, in modo tale che ogni sottoalbero della foresta ottenuta abbia peso al più W. Ci pensi un po' e ti sembra che possa funzionare un algoritmo *greedy* che probabilmente ha complessità lineare $O(n)$. Ma non ne sei sicuro. Quindi decidi di mettere l'esercizio nel prossimo Problem Set sperando che uno dei tuoi studenti risolvano per te il problema.

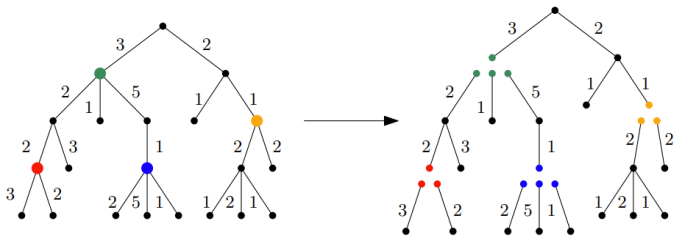

Figure 2: Esempio di soluzione (ottima?) per $W = 10$ che esegue 4 tagli.

Problema 2 *(Algo Run* `v2.0`*)*
Lo sviluppo di Algo Run è proseguito dal suo rilascio (vedi Problem Set 1 – a.a. 2024/2025, esercizio 3), ed è stata appena rilasciata la versione `2.0`. In questa versione, vengono implementate nuove dinamiche di gioco, che lo rendono più divertente (alcune recensioni dicono che la prima versione era troppo facile).

Ecco le note della patch rilasciata dagli sviluppatori del gioco:

- mentre nella versione `1.0` il protagonista poteva saltare da una corsia ad un'altra qualsiasi, in questa versione sarà possibile spostarsi solamente sulle corsie adiacenti a quella nella quale il giocatore si trova;
- il numero di corsie è stato ridotto da un valore generico k a 3 sole corsie;
- oltre i vari cfu coins, per la mappa sono sparsi anche degli *ostacoli*, i quali andranno evitati altrimenti si perde la partita;
- il giocatore ha anche l'opportunità di *saltare* in avanti di 2 caselle, evitando qualsiasi ostacolo (o moneta) immediatamente di fronte. Il giocatore potrà però saltare solamente δ volte per partita.

Questa volta la mappa di gioco si presenta come una griglia M di dimensione $n \times 3$, dove nella cella (i, j) è presente una X se in quel punto della mappa c'è un ostacolo, un intero positivo se in quella cella c'è un cfu coin con di valore $M[i, j]$, oppure niente.

Perciò, dati in input una mappa M di dimensione $n \times 3$ e un valore $\delta \geq 0$, per vincere il livello dovrete calcolare qual è il massimo numero di cfu coin che potrete raccogliere, sapendo che il personaggio può iniziare in una qualsiasi delle posizioni della prima riga (a vostra scelta).

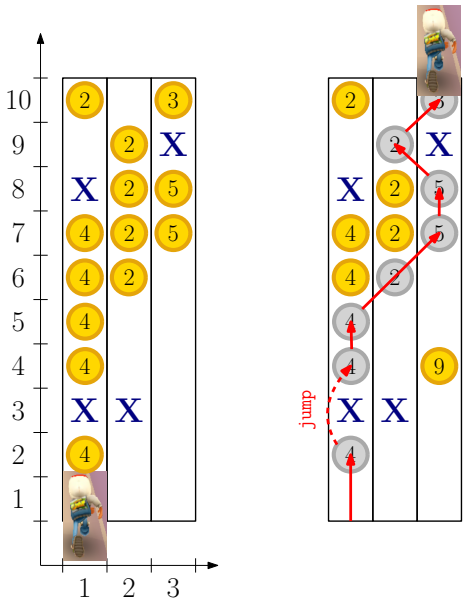

Figure 3: Soluzione ottima per un livello di Algo Run `v2.0`, con $\delta = 1$.

Problema 3 *(Il compagno copione delle superiori)*
L'altro giorno hai incontrato un tuo vecchio compagno delle superiori che copiava sempre gli scritti di matematica da te. Hai scoperto che si è iscritto anche lui a Informatica, ma in un altro ateneo di cui, per riservatezza, non faremo il nome.

Dopo i convenevoli di rito lui ti racconta che sta seguendo un corso difficile che ha a che fare con gli algoritmi e di cui ha già superato un esonero copiando dal suo compagno di banco.[1] Ora, però, per superare la seconda parte del corso, deve risolve un esercizio che lo sta facendo uscire pazzo. Deve capire se esiste un algoritmo veloce per il seguente problema:

Riconfigurazione di pedine: Ti è dato un grafo non orientato $G = (V, E)$ di n nodi e due nodi speciali s e t. Alcuni nodi di G inizialmente contengono delle *pedine*. Puoi *riconfigurare* le pedine come vuoi con il vincolo che ogni pedina può muoversi di al più 1, ovvero, o la lasci nella posizione iniziale v, o la muovi in un vertice adiacente a v. Il problema è capire se è possibile riconfigurare le pedine in modo tale che s e t siano "separati" dalle pedine, cioè, in modo tale che ogni cammino in G da s a t attraversi almeno un nodo che nella configurazione finale contiene una pedina. Un esempio di istanza e riconfigurazione è data in figura.

Il tuo amico è convinto che ci sia un algoritmo di complessità $O(n^4)$ che risolve il problema.[2] Tu però hai il forte dubbio che la cosa sia improbabile

[1] In questo ateneo ci sono talmente tanti studenti, che le classi sono enormi e si copia alla grande durante gli scritti, ti ha detto lui ridendo soddisfatto.

[2] Sembra che abbia sentito dire così dallo stesso collega da cui ha copiato l'esonero.

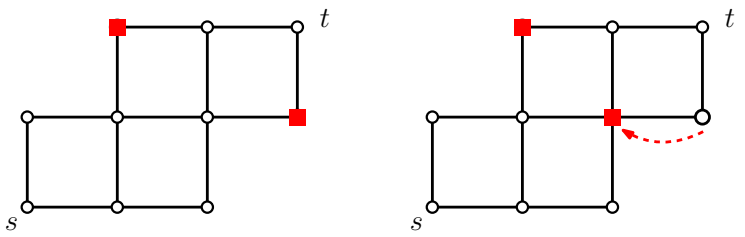

Figure 4: Istanza di esempio in cui è possibile riconfigurare le pedine in modo da separare s da t.

perché sospetti che il problema sia NP-completo. Ti va di onorare ancora la vecchia tradizione e fornire al tuo ex-compagno di scuola una riduzione da 3-SAT?