Editorial RoAlgo PreOJI 2025

1-8 Martie 2025

Cuprins

1 Multumiri

Acest concurs nu ar fi putut avea loc fără următoarele persoane:

- Matei Ionescu, Raul Ardelean, autorii problemelor și laureați la concursurile de informatică și membri activi ai comunității RoAlgo;
- Alex Vasiluță, fondatorul și dezvoltatorul principal al Kilonova;
- Ștefan Alecu, creatorul acestui șablon LaTeXpe care îl folosim;
- Ștefan Alexandru Nuță, Ștefan Neagu, Vlad Munteanu, Stefan Vilcescu, Tudor Iacob, David Curca, Vlad Tutunaru, Eduard Pirtac, Matei Neacsu, Rares Poinaru, testerii concursului, care au dat numeroase sugestii și sfaturi utile pentru buna desfășurare a rundei;
- Ștefan Dăscălescu, Andrei Iorgulescu și Luca Mureșan, coordonatorii rundelor;
- Comunității RoAlgo, pentru participarea la acest concurs.

,

2 Trumperika

AUTOR: IONESCU MATEI

2.1 Soluția oficială

Datorită faptului că taxa unui interval este o sumă de mai multe elemente, considerăm o soluție corectă una care calculează contribuția fiecărei litere în parte. Cu acestea fiind spuse, reprezentăm cu P_i suma taxelor tuturor intervalelor care se termină pe poziția i, dacă luăm în calcul doar litera b. Ușor deducem că P_i poate fi scris sub următoarea formă:

$$P_i = a_0 \cdot 2^0 + a_1 \cdot 2^1 + a_2 \cdot 2^2 + \cdots + a_k \cdot 2^k$$

În momentul în care vrem să calculăm P_{i+1}, avem de luat în calcul următoarele două posibilități:

(1) Caracterul de pe poziția $i+1$ este egal cu $b \rightarrow$ frecvența literei b crește cu o unitate.

(2) Caracterul de pe poziția $i+1$ nu este egal cu $b \rightarrow$ nu se întâmplă nimic.

Dacă creștem frecvența unui interval, atunci monomul de forma $a_i \cdot 2^i$ devine $a_i \cdot 2^{i+1}$, asta pentru fiecare i, ceea ce rezultă faptul că $P_{i+1} := P_i \cdot 2$.

Dacă adăugăm şi intervalul $[i + 1, i + 1]$ (care are, evident, frecvența 1), atunci $P_{i+1} := P_i \cdot 2 + 2$.
În caz contrar, singura modificare se află asupra intervalului $[i + 1, i + 1]$, deci $P_{i+1} := P_i + 1$.
Litera b va contribui la răspuns cu $P_1 + P_2 + \cdots + P_n$. Se poate generaliza uşor pentru fiecare literă.

2.1.1 Cod sursă

Soluție de 100

3 Orzeni

AUTOR: IONESCU MATEI

3.1 Soluția oficială

Pentru fiecare produs p cu $1 \le p \le K$, definim vectorul

$$L_p = \{i \mid 1 \le i \le N \text{ și } p \mid a_i\},$$

care conține indicii magazinelor în care produsul p este disponibil. Deoarece ordinea inițială a indicilor din L_p poate fi arbitrară, sortăm vectorul L_p în ordine crescătoare.

Obiectivul este să selectăm, pentru fiecare produs p, un magazin din L_p astfel încât suma deplasărilor între magazinele alese pentru produsele consecutive să fie minimă. Mai precis, dacă pentru produsul p alegem magazinul situat la poziția $L_p[j]$ și notăm cu $dp_{p,j}$ costul minim pentru a achiziționa produsele $1, 2, \ldots, p$ cu această alegere, atunci pentru tranziția la produsul $p + 1$ avem:

$$dp_{p+1,k} = \min_{1 \le j \le |L_p|} \left(dp_{p,j} + \left| L_{p+1}[k] - L_p[j] \right| \right),$$

unde $L_{p+1}[k]$ este magazinul selectat pentru produsul $p + 1$. Costul adițional $\left| L_{p+1}[k] - L_p[j] \right|$ reprezintă deplasarea între magazinul ales pentru produsul p și cel ales pentru produsul $p + 1$.

Inițial, pentru $p = 1$ nu se efectuează deplasări, astfel:

$$dp_{1,j} = 0 \quad \text{pentru orice } j.$$

Observați că, odată sortate, vectorii L_p și L_{p+1} sunt în ordine crescătoare, fapt ce ne permite să optimizăm calculul tranzițiilor folosind tehnici precum parcurgerea cu doi indicatori (”two-pointer”) sau precomputarea minimelor pe intervale (prefix/suffix). În acest mod, actualizarea stărilor pentru fiecare pas se realizează în timp $O(|L_p| + |L_{p+1}|)$.
Soluția finală va fi dată de:

$$\min_{1 \le j \le |L_K|} dp_{K,j},$$

care reprezintă costul minim total pentru achiziționarea tuturor celor K produse. Dacă pentru un anumit produs p vectorul L_p este gol, se concluzionează că nu se poate achiziționa produsul respectiv, caz în care soluția este -1.

4 Stars

Autor: Ardelean Raul

4.1 Subtask 1

Cum există doar un lac pe toată harta, de pe fiecare insulă putem intra în acel lac.
Astfel, putem intra în toate insulele. Rezultatul pentru fiecare query va fi suma stelelor de pe fiecare insulă.

4.2 Subtask 2

Harta se rezumă la un vector, astfel putem avea secvențe de 0 și de 1.
Să notăm insula lui Răzvan fiind insula cu indicele i, iar insula Andreei fiind insula cu indicele j.
Avem două cazuri:
* Dacă $i < j \Rightarrow$ putem intra doar pe insulele din secvența $[1, j]$.
* Dacă $i > j \Rightarrow$ putem intra doar pe insulele din secvența $[j, n]$.
Ca să aflăm în timp eficient suma stelelor pe un interval, ne vom folosi de sume parțiale.

4.3 Subtask 3

Pentru rezolvarea acestui subtask trebuie să observăm ce se întâmplă dacă legăm fiecare insulă de lacurile vecine, respectiv fiecare lac de insulele vecine. Ne rezultă un graf, astfel putem aplica pentru fiecare query câte un BFS.

4.4 Soluția oficială

Cheia rezolvări acestui subtask îi o extindere de la subtask-ul 3. Cum o insulă sau un lac este alcătuit din vecinii de pe cele 8 direcții din fiecare celulă din matrice $\Longrightarrow$ graful nostru este chiar un arbore.

Astfel, problema noastră se reduce la a afla: în câte noduri putem intra din u până în v, unde u = nodul corespunzător insulei lui Răzvan, respectiv v = nodul corespunzător insulei Andreei.

Dacă u se află mai aproape de rădăcina arborelui decât $v \Longrightarrow$ putem intra în toate nodurile din arbore, în afară de nodurile din subarborele nodului u. Astfel, putem răspunde în $\mathcal{O}(1)$ pentru aceast caz.

Dacă v se află mai aproape de rădăcina arborelui decât $u \Longrightarrow$ putem intra doar în nodurile din subarborele fiului lui v care îi pe același path cu nodul u. Astfel, trebuie să aflăm care îi fiul lui v care îi strămoș de a lui u.

Putem afla în $log(N)$ folosind K-th Ancestor sau ne putem folosi de ciclul eurelian.

Ne vom reține pentru fiecare nod de fiecare dată timer-ul când am ieșit din el, respectiv să ne reținem într-un vector parcurgerea euleriană.

Timer-urile din fiecare nod o să fie tot timpul în ordine crescătoare. Astfel, ca să aflăm prima intrare în nodul v după ce am ieșit din u va trebui să aflăm timpul în care am ieșit pentru ultima dată din u (să îl notăm cu K), iar ca să aflăm eficient prima intrare în v după ce am ieșit din u vom aplica un

upperbound() cu valoarea K în vectorul cu tout-urile din nodul v. Dacă notăm indicele rezultat cu i, fiul căutat va fi chiar cel de pe poziția $i - 1$ din vectorul ciclului eulerian.
Ambele idei intra in complexitate $log(N)$ pentru fiecare query.
Complexitate: $\mathcal{O}\left(Q \cdot log(S)\right)$, unde S = numărul de insule și lacuri de pe hartă.

4.4.1 Coduri sursă

Soluție de 100 puncte cu K-th Ancestor
Soluție de 100 puncte cu ciclul eulerian
Soluție de 45 puncte cu BFS
Soluție de 15 puncte cu sume parțiale
Soluție de 10 puncte