

Algoritmi e Strutture Dati - 15 Luglio 2015

Cognome Nome Matricola

1. La leggibilità è un prerequisito: parti difficili da leggere potranno essere ignorate.
2. Quando si presenta un algoritmo è fondamentale spiegare l'idea sottostante il suo funzionamento e motivarne la correttezza.

Domande

Domanda A (5 punti) Mostrare che la ricorrenza $T(n) = T(n/2) + T(n/4) + n$ ammette soluzione $T(n) = \Theta(n)$ utilizzando il metodo di sostituzione.

Domanda B (5 punti) Dare la definizione di B-albero. Qual è la minima altezza di un B-albero con grado minimo t contenente n chiavi? Motivare le risposte.

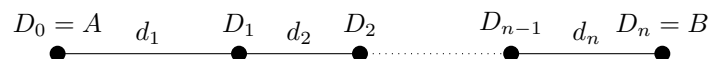
Domanda C (5 punti) Scrivere una funzione $blackHeight(T)$ che dato in input un albero binario di ricerca T , i cui nodi x hanno, oltre ai campi $x.key$, $x.left$ e $x.right$, hanno un campo $x.col$ che può essere B (per “black”) oppure R (per “red”), verifica se per ogni nodo, il cammino da quel nodo a qualsiasi foglia contiene lo stesso numero di nodi neri (altezza nera). In caso negativo, restituisce -1 , altrimenti restituisce l'altezza nera della radice.

Esercizi

Esercizio 1 (9 punti) Dato un array di interi $A[1..n]$, chiamiamo *gap* un indice $i \in [1, n)$ tale che $A[i+1] - A[i] > 1$.

- i. Mostrare per induzione su n che un array $A[1..n]$ tale che $A[n] - A[1] > n$ (quindi $n \geq 2$) contiene almeno un gap.
- ii. Fornire lo pseudocodice di una procedura ricorsiva divide et impera *gap* che dato un array $A[1..n]$ tale che $A[n] - A[1] > n$ restituisce un gap in A .
- iii. Valutare la complessità della funzione, utilizzando il master theorem.

Esercizio 2 (9 punti) Si supponga di voler viaggiare dalla città A alla città B con un'auto che ha un'autonomia pari a d km. Lungo il percorso si trovano $n - 1$ distributori D_1, \dots, D_{n-1} , a distanze di d_1, \dots, d_n km ($d_i \leq d$) come indicato in figura



L'auto ha inizialmente il serbatoio pieno e l'obiettivo è quello di percorrere il viaggio da A a B , minimizzando il numero di soste ai distributori per il rifornimento.

- i. Introdurre la nozione di soluzione per il problema e di costo della una soluzione. Mostrare che vale la proprietà della sottostruttura ottima e individuare una scelta che gode della proprietà della scelta greedy.
- ii. Sulla base della scelta greedy individuata al passo precedente, fornire un algoritmo greedy `stop(d,n)` che dato in input l'array delle distanze `d[1..n]` restituisce una soluzione ottima.
- iii. Valutare la complessità dell'algoritmo.