Opisi algoritama

IBT 2010., 4. kolo, juniori P1

1 Show

Pomoću četiri *for* petlje možemo isprobati sve moguće nizove od četiri ocjene iz intervala [1, 10] i ispisati prvi koji zadovoljava uvjete zadatka (razlika svake dvije ocjene je najviše 1 i ocjene odgovaraju strogoći sudaca).

2 Ljestve

Za svaki ton x najprije generiramo njegovu dursku ljestvicu tj. skup svih tonova u njoj; nazovimo ga $S(x)$.

Kada unesemo dani niz tonova, za svaki mogući osnovni ton x treba provjeriti nalaze li se svi tonovi iz danog niza u skupu $S(x)$ - ako da, onda je x traženi ton.

3 Stabalca

Desnu i lijevu stranu koordinatnog sustava rješavamo potpuno odvojeno.

Dvije točke leže na istom pravcu ako im je omjer y/x jednak. Budući da su x, y do 10^{18}, ne možemo samo podijeliti brojeve i usporediti ih jer realna aritmetika na računalima nije dovoljno precizna, stoga trebamo napraviti vlastitu strukturu za razlomke. Od svake točke napravimo razlomak y/x i skratimo ga. Sada je dovoljno sortirati te razlomke najprije po brojniku, a ako ima više takvih onda i po nazivniku. Ovo nam omogućava da jednim prolaskom po nizu prebrojimo koliko ima različitih razlomaka, što je i rješenje zadatka.

Složenost opisanog algoritma iznosi $O(N \log_2 N)$.

4 Informatijada

Za svakog natjecatelja izračunamo njegov najbolji mogući rezultat (koji se postiže ako dotični natjecatelj u sljedeća dva kola ostvari po 300 bodova), kao i njegov najlošiji mogući rezultat (ako dotični natjecatelj u sljedeća dva kola ostvari po 0 bodova).

Najbolji mogući plasman natjecatelja - nazovimo ga Marko - postiže se kada on ostvari svoj najbolji, a ostali svoje najlošije moguće rezultate. Markov najbolji plasman pronalazimo tako da odgovorimo na sljedeće pitanje: koliko je rezultata, u nizu svih najlošijih rezultata, veće od najboljeg Markovog rezultata? To su svi oni koji će pobijediti Marka u najpovoljnijem slučaju.

Taj potproblem rješavamo binarnim pretraživanjem na silazno sortiranom nizu najlošijih rezultata, nalazeći prvi rezultat koji je manji od Markovog najboljeg - svi prethodni su tada sigurno bolji od Marka.

Najlošiji mogući Markov plasman računamo analogno, binarno pretražujući sortirani niz najboljih mogućih rezultata svih natjecatelja s obzirom na Markov najlošiji mogući rezultat.

Naravno, sortirane nizove najboljih i najlošijih rezultata generiramo unaprijed, prije traženja mogućih plasmana natjecatelja. Složenost algoritma je $O(N \log_2 N)$.

5 Jadranka

Cilj je odrediti dubinu stabla, ako svaki vrh postavimo u korijen stabla.

Za rješavanje ovog zadatka stablo ćemo promatrati kao da je vrh 1 korijen stabla.

Promotrimo koji podaci nam sada trebaju da bismo izračunali koja će biti dubina stabla ako je vrh X korijen. Najdalji čvor od čvora X može biti u njegovom podstablu, a i ne mora.

Da bismo riješili slučaj kada je najdalji čvor u podstablu X, za svaki vrh izračunamo dubinu njegovog podstabla (rekurzivno obrađujemo vrhove).

Drugi slučaj, kada je najudaljeniji vrh negdje izvan podstabla, rješavamo tako da u rekurzivnom obilasku stabla pamtimo još jednu vrijednost: udaljenost najudaljenijeg vrha od trenutnog vrha X, izvan podstabla X.

Sada se postavlja pitanje kako stalno održavati tu vrijednost.

Ako smo u čvoru X, te je najudaljeniji vrh izvan njegovog podstabla udaljen V, i želimo izračunati tu udaljenost za jedno njegovo dijete Y, tada postoje dvije mogućnosti za održavanje te vrijednosti:

a) Možemo samo produžiti dosadašnju, odnosno $V + C$ (C je duljina veze $X - Y$)

b) Možemo se popeti od $Y - X$, po duljini C, te onda spustiti do najdubljeg čvora od X; tu vrijednost već smo izračunali. Problem nastaje kada se čvor Y nalazi na putu do najdubljeg čvora od X; stoga moramo još i izračunati za svaki čvor i drugu najveću dubinu, koju ćemo označiti sa $MX2[X]$, a najveću sa $MX1[X]$. Ako je $MX1[X]$ jednako $C + MX1[Y]$ (tj. ako se Y nalazi na najvećoj dubini iz X), tada je nova vrijednost $MX2[X] + C$, a inače je $MX1[X] + C$.

Da sumiramo: ako vrh X postavimo kao korijen stabla, tada je najudaljeniji vrh udaljen $\max\{MX1[X], V\}$, a vrijednost V možemo održavati ako stablo obilazimo rekurzivno. Za implementacijske detalje pogledajte službeno rješenje. Složenost opisanog algoritma iznosi $O(N)$.

Komentare i pitanja uputite na frane.kurtovic@gmail.com *ili na* askurdija@gmail.com.