Problema `bvârcolaci`

Autori: Eugenie Daniel Posdărăscu

Soluţie O(N * N):

Fixăm Left şi Right, capătul stânga respectiv dreapta al subsecvenţei. Este suficient să verificăm doar cel mai frecvent element al subsecvenţei dacă are sau nu size / 2 + 1 elemente. Pe măsură ce variem capătul dreapta, actualizăm frecvenţa elementului nou inserat într-un vector de frecvenţă. Cel mai frecvent element va fi ori precedentul cel mai frecvent element, ori elementul nou inserat (ne decidem în funcţie de valorile din vectorul de frecvenţe).

Acestă soluţie obţine între 20 şi 30 de puncte.

Soluţie O(N * sigma):

Sigma reprezintă numărul de elemente distincte din vector. Fixăm X, un element distinct din vectorul initial (vectorul V). Acum dorim să aflăm în O(N) câte subsecvenţe există care îl au pe X ca element majoritar.

Primul pas este să construim un vector auxiliar în felul următor: la poziţia P, dacă avem valoarea X în V, punem +1, altfel -1. Facem sume parţiale pe acest vector şi fixăm capătul dreapta Right al subsecvenţei. Rămâne să selectăm un capăt stânga Left astfel încât elementul X să fie majoritar în intervalul [Left, Right]. Acest lucru este echivalent cu a selecta un capăt stânga Left astfel încât suma elementelor în vectorul auxiliar pe intervalul [Left, Right] să fie strict pozitivă. Cu ajutorul sumelor parţiale, se pot număra uşor toate capetele Left care respectă această proprietate. O implementare uşoară poate să fie realizată cu AIB, dar se poate scăpa şi de log dacă aveţi în vedere că sumele cresc/scad cu 1.

Soluţie O(N * sqrt(N)):

Impărţim numerele în 2 categorii:

- Elementele care au frecvenţa mai mică de sqrt. Pentru aceste elemente observăm că nu pot să fie elemente majoritare într-o subsecvenţă de lungime mai mare ca 2 * sqrt. Aceste elemente pot fi tratate brute-force.
- Elemente care au frecvenţa mai mare de sqrt. Pentru aceste elemente observăm că sunt maxim sqrt(N) astfel de valori distincte. Ca urmare, putem pentru fiecare din ele să aplicăm algoritmul prezentat mai sus în soluţia de O(N * sigma).

Exista multe solutii alternative în O(N * sqrt(N)). O alta variantă se bazează pe algoritmul lui Mo în care facem brute-force secvenţele mici, iar pentru un bloc de bucăţi de radical observăm că este suficient să ţinem doar cele mai frecvente 2-3 elemente. Acestă soluţie este în schimb mai tehnica şi mai neintuitivă.

Soluţiile în această complexitate obţin între 70-90 de puncte, dar nu este exclus să se obţina şi 100 de puncte.

Soluţie O(N * log(N) * log(N)):

Aplicăm Divide-Et-Impera. Vrem să aflăm numărul de subsecvenţe bune pe un interval [Left, Right]. Impărţim intervalul în 2 şi rezolvăm recursiv partea stângă şi partea dreaptă. Rămâne de văzut cum combinăm cele 2 zone.

Observaţia de bază este că sunt maxim log(N) candidaţi pentru un posibil element majoritar. Pentru fiecare candidat, determinăm câte subsecvenţe îl au ca şi element majoritar în O(N) folosind algoritmul prezentat în soluţiile anterioare. Întrebarea rămâne de ce sunt maxim log(N) candidaţi şi cum îi determinăm.

O secvenţă [Left, Right] care admite un element majoritar X are următoarea proprietate: X este element majoritar ori în secvenţa [Left, Mid], ori în secvenţa [Mid + 1, Right]. Deci putem să analizăm independent fiecare jumătate şi să extragem posibilele elemente majoritare. Să zicem că ne uităm la jumătatea stângă (capătul dreapta va fi mereu Mid). Dacă ar fi să completăm element cu element pornind din capătul Mid, am pune 1 element ca să obţinem primul candidat, 2 elemente ca sa obţinem al doilea candidat, 4 elemente ca sa obţinem al 3-lea candidat, etc. În final avem log candidaţi.

Această soluţie obţine 100 de puncte.

Soluţie O(N * log(N)):

Pornind de la soluţia în O(N * sigma), ideea acestei soluţii se bazează pe următorul principiu. Pentru un element X fixat, am dori să aflăm numărul de subsecvenţe care îl conţin ca element majoritar în O(numărul de apariţii a lui X în vector) * log(N) în loc de O(N). Complexitatea astfel se amortizează la O(N * log(N)) pentru toate elementele.

Algoritmul cu +1,-1 poate să fie adaptat cu ajutorul unor structuri de date (arbori de intervale spre exemplu), dar trebuie să aveţi grijă la foarte multe detalii (în principiu, trebuie avut în vedere faptul că o secvenţa nu incepe/termină într-un capăt cu +1). Dezvoltarea soluţiei rămâne temă pentru acasă întrucât detaliile sunt mai complicate.

Această soluţie obţine 100 de puncte.