Progettazione di Algoritmi
Anno Accademico 2015/2016
Appello del 16/2/2016

Cognome e Nome:
Numero di Matricola:

Spazio riservato alla correzione

1	2	3	4	Bonus	Totale
/25	/25	/25	/25		/100

1. Grafi

a) Si scriva lo pseudocodice dell'algoritmo BFS che utilizza un array Discovered e un array L, come illustrato nel libro di testo, e si analizzi il tempo di esecuzione dell'algoritmo proposto. Analizzare il tempo di esecuzione significa fornire un limite superiore asintotico quanto migliore e` possibile al tempo di esecuzione dell'algoritmo giustificando la risposta.

b) Descrivere un algoritmo che, dato in input un grafo non direzionato G, scopre se il grafo contiene o meno cicli e in caso affermativo restituisce in output uno dei cicli.
L'algoritmo deve avere tempo di esecuzione nel caso pessimo O(n+m).

c) **Bonus.** Si dica se l'algoritmo descritto al punto precedente funziona anche nel caso di un grafo direzionato. Se pensate che l'algoritmo funzioni anche in questo caso, dare una breve spiegazione del perché ciò è vero; altrimenti fornire un controesempio che dimostri che l'algoritmo non funziona con i grafi direzionati.

2. Algoritmi greedy

   a) Una persona vuole andare dalla citta` 1 alla citta` n, con n>1, seguendo un itinerario che passa attraverso le citta` 2,3,…,n-1 numerate in ordine di distanza dalla citta` 1. In ognuna delle n citta` c'e` una stazione di benzina alla quale la persona puo` fermarsi per fare rifornimento di carburante. La persona conosce il numero chilometri k che l'auto puo` percorrere con un pieno e la distanza tra le pompe di benzina di due citta` adiacenti per cui sa fino a dove puo` arrivare se fa il pieno in una certa citta`. Si descriva un algoritmo che minimizza il numero di rifornimenti. L'auto ha inizialmente il serbatoio completamente vuoto per cui l'autista deve fare un primo rifornimento nella citta` 1.
   NB: si pensi alla strategia piu` ovvia, quella che suggerisce il buon senso.

b) Si dimostri che la strategia greedy del vostro algoritmo minimizza il numero di rifornimenti. Suggerimento: Si supponga per assurdo che la strategia greedy non sia ottima. Siano $g_1,\ldots,g_m$ le stazioni in cui si ferma l'autista se applica la strategia greedy e siano $c_1,\ldots,c_q$ le stazioni in cui si ferma l'autista nel caso in cui applica la strategia ottima. Ovviamente si ha m> q.

1. Si mostri che a partire da sinistra verso destra, $c_1,\ldots,c_q$ puo` essere trasformata nella soluzione $g_1,\ldots,g_q$ , cioe` in una sequenza costituita dallo stesso numero di fermate della soluzione ottima e tale che la distanza da g_q ad n e` minore o uguale di k.
2. Si spieghi perche` cio` che e` stato dimostrato al punto 1 porta ad un assurdo.

3. Programmazione dinamica.
   a) Formulare in modo chiaro il problema Segmented Least Squares

b) Fornire la formula per il calcolo di OPT(j) spiegando in modo chiaro come si arriva a quella formula e cosa rappresentano le quantita` che compaiono in essa.

c) Scrivere lo pseudocodice dell' algoritmo che calcola la soluzione ottima per il probema Segmented Least Squares. Analizzare il tempo di esecuzione dell'algoritmo nel caso pessimo. Analizzare il tempo di esecuzione significa fornire un limite superiore asintotico quanto migliore e` possibile al tempo di esecuzione dell'algoritmo giustificando la risposta.

4. **Applicazioni del problema del massimo flusso**
   a) Formulare in modo chiaro il problema del massimo matching bipartito

b) Formulare il problema del massimo matching bipartito come un problema di massimo flusso.

c) Descrivere un algoritmo che computa il massimo matching bipartito. Occorre spiegare in modo chiaro anche da quali archi e` formato il massimo matching bipartito. Analizzare il tempo di esecuzione dell'algoritmo. Analizzare il tempo di esecuzione significa fornire un limite superiore asintotico quanto migliore e` possibile al tempo di esecuzione dell'algoritmo giustificando la risposta.

d) Mostrare i passi dell'algoritmo di cui al punto precedente sul seguente grafo. Occorre mostrare l'esecuzione di tutto l'algoritmo e mostrare l'output prodotto dall'algoritmo per il grafo considerato.