Esame di algoritmi e strutture dati

5 febbraio 2021

Tempo a disposizione: 2 ore

Esercizio 1

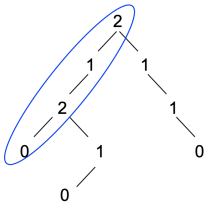
(7 punti)

- Progettare un algoritmo (pseudocodice) con segnatura

$$profonditaSx(Albero\ T) \rightarrow\ Intero,$$

che, preso in input un albero binario T, restituisca il numero di nodi contenuti nel cammino radice-foglia ottenuto partendo dalla radice e prendendo come successore di ciascun nodo il suo figlio sinistro.

Ad esempio, con l'albero sotto riportato, l'algoritmo deve restituire il valore 4:



- Dimostrare la complessià temporale dell'algoritmo.

Esercizio 2

(7 punti)

- Definire la nozione di struttura dati *heap*, indicando quale sia la relazione tra il numero di nodi contenuti in un heap e la sua altezza;
- Illustrare la procedura $heapify(AlberoBinario\ T)$ che, dato un albero binario T completo (almeno) fino al penultimo livello, modifica T in maniera tale da renderlo un heap;
- Illustrare la complessità temporale dell'algoritmo.

Esercizio 3

(7 punti)

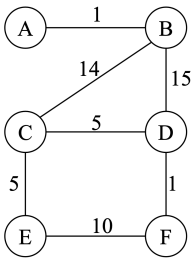
- Si descriva, mediante un opportuno esempio, la rappresentazione del tipo di dato Albero con lista dei figli;
- Si descriva l'algoritmo di visita in profondità di un albero e se ne discuta la complessità nel caso della rappresentazione sopra descritta.

(NOTA: il numero dei figli di ciascun nodo non è noto a priori)

Esercizio 4

(6 punti)

Si consideri il grafo in figura:



1. Illustrare un algoritmo per la costruzione di un albero dei cammini minimi che sia applicabile al grafo dato.
2. Mostrare i passi d'esecuzione dell'algoritmo sul grafo, prendendo A come nodo sorgente.

Esercizio 5

(6 punti)

Date le funzioni $f(n) = n^3 + 2$ e $g(n) = 2n^3 + 2n + 1$, si dimostri che:

- $f(n) = \Omega(g(n))$;
- $g(n) = \Omega(f(n))$;
- $f(n) = \Theta(g(n))$.