BARAJ 1 – Descrierea soluţiilor

Sortare prin inversiuni

O sugestie valoroasă ascunsă în textul problemei este să consideraţi întâi numai şiruri de 0 şi 1. Pentru acest caz, se poate găsi un algoritm cu costul **O(NlgN)**, bazat de divide-and-conquer. Se împarte şirul în două bucăţi de mărimi egale. Fiecare bucată se sortează recursiv. Apoi şirul va avea forma: 0..01..1 0..01..1

Acum se poate efectua o singura inversiune în jurul punctului unde am împărţit şirul. Astfel, inversăm elementele 1 din prima jumătate şi elementele 0 din a doua jumătate, şi obţinem un şir sortat. Costul acestei inversiuni este cel mult numărul de elemente din şirul pe care îl sortăm. Deci pe fiecare nivel de recursivitate avem un cost de **O(N)**, şi în total costul este **O(NlgN)**.

Pentru cazul general, folosim algoritm anterior ca pas auxiliar pentru a implementa QuickSort, şi obţinem un cost de **$O(Nlg^2N)$**. Pentru a sorta cu QuickSort, alegem mediana şirului, şi partiţionăm şirul, punând elementele mai mici ca mediana în stânga şi elementele mai mari ca mediana în dreapta. Cum se poate face această partiţionare eficient? Să presupunem ca toate elementele mai mici ca mediana le renumerotăm ca „0”, şi toate elementele mai mari ca mediana le renumerotăm ca „1”. Acum putem aplica algoritmul pentru cazul 0-1, şi cu un cost de O(NlgN) putem face partiţionarea. Apoi continuăm sortarea generală pe bucata din stânga şi cea din dreapta. Fiindcă costul de partiţionare e **O(NlgN)**, spre deosebire de **O(N)** la un QuickSort normal, avem un cost total de **$O(Nlg^2N)$.**

Constanta se poate calcula fără probleme şi se vede uşor ca e mai mică decât ½. Algoritmul are un cost mai mic de 2 milioane pentru cel mai dificil caz.

Trans

Problema admite o soluţie simplă, având complexitatea **O(N*K)** pentru fiecare tip de camion. Se calculează pentru fiecare bloc de piatră **i**, suma totală minimă plătită, SMIN[i], astfel încât fie transportate primele **i** blocuri. Pentru aceasta, se variază **j** între **i** şi **i-K+1**, reprezentând primul bloc din cadrul transportului care conţine blocul **i**. Acestă soluţie poate fi optimizată până la o complexitate liniară. Se va menţine o structură de tip coadă, sortată crescător. Fiecare element al structurii corespunde cazului în care ultimul transport efectuat începe la una din poziţiile 1,2,..,i. Pe măsură ce se calculează valorile SMIN[i], se introduc aceste valori (puţin modificate), la sfârşitul cozii, eliminându-se elementele de la sfârşit care au valori mai mari decât valoarea nou introdusă (sfârşitul cozii funcţionează ca o stivă). Pentru a calcula SMIN[i], se alege valoarea de la începutul cozii. Dacă această valoare corespunde unei poziţii de start la distanţă mai mare decât **K** faţă de **i**, atunci valoarea se elimină şi se încearcă utilizarea valorii următoare. Complexitatea finală a soluţiei este **O(N*Q)**.

Peri

Soluţia simplă (de 40 de puncte) are complexitatea **$O(N^4)$** şi nu vom insista asupra ei. Există o soluţie mai bună, cu complexitate **$O(N^3)$**. Se alege linia de sus şi linia de jos a dreptunghiului (**$O(N^2)$** posibilităţi). Pentru fiecare pereche avem o „fâşie” şi trebuie să căutăm soluţia cea mai bună din fâşia respectivă. Pentru aceasta vom parcurge coloană cu coloană, iar la fiecare pas vom ţine soluţia cea mai bună din stânga (şi numărul soluţiilor cele mai bune). O soluţie din stanga constă în 3 părţi ale perimetrului dreptunghiului (linia verticală din stanga şi cele două linii orizontale). La fiecare pas adăugăm la soluţia curentă diferenţa dintre numărul de **1** şi **0** de la două poziţii ale matricii (una care completează linia de sus şi una pentru cea de jos) şi comparăm cu o nouă soluţie posibilă ce ar corespunde dreptunghiurilor care au marginea din stânga la coloana curentă. Acest pas constă în **O(N)** operaţii, rezultând complexitatea totală de **$O(N^3)$**.