**UNIVERSITATEA DIN BUCURESTI**

**STUDENT: VLASCENCO DANIEL-MARIAN**

**GRUPA: 143 – ANUL I**

**Tema 1**

**Curs Structuri de Date**

**1)** Demonstrati ca orice algoritm care construieste un arbore binar de cautare cu n numere ruleaza in timp Ω(nlogn).

**Demonstratie:**

Un algoritm de creare a unui arbore binar de cautare cu n numere este echivalent cu un algoritm de sortare al unui vector cu n elemente.

Astfel problema se rezuma la a demonstra ca orice algoritm de sortare are timp de rulare Ω(nlogn). Stim ca un astfel de algoritm poate fi vazut sub forma unui arbore de decizie.

In cel mai nefavorabil caz, numarul comparatiilor realizate de arborele de decizie este egal cu inaltimea arborelui (h). Un arbore binar cu inaltimea h nu poate avea mai mult de frunze.

Prin aproximarea lui Stirling ( ) .

**2)** Demonstrati ca daca f(n) = Θ(g(n)) si g(n) = Θ(h(n)) atunci f(n) = Θ(h(n)).

**Demonstratie:**

Din f(n) = Θ(g(n)) avem:

Din g(n) = Θ(h(n)) avem:

Vrem sa aratam ca:

Din primele doua relatii rezulta:

**3)** Demonstrati ca

**Demonstratie:**

Vom considera, pentru simplicitate, logaritmul natural:

Problema se reduce la urmatoarea inecuatie:

Notam cu si incercam sa-i aflam punctele de extrem. Calculam astfel derivata: pe care o egalam cu 0 si obtinem: . A se nota ca

Pentru a calcula mai usor monotonia functiei f, vom considera c = 1.

Se observa ca:

, deci f e descrescatoare pe intervalul ( ] .

, deci f e crescatoare pe intervalul ( ) .

Astfel, rezulta ca punctul este punct de minim, asadar pentru orice n > .

Alegem, deci , de unde concluzia.

**4)** Se da un sir cu n numere de la 1 la n, cu exceptia unui numar care apare de 2 ori. Determinati numarul care apare de doua ori.

**Solutie:**

Consideram ca elementele sunt citite intr-un vector indexat de la 1(prima pozitie e v[1]), de asemenea, functia abs(x) reprezinta modulul numarul x.

pentru i = 1, n

daca v[abs(v[i])] < 0

cout<< v[i] << “se repeta”;

break;

altfel

v[i] = v[i] \* (-1)

In scrierea acestui algoritm este folosita o singura structura repetitiva (exceptant citirea), complexitatea algoritmului este O(n).

**5)** Fie X[1 :: n] si Y [1 :: n] doi vectori, ﬁecare continand n numere sortate. Prezentati un algoritm care sa gaseasca mediana celor 2n elemente. Mediana unei multimi de n elemente este elementul de pe pozitia [n/2] in sirul sortat. De exemplu, mediana multimii 3,1,7,6,4,9 este 4.

**Solutie:**

* Calculam medianele celor doi vectori (pentru X) si (pentru Y).
* Daca si sunt egale, atunci mediana celor doi vectori este chiar .
* Daca > , atunci mediana celor doi vectori este prezenta in urmatoarele intervale:
* X[ 1 :: ]
* Y[ :: n]
* Daca > , atunci mediana celor doi vectori este prezenta in urmatoarele intervale:
* X[ :: n]
* Y[ 1 :: ]
* Aplicam recursiv metoda explicata mai sus pana cand lungimea vectorilor devine 2, caz in care afisam mediana

Se poate observa ca la fiecare recursie se “elimina” n/2 elemente din X, respectiv elemente din Y. Avand in vedere ca timpul folosit de gasirea medianelor si este constant(O(1) ), timpul de care se foloseste algoritmul poate fi scris sub forma:

T[n] = T[n/2] + 1.

Folosind Teorema Master se poate observa ca:

a = 1, b = 2, f(n) = 1, asadar:

**6)** Sa presupunem urmatoarele. Ati castigat la loterie si v-ati cumparat o vila pe care doriti sa o mobilati. Deoarece Ferrari-ul dvs. are capacitate limitata, doriti sa faceti cat mai putine drumuri de la magazin la vila. Mai exact, Ferrari-ul are capacitate n, iar dumneavoastra aveti de cumparat k bunuri de dimensiune

Fie urmatorul algoritm greedy. Parcurgem bunurile in ordinea 1,2,...k si incercam sa le punem in masina. In momentul in care un bun nu mai incape in masina, efecutam un transport si continuam algoritmul.

1. Demonstarti ca algoritmul prezentat mai sus nu este optim. (0.5 puncte)

**Demonstratie:**

Presupunem ca algoritmul greedy descris mai sus este optim. Acest lucru presupune efectuarea unui numar minim de drumuri. Consideram urmatorul caz:

Capacitate = 20kg.

k = 3 (bunuri)

Conform algoritmului descris, sunt parcurse 3 drumuri ( Pune obiectul de 5 kg in masina, observa ca cel de-al doilea obiect nu incape si face primul drum. Pune apoi obiectul de 18kg in masina, observa ca cel de-al treilea obiect nu incape si face cel de-al doilea drum. In final ia ultimul obiect si face al treilea drum. )

Exista, totusi, o solutie in care pot fi parcurse 2 drumuri: Punem primul si al treilea obiect in masina (12kg in total) si facem un drum, si inca un drum pentru obiectul de 18kg.

Dar am presupus ca algoritmul greedy descris este optim, deci contradicitie.

2. Fie OPT, numarul de drumuri in solutia optima. Demonstrati ca algoritmul greedy prezentat mai sus efectueaza cel mult 2OPT drumuri. (1 punct).

**Demonstratie:**