Università degli Studi di Roma Tre - Facoltà di Ingegneria C. S. Ing. Informatica

Algoritmi e Strutture Dati – (Fondamenti di Informatica 2, II modulo)

Prova del 27 settembre 2006

Indicare su tutti i fogli in alto a destra, con chiarezza, cognome e nome. Tempo a disposizione: 3 ore.
REGOLE:
Niente libri, quaderni o appunti: solo documento di riconoscimento, matita, penna gomma, etc.../ I fogli verranno distribuiti successivamente al testo./ Se ci si ritira non si può portare fuori il foglio con il testo./ Non si può uscire durante la prova./ Cellulari spenti.

Esercizio n.1 (4 punti) Descrivere, senza scrivere la pseudocodifica, l'algoritmo di ordinamento *quicksort* e dire qual è la sua complessità asintotica nei casi migliore, medio e peggiore, giustificando le affermazioni.

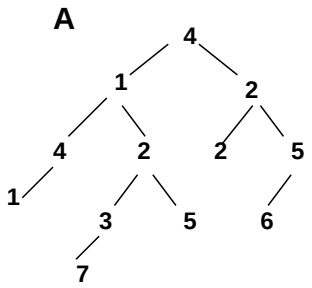
Esercizio n. 2 (4 punti) Illustrare la strategia *greedy*, discutendo in particolare a quali classi di problemi può essere applicata e il tipo di soluzione che può essere trovata. Descrivere come viene applicata la strategia greedy nel caso del problema della bisaccia.

Esercizio n. 3 (4 punti) Illustrare le proprietà degli ABR e riportare la specifica delle relative operazioni di base. Descrivere in dettaglio l'operazione di cancellazione di un elemento da un ABR e calcolarne la complessità asintotica.

Esercizio n. 4 A partire dalla rappresentazione di un albero bianrio etichettato con interi compresi tra 1 e n e rappresentato mediante record e puntatori, si vuole costruire un grafo **non orientato** rappresentato mediante liste di adiacenza.

4.1 (11 punti) Scrivere a tal scopo un algoritmo che preso in input un albero binario A, restituisca il grafo non orientato G.

Si ipotizzi che nell'albero ci possano essere degli archi ripetuti (vedi figura)



che non devono essere tenuti in considerazione nella rappresentazione mediante liste di adiacenza (un nodo non può avere due successori con la stessa etichetta).

Il grafo non orientato deve essere rappresentato mediante liste di adiacenza.

Spiegare chiaramente l'algoritmo utilizzato e commentare la pseudocodifica.

4.2 (4 punti) Calcolare la complessità asintotica dell'algoritmo scritto al punto 4.1.

4.3 (5 punti) Codificare l'algoritmo nel linguaggio `C`, senza codificare le funzioni di acquisizione e stampa dei dati. Iniziare a codificare l'algoritmo e successivamente le eventuali funzioni di supporto.

Si ricordano, le definizioni dei teoremi da applicare, eventualmente, per il calcolo della complessità:

Primo teorema per risolvere le equazioni di ricorrenza:
Se

$$T(n) = \begin{cases} a & \text{se } n = 0 \\ T(n-1) + g(n) & \text{se } n > 0 \end{cases}$$

allora:

$$T(n) = a + \sum_{k=1}^{n} g(k)$$

Master Theorem: Sia $p(n^k)$ un polinomio di grado k. Se

$$T(n) = \begin{cases} c_0 & \text{se } n = 0 \\ a\,T(n/b) + p(n^k) & \text{se } n > 0 \end{cases}$$

per $a, b \geq 1$ e $p(n^k)$ un polinomio di grado k.

Allora:

caso 1: $T(n) = \Theta(n^{log_b a})$ se $a > b^k$.
caso 2: $T(n) = \Theta(n^k lg\, n)$ se $a = b^k$.
caso 3: $T(n) = \Theta(n^k)$ se $a < b^k$.