

maşina – Soluţie

Problema este de programare dinamică. Există mai multe feluri de rezolvare, iar acesta este unul dintre ele:

Asemănăm acel drum din mijloc cu o stiva, iar operațiile sunt paranteze deschise (atunci când vine o mașina în stivă) si paranteze închise (când pleacă o mașină din stivă).

Construim:

A[i][j] = numărul de posibilități de a ajunge în poziția cu i paranteze deschise si j închise
B[i][j] = numărul de posibilități de a încheia situația (N deshise, N închise) din poziția (i, j), adică i deschise, j închise

Presupunem X < Y, altfel inversăm si scădem la sfârșit.

Variem momentul în care apare X în stivă (baza), și anume poziția pe care se gasește în stivă când apare.

Construim:

Q[i][j] = numărul de posibilități de procesare a i mașini având j în stivă, ținem i între x și y și j între baza și baza + diferența între x și y

Momentul în care se ia o decizie (dacă x e în fața lui y sau invers), este atunci când stiva e la nivelul baza sau când urmează să procesăm pe y. Adunăm aceste momente, înmulțindu-le cu B-ul corespunzător (adică câte posibilități sunt până la sfârșit) și cu A-ul corespunzător (câte posibilități să ajungem la baza).

Întrucât precizia este doar de două zecimale, putem folosi tipul double pentru a calcula toate posibilitățile.



matrice - Soluție

Ideea este de a permuta rândurile matricii astfel încât să respecte proprietatea cerută și anume aceea ca oricare coloană să nu conțină un subșir strict crescător de lungime mai mare decât $\lceil \sqrt{N} \rceil$ și de a scrie permutarea respectivă folosind operațiile permise.

Permutarea cerută se obține astfel: mai întâi se sortează rândurile matricei descrescător după valorile primei coloane. Apoi pentru grupe de câte $\lceil \sqrt{N} \rceil$ rânduri, se sortează descrescător după valorile celei de-a doua coloane. Cel mai lung subșir crescător în prima coloană poate fi găsit numai în cadrul grupelor și deci are lungimea maxim $\lceil \sqrt{N} \rceil$. În a doua coloană, cel mai lung subșir crescător poate fi obținut alegând cât mai convenabil un element din fiecare grupă (cele din cadrul grupelor sunt sortate descrescător) și deci are lungimea maximă posibilă egală cu numărul grupelor, adică $\lceil \sqrt{N} \rceil$.

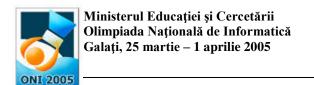
Pentru a obține o permutare cu ajutorul operațiilor S_3 , J_3 , S_4 , J_4 , observăm că orice transpoziție se poate scrie în funcție de aceste operații. Orice permutare se poate descompune în cicluri (suma lungimilor lor fiind egală cu N). Cum orice ciclu de lungime L se poate descompune în L transpoziții și (vom demonstra că) orice transpoziție se poate obține prin efectuarea a 2 dintre operațiile permise, rezultă că orice permutare se poate obține prin efectuarea a cel mult 2*N operații permise.

Pentru a arăta că orice transpoziție se poate obține din operațiile S_3, J_3, S_4, J_4 , considerăm cazurile

$$\begin{bmatrix} a \\ b \\ c \\ d \\ d \end{bmatrix} \underbrace{S_4}_{a} \begin{bmatrix} b \\ c \\ d \\ a \end{bmatrix} \underbrace{J_3 \text{ (pentru ultimele 3)}}_{c} \begin{bmatrix} b \\ a \\ c \\ d \end{bmatrix} \Rightarrow \text{putem interschimba rândul } a \text{ cu } b \text{ dacă după ele mai sunt cel puţin 2 rânduri}$$

$$\begin{bmatrix} a \\ b \\ c \\ d \\ d \end{bmatrix} \underbrace{S_4}_{a} \begin{bmatrix} b \\ c \\ d \\ a \end{bmatrix} \underbrace{J_3 \text{ (pentru primele 3)}}_{c} \begin{bmatrix} d \\ b \\ c \\ a \end{bmatrix} \Rightarrow \text{putem interschimba rândul } a \text{ cu } d \text{ dacă}$$
 între ele mai sunt cel puţin 2 rânduri

Pentru două rânduri a și b, cel puțin unul dintre cazuri este adevărat.



ziduri – Soluție

Soluția 1

Se considera graful format astfel: pentru fiecare camera se considera un nod, iar intre doua camere vecine pe orizontala/verticala (deci doua noduri) o muchie de cost 1 daca nu exista zid intre cele doua camere, respectiv p daca exista zid. Pe acest graf se calculeaza cu algoritmul lui Dijkstra costurile minime de la camera stanga-sus la toate celelalte camere. Apoi se construieste graful orientat aciclic al tuturor drumurilor minime de la camera stanga sus la camera dreapta jos. Pe acest graf aciclic se calculeaza cu o parcurgere drumul dintre cele doua camere care foloseste numar minim de muchii formate din ziduri. Pentru o complexitate mica, algoritmul lui Dijkstra trebuie implementat cu heapuri. Astfel, complexitatea totala e de N^2logN.

Soluția 2

Pe acelasi graf, putem afla aceasta distanta folosind algoritmul Bellman – Ford – Moore. Desi acesta poate conduce la o complexitate de $O(N^4)$, ne putem baza pe faptul ca p este destul de mic (mai mic decat 100 (adica maximul lui N)) si acesta reduce la $O(N^3)$ complexitatea, cu care putem obtine punctaj maxim.