Sub2max

Autor: stud. Rusu Raluca-Maria, Universitatea Babeș-Bolyai, Cluj-Napoca

Cerința problemei este de a descoperi, într-un șir de numere, subsecvența cu lungimea cuprinsă între S și D care are cea mai mare sumă a elementelor, iar această sumă trebuie să fie și o putere a lui K.

Soluție $O(N \cdot D^2 \cdot \log_K N)$ - 30 de puncte

O soluție posibilă constă în explorarea tuturor subsecvențelor posibile cu lungimi între S și D, calculând suma fiecărei subsecvențe prin adunarea directă a elementelor la fiecare pas.

Se citește din fișierul de intrare *sub2max.in* dimensiunile necesare (N, S, D, K) și elementele șirului.

Pentru fiecare index de start posibil i de la 1 la N, se verifică fiecare subsecvență posibilă cu lungimea între S și D din șir, de la $i + S - 1$ până la $i + D - 1$ sau N, apoi se calculează suma numerelor din subsecvențele generate parcurgând toate numerele.

După calcularea sumei unei subsecvențe, verificăm dacă suma este o putere a lui K prin metoda de schimbare a bazei. Aceasta este cea mai rapidă variantă pentru această problemă:

- Dacă $m = 1$, acesta poate fi considerat pentru puterea 0 a oricărui număr K. Astfel, se returnează adevărat.
- Dacă K este egal cu 1, singura valoare a lui m care ar fi considerată este 1 însuși (deoarece 1 la puterea 0 este 1). Astfel, se returnează adevărat doar dacă m este și el 1.
- Urmează o cautare care continuă cât timp m este divizibil cu K în care m este împărțit prin K. Această operație este repetată până când m nu mai este divizibil cu K (dacă restul nu este 0).
- Returnează adevărat doar daca m a ramas 1, altfel returnează fals.

Dacă o sumă respectă condițiile (este mai mare decât suma maximă găsită până în momentul respectiv și este putere a lui K), sau dacă suma este egală cu suma maximă dar subsecvența este mai scurtă sau are un indice de început mai mare în cazul lungimilor egale, se actualizează suma maximă, indicele de început al subsecvenței și lungimea minimă.

În final, programul scrie în fișierul *sub2max.out* suma maximă care este putere a lui K și indicele de început al subsecvenței care îndeplinește condițiile cerute. Dacă nu se găsește nicio subsecvență care să respecte condițiile, se va afișa -1 atât pentru valoarea sumei maxime, cât și pentru indicele de început.

Soluție $O(N \cdot D \cdot \log_K N)$ - 100 de puncte

Pentru optimizarea soluției putem menține un vector de sume (A) ale elementelor șirului (E) dat. Acest pas ne ajută pentru a identifica rapid sumele subsecvențelor din șir și verificarea dacă aceste sume sunt puteri ale lui K.

Mai întâi, se citește din fișierul de intrare *sub2max.in* dimensiunile necesare (N, S, D, K) și elementele șirului (E). Se construiește vectorul de sume (A) definit mai sus odată cu citirea elementelor șirului, astfel încât fiecare element $A[i]$ reprezintă suma elementelor de la începutul șirului până la poziția i $(A[i] = A[i-1] + E[i])$.

Pentru fiecare index de start posibil i de la 1 la N, se verifică fiecare subsecvență posibilă cu lungimea între S și D din șir, de la $i + S - 1$ până la $i + D - 1$ sau N, apoi se calculează suma numerelor din

subsecvențele generate, utilizând sumele precalculate pentru eficiență ($sum = A[j] - A[i-1]$), și lungimea acesteia ($length = j - i + 1$). Suma este verificată dacă este putere a lui K prin metoda de schimbare a bazei, definită mai sus.

Dacă o sumă respectă condițiile (este mai mare decât suma maximă găsită până în momentul respectiv și este putere a lui K), sau dacă suma este egală cu suma maximă dar subsecvența este mai scurtă sau are un indice de început mai mare în cazul lungimilor egale, se actualizează suma maximă, indicele de început al subsecvenței și lungimea minimă.

În final, programul scrie în fișierul *sub2max.out* suma maximă care este putere a lui K și indicele de început al subsecvenței care îndeplinește condițiile cerute. Dacă nu se găsește nicio subsecvență care să respecte condițiile, se va afișa -1 atât pentru valoarea sumei maxime, cât și pentru indicele de început.