Soluție problema DIF2

Gheorghe Manolache,
Colegiul Național de Informatică, Piatra Neamț

Se observă că valoarea d1 este egală cu diferența dintre valoarea maximă și valoarea minimă a elementelor din șirul Y. Se pot calcula valorile extreme ale elementelor lui X iar d1 va fi diferența pătratelor lor.

Pentru d2 nu se poate construi șirul Y și apoi să se ordoneze descrescător deoarece există o limitare a memoriei disponibile și algoritmi klogk pentru sortare(k=nxn). Voi prezenta un algoritm de complexitate $O(DIM\_n+nlogDIM\_V^2)$, unde DIM_V este valoarea maxima din X. Se vor număra câte perechi de elemente din X au valoarea mai mică sau egală decât i. Se poate face numărarea în O(n) dacă vom precalcula un sir c[i] cu numărul de valori din X mai mici sau egale cu i. Pentru a afla pe ce poziție va fi un produs p, vom utiliza șirul c precalculat și vom număra pentru fiecare element din șir câte elemente contribuie la construirea unei valori ce nu depășește p. Vom utiliza căutarea binara pentru determinarea celor două valori. Nu e necesar să facem sortări descrescătoare, se deduce ușor poziția necesară dacă vom face calculul în cazul ordonării crescătoare.

SOLUTIA COMISIEI (STUDENTI LUCIAN BICSI si PETRU-ERIC STAVARACHE)

Soluție O(V logV):

Ținem un vector de frecvență freq[i] pentru valorile elementelor împreună cu sume parțiale pe prefixe pe acest vector Sum[i].

Pentru a afla produsul aflat pe o poziție Z, căutăm binar produsul, iar pentru un produs P fixat trebuie sa vedem cate perechi (a, b) exista cu a * b <= P.

Pentru asta vom parcurge fiecare valoare posibila, iar pentru o valoare A fixata deducem ca B <= P/A.

Așadar toate elementele

cu valoarea cel mult B vor forma perechi posibile împreună cu A. Vor exista f[A] * Sum[B] astfel de perechi.

Daca numarul total de perechi de acest fel este <= Z, atunci soluția >= P.

Solutie O(N log^2V):

Similar cu soluția anterioara, doar ca in loc sa ținem un vector de frecventa, sortam vectorul inițial.

Cu Z si P fixate(ca in soluția anterioara), pentru a numără perechile vom face:

Pentru o poziție i din vector, trebuie sa găsim j maxim astfel încât v[i] * v[j] <= P.

Odată găsit, (v[i], v[1]), ..., (v[i], [j]) toate vor fi perechi posibile.

Acest j poate fi căutat binar, deoarece vectorul este sortat.

Soluție O(N logV):

Plecând de la soluția anterioară, observăm că:

Suntem la pasul și știm j pentru acest pas.

Când trecem la pasul i + 1, j poate doar sa scadă.

(Deoarece v[i + 1] >= v[i])

Așadar în loc să căutăm binar vom ține doi pointeri cu care vom parcurge vectorul.