cb

Autori: stud. Lorintz Alexandru, stud. Moca Andrei-Cătălin, Universitatea Babeș-Bolyai Cluj-Napoca

Descrierea problemei

Se dau două numere naturale n și m și un șir de n numere naturale **diferite** s. Sarcina este de a construi un șir a de m numere naturale astfel încât:

1. Fiecare element din a aparține intervalului $[1, 10^9]$.
2. Șirul a este crescător (adică $a_1 \leq a_2 \leq \cdots \leq a_m$).
3. Dacă, pentru fiecare număr x din s, se efectuează o căutare binară în a folosind algoritmul dat , atunci suma valorilor returnate să fie minimă.

Observații

Cheia problemei constă în reducerea numărului de comparații efectuate de funcția `binary_search` atunci când se caută elementele din s în șirul a. Observația esențială este că algoritmul de căutare binară începe prin verificarea elementului aflat la mijlocul intervalului curent. Astfel, pentru a opri cât mai devreme fiecare căutare, elementele din s trebuie plasate în a în poziții care să le permită să fie descoperite rapid.

Abordarea soluției

1. **Sortarea șirului s:** Pentru a putea plasa elementele în ordine crescătoare în șirul a, se sortează inițial șirul s.
2. **Determinarea pozițiilor optime pentru plasarea elementelor din s:** Pentru a minimiza numărul de comparații în căutările binare, se plasează elementele din s în pozițiile mediane ale intervalelor din șirul a.
   - Se consideră că șirul a are m poziții indexate de la 0 la $m-1$.
   - Se inițializează o coadă cu intervalul complet $[0, m-1]$.
   - Pentru fiecare interval $[l, r]$ extras din coadă se calculează poziția mediană $mid = \lfloor (l+r)/2 \rfloor$ și se memorează această poziție.
   - Se adaugă într-o coadă intervalele rămase, $[l, mid-1]$ și $[mid+1, r]$, dacă acestea sunt valide.
   - Procedura se repetă până se selectează exact n poziții (corespunzătoare elementelor din s) sau șirul a este completat complet (asta în cazul în care $m \leq n$).
3. **Construirea șirului a:**
   - Se sortează pozițiile selectate astfel încât elementele din s să fie plasate în ordine crescătoare.
   - Se inițializează șirul a cu valoarea 1 pe toate pozițiile, asigurând astfel că elementele care nu provin din s sunt cele mai mici posibile și contribuie la minimizarea soluției lexicografice.
   - Se plasează elementele din s (după sortare) în pozițiile determinate.
   - Pentru a respecta condiția de ordine nen-descrescătoare, se parcurge șirul a și, pentru fiecare $i \geq 1$, dacă $a[i] < a[i-1]$, se actualizează $a[i]$ la $a[i-1]$.

Analiza complexității

- Sortarea șirului s se face în $O(n \log n)$.
- Determinarea pozițiilor pentru elementele din s se realizează printr-un proces de împărțire recursivă a intervalului complet $[0, m-1]$ în segmente, în care se selectează poziția mediană a fiecărui segment. Astfel, se obțin poziții distribuite uniform pentru plasarea elementelor din s. Numărul de operații efectuate este proporțional cu numărul de poziții selectate, adică $O(n)$.
- Construirea și actualizarea șirului a se realizează în $O(m)$.

Astfel, complexitatea totală este $O(n \log n + m)$, suficientă pentru limitele impuse de problemă.