Checkin

Prof. Emanuela Cerchez, Liceul de Informatică „Grigore Moisil” Iaşi

O primă observaţie: la ghişeu vor sta maxim `N` persoane (nu are rost ca la un ghişeu să stea două persoane). Deci `K=min{K,N}`.

Determinăm timpul minim prin căutare binară în intervalul `[0, TMIN]` (unde `TMIN` este timpul minim care s-ar obţine dacă bagajele ar fi date de o singură persoană, la un singur ghişeu: `Tmin=min{P*a[i]+b[i]|1<=i<=N}`

Pentru a verifica dacă check-in-ul se poate termina în timpul `T`:

Livrăm în ordinea ghişeelor bagajele care pot fi livrate în timpul `T`. Dacă nu reuşim să livrăm toate bagajele în timpul `T` folosind primele `K` ghişee, mărim timpul. În caz contrar, micşorăm timpul.

Pentru fiecare verificare, sortăm ghişeele după `(T-b[i])/a[i]`.

(sau determinăm K maxime)

Complexitate O(`n log N log TMIN`).

volei

Filip Cristian Buruiana, Universitatea Politehnica Bucuresti

Pentru a putea rezolva problema este necesar mai întâi să determinăm dacă pentru o anumită amplasare a fetelor şi băieţilor este posibilă trasarea unei drepte care să separe cele două mulţimi corespunzătoare de puncte.

Propoziţie: Fie `A` şi `B` două mulţimi de puncte şi o dreaptă `d` astfel încât punctele din `A` să fie într-un semiplan, iar punctele din `B` în celalalt semiplan determinat de dreapta `d` (spunem că `A` şi `B` sunt separate de dreapta `d`). Atunci există un punct din `A` şi un punct din `B` astfel încât dreapta care conţine cele două puncte să separe mulţimile `A` şi `B`.

Demonstraţia propoziţiei este intuitivă. Dreapta `d` de separaţie poate fi translatată până când conţine un punct din `A`, iar apoi rotită până când conţine şi un punct din `B`.

În consecinţă, dacă avem două mulţimi de puncte, putem verifica în complexitate $O(N^3)$ dacă mulţimile pot fi separate, alegând toate perechile de puncte (P_1 P_2), unde P_1 este în prima mulţime, iar P_2 în cea de a doua, şi verificând dacă condiţia este îndeplinită pentru dreapta P_1P_2.

Amplasarea iniţială a fetelor şi băieţilor poate fi privită ca un numar în baza 2, unde un bit 0 reprezintă o fată iar un bit 1 un băiat. Trebuie să ajungem în alta configuraţie cu număr minim de mutări astfel încât configuraţia finală să fie validă (cele două mulţimi să fie separabile). Prin mutare se inţelege selectarea a doi biţi `i` şi `j` de valori diferite şi negarea lor (bitul 0 va deveni 1, iar bitul 1 va deveni 0, ceea ce este echivalent cu interschimbarea unei fete cu un băiat). În plus, poziţiile corespunzătoare biţilor interschimbaţi trebuie să fie la distanţa euclidiană maxim `D`.

Pentru a determina numărul minim de mutări, pornim cu o coadă care conţine iniţial doar configuraţia de start. În afară de coadă, vom reţine şi un vector de distanţe, unde elementul al `p`-lea din vector reprezintă distanţa de la configuraţia a `p`-a din coadă la punctul de start. La fiecare pas adăugăm în coadă toţi vecinii configuraţiei curente (configuraţiile care se pot obţine din configuraţia curentă prin exact o mutare). Procedeul se repetă până când în coadă este introdusă o configuraţie validă.

bile

Mugurel Ionuţ Andreica, Universitatea Politehnica Bucureşti

Pentru o valoare fixată a lui x, următorul algoritm de tip **greedy** determină numărul maxim total de bile extrase în timp liniar (O(N)). Prietenul 0 extrage x bile din urna 0. Restul de bile din urna 0 pot fi extrase doar de către prietenul 1, astfel că acesta va extrage cât mai multe dintre aceste bile. Vom parcurge apoi celelalte urne în ordine, de la 1 la N-1, şi pentru fiecare vom proceda după cum urmează: prietenul i va extrage cât mai multe bile posibile din urna i (în limita numărului de bile rămase în urna i şi în limita capacităţii buzunarelor prietenului i). Restul de bile rămase în urna i vor fi extrase de prietenul ((i+1) mod N) (din nou, în limita capacităţii buzunarelor acestuia). Vom nota prin f(x) numărul maxim de bile ce pot fi extrase de toţi prietenii, dacă prietenul 0 extrage exact x bile din urna 0. Întrucât f(x) poate fi calculat în timp O(N) folosind algoritmul descris anterior, am obţinut deja o soluţie de complexitate O(N·XMAX), unde XMAX este numărul de valori pe care le poate lua x (x ia valori de la 0 la xm=min{S_0,P_0}). Această soluţie ar trebui să obţină aproximativ 40 puncte.

Pentru a obţine punctajul maxim, trebuie observat că graficul funcţiei f(x) are o formă particulară. Fie y_1=f(0) şi y_2=f(xm). Pe intervalul [0,A] valoarea funcţiei creşte cu câte 1 unitate (faţă de valoarea anterioară); pe intervalul [A,B] funcţia are valori constante, iar pe intervalul [B,xm], valoarea funcţiei scade cu câte 1 unitate (faţă de valoarea anterioară). Demonstraţia acestui fapt nu este complicată şi nu foloseşte elemente care depăşesc nivelul de cunoştinţe al clasei a X-a.

Aşadar, problema se reduce la a determina în mod eficient valorile A şi B (de menţionat că oricare din intervalele [0,A], [A,B] sau [B,xm] pot avea lungime 0). O modalitate simplă de a calcula aceste valori este să calculăm acea valoare xs, care are proprietatea că y_1+xs=y_2+(xm-xs) (adică acea valoare xs ce ar corespunde cazului în care lungimea intervalului [A,B] ar fi 0). Vom calcula ys=f(xs), iar apoi vom calcula diferenţa d=y_1+xs-ys. Valorile lui A şi B sunt xs-d şi xs+d. Mai există încă un caz, pentru situaţia în care xs nu este un număr întreg, care se tratează în mod similar. În ambele situaţii, valorile A şi B se pot determina folosind un număr constant (O(1)) de evaluări ale funcţiei f. Odată ce am determinat valorile A şi B, putem calcula foarte simplu valoarea f(x) pentru fiecare valoare a lui x. Aşadar, complexitatea soluţiei optime este O(N+XMAX).