Algoritmi e Strutture Dati (modulo II)

Testo della prova scritta del 23 giugno 2017

docente: Gualà

Cognome:...................................... Nome:... Matr.:........................

Esercizio 1 Alice sta crescendo e vuole vedere il mondo, che è modellato come un grafo non orientato $G = (V, E)$ di n nodi e m archi, dove ogni nodo rappresenta un posto bellissimo e ogni arco $(u, v) \in E$ indica la possibilità di spostarsi da u a v. Alice si trova sul nodo s e vorrebbe visitare tutti i restanti $n - 1$ posti meravigliosi. Purtroppo, però, non tutti gli archi sono percorribili a tutte le età. Ad ogni arco $e \in E$ quindi è associato un valore $w(e)$ che indica l'età minima necessaria per attraversare l'arco. Diremo che un posto v è raggiungibile da Alice all'età x se esiste un cammino da s a v composto di soli archi di peso minore o uguale a x. Progettate un algoritmo che calcoli l'età minima che consente ad Alice di vedere il mondo intero, ovvero la più piccola età x per cui tutti i posti sono raggiungibili da Alice all'età x.

Nota: Il punteggio ottenuto in questo esercizio dipende anche dall'efficienza della soluzione proposta. Una soluzione con complessità temporale $O(m \log n)$ da diritto a un punteggio pieno, ma anche soluzioni più inefficienti saranno valutate.

Esercizio 2 Siano $G_1, \ldots, G_n$ n grafi diretti ognuno con n vertici, $\Theta(n^2)$ archi, e con capacità intere associate agli archi. Per ogni $i \in \{1, \ldots, n-1\}$, G_i è collegato a G_{i+1} con un arco diretto (t_i, s_{i+1}) che ha capacità x (quindi $t_i \in G_i$ e $s_{i+1} \in G_{i+1}$ e non esistono altri archi che vanno da G_i a G_{i+1}). Sia s_1 un nodo di G_1 e t_n un nodo di G_n. Supponendo che $x = 2^n$, progettare un algoritmo più efficiente possibile che trova il flusso massimo da s_1 a t_n nel grafo risultante dall'unione di tutti i G_i più gli archi tra vanno da un grafo all'altro.

Si riporta di seguito una tabella con i tre algoritmi per il maxflow presentati a lezione e le relative complessità, dove E indica il numero di archi e V il numero di nodi del grafo su cui si applica l'algoritmo e C è una delimitazione superiore (upper bound) al massimo flusso:

Ford–Fulkerson (scegliendo un cammino aumentante qualsiasi)	$C \cdot E$
Δ-scaling	$E^2 \cdot \log C$
Edmonds–Karp (F-F scegliendo un cammino aumentante di lunghezza minima)	$E^2 \cdot V$