Solutie Flower

O(N*M) – 20 de puncte

O solutie care rezolva fiecare operatie in O(N), prin actualizarea tuturor valorilor mirosului in custi pentru operatiile de tip 1, respectiv cautarea liniara a minimului pentru operatiile de tip 2 obtine 20 de puncte.

O(M*log^2 N) – 100 de puncte

Pentru 100 de puncte trebuie sa facem urmatoarea observatie: adaugarea unui sconcs nou va modifica valorile mirosului intr-un interval compact de custi. Aceasta observatie se face pe baza restrictiei p[x] <= p[x+1].

Detaliind, putem vedea graficul mirosului fiecarui sconcs x ca fiind de forma unui V intors, cu maximul in pozitia (c[x], m[x]), V-ul avand panta p[x] pe partea crescatoare si -p[x] pe panta descrescatoare. Daca p[x]<=p[x+1], atunci V-ul fiecarui sconcs nou adaugat va fi cel putin la fel de "ascutit" ca celelalte. Deci, pentru fiecare parte a V-ului unui sconcs x, odata ce graficul acestuia coboara sub valoarea data de un V al unui sconcs y<x pus anterior, este garantat ca nu va mai putea depasi graficul celuilalt V niciodata.

Pe baza acestei observatii putem tine un arbore de intervale cu lazy update, care mentine pentru fiecare interval urmatoarele date:

valid = 0/1 – daca intervalul este un interval cu panta compacta (1) sau nu (0)
start – daca intervalul este valid, care este valoarea primului element din interval
panta – daca intervalul este valid, care este panta pe interval
mn – care este minimul pe intervalul curent

Cand adaugam un sconcs nou, vom proceda in felul urmator pentru partea stanga a V-ului dat de acesta: cautam binar care este cusca cea mai din stanga pentru care valoarea mirosului data de sconcsul nou adaugat este mai mare decat valoarea anterioara care acesta o va actualiza in timp real. Cand cautam binar, vom putea gasi valorile mirosului in diferite custi facand query-uri pe arborele de intervale. Pentru partea dreapta procedam analog

Pentru a raspunde la operatiile de tip 2, vom face un query pe intervalul [l, r] in arbore.

Complexitatea unei operatii de update este O(log^2 N), iar cea a unei operatii de query este O(log N).

O(M log N) – 100 de puncte.

Putem obtine complexitatea O(M log N), vom mai tine o structura de date care ne va permite sa facem operatiile de update in O(log N). Astfel, vom tine o lista dublu inlantuita, pentru care fiecare element reprezinta un interval compact de pante. In fiecare element vom tine pozitia de start, cea de final, valoarea la inceputul intervalului si panta. Pentru a face un update, facem urmatorii pasi:

1. Cautam intr-un nou arbore de intervale care este elementul listei dublu inlantuite care acopera pozitia pe care adaugam sconcsul.
2. Spargem acest element in doua, pentru a ne fi mai usor cand cautam fiecare parte a V-ului.
3. Pentru partea stanga, eliminam din lista elementele ale caror valori sunt complet sub V-ul sconcsului nou adaugat. Cand gasim un element care nu este complet sub V-ul nostru, cautam binar cea mai din stanga pozitie pentru care V-ul nou-adaugat este deasupra vechii valori. Spargem elementul respectiv la acea pozitie, iar in dreapta lui adaugam partea crescatoare a V-ului noi adaugat. Ultima operatie va fi insotita de un update pe arborele de intervale initial.
4. Analog vom proceda si pentru partea dreapta.