DESCRIEREA SOLUȚIILOR, OLIMPIADA JUDEȚEANĂ DE INFORMATICĂ, CLASA A X-A

Problema Circular

Propusă de: Prof. Boca Alina Gabriela – Colegiul Național de Informatică ,,Tudor Vianu" București

Cerința 1. Pentru rezolvarea primei cerințe se parcurge șirul de litere albastre și pentru oricare două litere alăturate se calculează cea mai mică distanță dintre litera curentă și litera următoare din șir.

Pentru a calcula distanta minimă dintre două litere A_i și A_{i+1} va trebui sa calculăm minimul dintre cele două cazuri:

- de la A_i la A_{i+1} în sensul acelor de ceasornic
- de la A_i la A_{i+1} în sens opus acelor de ceasornic

Pentru a lua în considerare faptul că imprimanta începe de pe poziția A, putem considera A_0 =A.

Complexitatea temporală este $\mathcal{O}(N)$.

Rezolvarea primei cerințe obține 24 de puncte.

Cerința 2. Pentru a rezolva cerința 2, vom precalcula un tablou bidimensional de 26×26:

$cost_{i,j}$ = numărul minim de pasi pentru a ajunge de la a i-a litera din alfabet, la a j-a.

Ulterior, se parcurge șirul literelor albastre și între fiecare două litere albastre A_i, A_{i+1}, se caută în șirul literelor roșii acele litere R pentru care distanța $cost_{A_i,R} + cost_{R,A_{i+1}}$ este minimă.

Costul minim reprezintă suma distantelor minime obținute, la care trebuie adăugat costul de a aduce capul de printare de la L_0 =A la L_1.

Pentru a construi șirul minim lexicografic, între oricare două litere albastre, vom insera dintre toate literele roșii ce obțin un cost minim pe cea mai mică lexicografic.

Numărul de soluții reprezintă produsul dintre numărul de posibilități de a insera o literă roșie între două litere albastre care generează distanța minimă.

Complexitatea temporală este: $\mathcal{O}(N \cdot \Sigma)$, unde $\Sigma = 26$ reprezintă dimensiunea alfabetului.

Rezolvarea celei de-a doua cerințe obține 76 de puncte.

Problema Pulsar

Propusă de: Stud. Tulbă-Lecu Theodor-Gabriel – Universitatea Politehnica din București

Observații. Pentru a rezolva problema, inițial trebuie făcută următoarea observație: După un anumit timp minim T, pulsarele vor reveni înapoi în starea inițială de la momentul de timp $t = 0$. Vom numi acest timp T, *perioada pulsarelor*.

În continuare, trebuie făcută observația că dacă toate pulsarele revin în starea inițială după T unități de timp, cum un pulsar P_i de perioadă r_i, se află în starea inițială doar la momente de timp care sunt multiplii de r_i, atunci T este și el multiplu al lui r_i.

Astfel, T este cel mai mic multiplu comun al perioadelor pulsarilor: $T = cmmmc(r_1, r_2, \ldots, r_P)$. Cum pentru restricțiile problemei: $1 \leq r_i \leq 6 \; \forall 1 \leq i \leq P$, rezultă că $T \leq 60$.

Cerința 1 – Subtask 1. În urma observațiilor făcute, cerința 1, poate fi exprimată astfel: Pentru fiecare moment de timp de la 0 la $T-1$ câte sectoare ale galaxiei din cele $N \times N$ sunt afectate de cel puțin un pulsar?

Acest lucru poate fi rezolvat calculând pentru fiecare moment de timp, un tablou bidimensional:

$$afectat_{i,j} = \begin{cases} 1, \text{dacă există cel puțin un pulsar care afectează sectorul}(i,j) \\ 0, \text{altfel} \end{cases}$$

Tabloul bidimensional *afectat* poate fi calculat prin marcarea pentru fiecare pulsar în parte a tuturor sectoarelor afectate de acesta la momentul de timp actual cu 1, toate valorile din *afectat* fiind inițial inițializate cu 0.

Răspunsul pentru cerința 1 este maximul dintre numărul de valori de 1 din tabloul *afectat*, pentru fiecare moment de timp de la 0 la $T-1$.

Complexitatea temporală este: $\mathcal{O}(T \cdot (N^2 + P * {R_{max}}^2))$, unde R_{max} este perioada maximă a unui pulsar.

Pentru rezolvarea corectă a cerinței 1 se pot obține 19 puncte.

Cerința 2.

Subtask 2. Primul subtask al cerinței a doua reprezintă un caz particular al problemei. Dacă $r_i = 1 \; \forall 1 \leq i \leq P$, atunci toti pulsarii vor afecta doar sectoarele în care aceștia se află.

Astfel, problema devine găsirea un drum de lungime minimă de la sectorul (x_s, y_s) la sectorul (x_f, y_f) într-o hartă ce conține obstacole.

Acest lucru se poate rezolva utilizând algoritmul lui Lee. (1)

Complexitatea temporală este: $\mathcal{O}(N^2 + P)$.

Pentru rezolvarea corectă a subtaskului 2 se pot obține 22 de puncte.

Subtask 3. Pentru acest subtask, cum $N \leq 10$, harta galaxiei are dimensiuni suficient de mici pentru ca problema să fie rezolvată utilizând metoda backtracking. Se vor genera toate drumurile posibile de la sectorul (x_s, y_s) la sectorul (x_f, y_f), iar la fiecare pas se verifică dacă celula adăugată la drum, este afectată de vreun pulsar.

Subtask-urile 4 și 5. Pentru a rezolva complet cerința a doua, trebuie să ne folosim de observațiile făcute anterior.

Știind că harta galaxiei este periodică cu o perioadă T putem să reprezentăm starea navei sub forma unui triplet (x, y, t) unde x și y reprezintă coordonatele navei, iar t reprezintă starea hărții galaxiei.

Dacă nava se află la într-o stare (x, y, t), aceasta la următorul moment de timp se va afla la:

(1) $(x, y, (t+1)\%T)$, dacă nava stă pe loc
(2) $(x, y-1, (t+1)\%T)$, dacă nava se deplasează la stânga
(3) $(x, y+1, (t+1)\%T)$, dacă nava se deplasează la dreapta
(4) $(x+1, y, (t+1)\%T)$, dacă nava se deplasează în jos
(5) $(x-1, y, (t+1)\%T)$, dacă nava se deplasează în sus

Astfel, nava se va deplasa intr-un tabel tridimensional, a treia dimensiune fiind starea hărții, iar obstacolele vor fi create de zonele afectate de pulsari. Această problemă se poate rezolva tot cu ajutorul algoritmului lui Lee, care va trebui modificat pentru a funcționa pe trei dimensiuni.

Răspunsul va fi timpul minim cu care se poate ajunge din $(x_s, y_s, 0)$ în (x_f, y_f, t) $\forall 0 \leq t < T$.

Complexitatea temporală este: $\mathcal{O}(T \cdot (N^2 + P * {R_{max}}^2))$.

Problema Transport

Propusă de: Stud. Cotor Andrei – Universitatea "Babeș-Bolyai" din Cluj-Napoca

Observații.

(1) O rută Regio reprezintă o subsecvență din șirul de stații de forma: $[st, st+1, st+2 \ldots, dr]$, $1 \leq st < dr \leq N$, unde st și dr reprezintă capetele rutei.
(2) O rută Expres reprezintă un subșir din șirul de stații de forma: $[st, i_1, i_2, \ldots, i_k, dr]$, $1 \leq st < i_1 < i_2 < \cdots < i_k < dr \leq N$, $k \geq 0$, unde st și dr reprezintă capetele rutei.
(3) Relația care corespunde restricțiilor din enunț e: $D_{st} + D_{dr} = C \cdot (X_{dr} - X_{st})$. Aceasta poate fi rescrisă în felul următor: $D_{st} + D_{dr} = C \cdot (X_{dr} - X_{st}) \implies D_{st} + D_{dr} = C \cdot X_{dr} - C \cdot X_{st} \implies D_{st} + C \cdot X_{st} = C \cdot X_{dr} - D_{dr}$
(4) În relația obținută la *Observația 3* expresia din stânga egalului depinde doar de st (X_{st}, D_{st}), iar cea din dreapta egalului doar de dr (X_{dr}, D_{dr})

Cerința 1 ($T = 1$).

Subtask 1 (12 puncte). Se parcurge fiecare subsecvență, fixând capătul din stânga și cel din dreapta, și se verifică condiția $D_{st} + D_{dr} = C \cdot (X_{dr} - X_{st})$. Dacă este respectată condiția se incrementează rezultatul.

Complexitate temporală: $\mathcal{O}(N^2)$.

Optimizare. Soluția pentru *Subtask-ul 1* poate fi optimizată parcurgând, pentru un dr fixat, doar indicii st pentru care (st, dr) pot forma o pereche de capete validă. Mai exact, după ce a fost fixat dr, se calculează valoare expresiei $C \cdot X_{dr} - D_{dr}$ și se parcurg doar indicii st pentru care $D_{st} + C \cdot X_{st} = C \cdot X_{dr} - D_{dr}$ (*Observația 3*).

Cu ajutorul acestei optimizări se pot obține 12 puncte din cele 26 acordate pentru *Subtask-ul 2* (în plus față de cele acordate pentru subtask-urile precedente).

Subtask 2 (26 puncte). Se fixează capătul dreapta dr. Trebuie numărate câte capete stânga st există astfel încât perechea de capete (st, dr) este una validă, mai exact $D_{st} + C \cdot X_{st} = C \cdot X_{dr} - D_{dr}$ (*Observația 3*).

Pentru a obține aceste valori va fi nevoie de o normalizare, valorile expresiilor $D_{st} + C \cdot X_{st}$ sau $C \cdot X_{dr} - D_{dr}$ putând fi foarte mari. Astfel pentru fiecare stație i se reține într-un vector de lungime $2N$, care urmează să fie utilizat pentru normalizare, valorile $D_i + C \cdot X_i$ și $C \cdot X_i - D_i$.

Complexitate temporală: $\mathcal{O}(NlogN)$.

Cerința 2 ($T = 2$).

Subtask 3 (6 puncte). N fiind mic se poate utiliza Bactracking pentru a genera toate subșirurile din șirul de stații. Pentru fiecare subșir generat se verifică dacă respectă condițiile din enunț.

Subtask 4 (15 puncte). Se fixează fiecare combinație de două capete ale unei rute (notate cu st, respectiv dr), care respectă condițiile din enunț. Între cele două capete fixate există $dr - st - 1$ stații. Astfel pentru perechea de capete (st, dr) numărul de rute Expres este egal cu numărul de subșiruri care se pot forma din subsecvența $[st+1, st+2, \ldots, dr-1]$, adică $2^{dr-st-1}$.

Complexitate temporală: $\mathcal{O}(N^2)$ sau $\mathcal{O}(N^2 logN)$ în funcție de implementare.

Optimizare. Optimizarea prezentată pentru *Cerința 1* poate fi utilizată și în acest caz.

Cu ajutorul acestei optimizări se pot obține 14 puncte din cele 41 acordate pentru *Subtask-ul 5* (în plus față de cele acordate pentru subtask-urile precedente).

Subtask 5 (41 puncte). În mod asemănător cu *Subtask-ul 2*, se face normalizarea și se fixează capătul dreapta dr. Fie $\{st_1, st_2, st_3 \dots st_k\}$ mulțimea de capete stanga cu care dr formează o pereche de capete validă. Astfel numărul de rute valide care îl au capăt dreapta pe dr este:

$$2^{dr-st_1-1} + 2^{dr-st_2-1} + 2^{dr-st_3-1} + \cdots + 2^{dr-st_k-1}$$

Relația se prelucrează și se obține:

$$2^{dr} \cdot (\frac{1}{2^{st_1+1}} + \frac{1}{2^{st_2+1}} + \frac{1}{2^{st_3+1}} + \cdots + \frac{1}{2^{st_k+1}})$$

Relația se înmulțește și se împarte cu 2^N și se obține:

$$2^{dr-N} \cdot (2^{N-st_1-1} + 2^{N-st_2-1} + 2^{N-st_3-1} + \cdots + 2^{N-st_k-1})$$

Suma $2^{N-st_1-1} + 2^{N-st_2-1} + 2^{N-st_3-1} + \cdots + 2^{N-st_k-1}$ se calculează în timpul parcurgerii pentru fixarea capătului dreapta.

Astfel complexitatea temporală este: $\mathcal{O}(NlogN)$.

Echipa

Problemele pentru această etapă au fost pregătite de:

- prof. Szabó Zoltan - Inspectoratul Școlar Județean Mureș
- prof. Boca Alina Gabriela – Colegiul Național de Informatică ”Tudor Vianu” București
- prof. Popescu Carmen - Colegiul Național ”Gheorghe Lazăr” Sibiu
- prof. Moț Nistor - Școala Gimnazială ”Dr. Luca” Brăila
- Stud. Tulbă-Lecu Theodor-Gabriel - Universitatea Politehnica București
- Stud. Cotor Andrei - Universitatea ¨Babeș-Bolyai¨ Cluj-Napoca
- Stud. Dăscălescu Ștefan Cosmin - Universitatea București
- Stud. Popa Bogdan-Ioan - Universitatea București
- Stud. Râpeanu George-Alexandru - Universitatea ¨Babeș-Bolyai¨ Cluj-Napoca
- Stud. Coroian David-Nicolae - Delft University of Technology
- Stud. Popescu Ioan - Universitatea Politehnica București

Referințe

[1] articol pbInfo – Algoritmul lui Lee, Prof. Silviu Candale