

Checkin

Prof. Emanuela Cercez, Liceul de Informatică „Grigore Moisil” Iași

O primă observație: la ghișeu vor sta maxim N persoane (nu are rost ca la un ghișeu să stea două persoane). Deci $K = \min\{K, N\}$.

Determinăm timpul minim prin căutare binară în intervalul $[0, T_{\min}]$ (unde $T_{\min}$ este timpul minim care s-ar obține dacă bagajele ar fi date de o singură persoană, la un singur ghișeu:
 $T_{\min} = \min\{P \cdot a[i] + b[i] \mid 1 \leq i \leq N\}$

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

Livrăm în ordinea ghișeelor bagajele care pot fi livrate în timpul T . Dacă nu reușim să livrăm toate bagajele în timpul T folosind primele K ghișee, mărim timpul. În caz contrar, micșorăm timpul.

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

(sau determinăm K maxime)

Complexitate $O(n \log N \log T_{\min})$.

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 și băieților este posibilă trasarea unei drepte care să separe cele două mulțimi corespunzătoare de puncte.

Propoziție: Fie A și B două mulțimi de puncte și 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 și B sunt separate de dreapta d). Atunci există un punct din A și un punct din B astfel încât dreapta care conține cele două puncte să separe mulțimile A și B .

Demonstrația propoziției este intuitivă. Dreapta d de separație poate fi trasată până când conține un punct din A , iar apoi rotită până când conține și un punct din B .

În consecință, dacă avem două mulțimi de puncte, putem verifica în complexitate $O(N^3)$ dacă mulțimile pot fi separate, alegând toate perechile de puncte (P_1, P_2) , unde P_1 este în prima mulțime, iar P_2 în cea de a doua, și verificând dacă condiția este îndeplinită pentru dreapta P_1P_2 .

Amplasarea inițială a fetelor și băieților poate fi privită ca un număr în baza 2, unde un bit 0 reprezintă o fată iar un bit 1 un băiat. Trebuie să ajungem în alta configurație cu număr minim de mutări astfel încât configurația finală să fie validă (cele două mulțimi să fie separabile). Prin mutare se înțelege selectarea a doi biți i și j de valori diferite și 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, pozițiile corespunzătoare biților interschimbați trebuie să fie la distanța euclidiană maxim D .

Pentru a determina numărul minim de mutări, pornim cu o coadă care conține inițial doar configurația de start. În afară de coadă, vom reține și un vector de distanțe, unde elementul al p -lea din vector reprezintă distanța de la configurația a p -a din coadă la punctul de start. La fiecare pas adăugăm în coadă toți vecinii configurației curente (configurațiile care se pot obține din configurația curentă prin exact o mutare). Procedul se repetă până când în coadă este introdusă o configurație validă.

bile

Mugurel Ionuț Andreica, Universitatea Politehnica București

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 $N-1$, și 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 și în limita capacității buzunarelor prietenului i). Restul de bile rămase în urna i vor fi extrase de prietenul $((i+1) \bmod N)$ (din nou, în limita capacității buzunarelor acestuia). Vom nota prin $f(x)$ numărul maxim de bile ce pot fi extrase de toți 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 obținut deja o soluție de complexitate $O(N \cdot X_{MAX})$, unde X_{MAX} este numărul de valori pe care le poate lua x (x ia valori de la 0 la $x_m = \min\{S_0, P_0\}$). Această soluție ar trebui să obțină aproximativ 40 puncte.

Pentru a obține punctajul maxim, trebuie observat că graficul funcției $f(x)$ are o formă particulară. Fie $y_1 = f(0)$ și $y_2 = f(x_m)$. Pe intervalul $[0, A]$ valoarea funcției crește cu câte 1 unitate (față de valoarea anterioară); pe intervalul $[A, B]$ funcția are valori constante, iar pe intervalul $[B, x_m]$, valoarea funcției scade cu câte 1 unitate (față de valoarea anterioară). Demonstrația acestui fapt nu este complicată și nu folosește elemente care depășesc nivelul de cunoștințe al clasei a X-a.

Așadar, problema se reduce la a determina în mod eficient valorile A și B (de menționat că oricare din intervalele $[0, A]$, $[A, B]$ sau $[B, x_m]$ pot avea lungime 0). O modalitate simplă de a calcula aceste valori este să calculăm acea valoare x_s , care are proprietatea că $y_1 + x_s = y_2 + (x_m - x_s)$ (adică acea valoare x_s ce ar corespunde cazului în care lungimea intervalului $[A, B]$ ar fi 0). Vom calcula $y_s = f(x_s)$, iar apoi vom calcula diferența $d = y_1 + x_s - y_s$. Valorile lui A și B sunt $x_s - d$ și $x_s + d$. Mai există încă un caz, pentru situația în care x_s nu este un număr întreg, care se tratează în mod similar. În ambele situații, valorile A și B se pot determina folosind un număr constant ($O(1)$) de evaluări ale funcției f . Odată ce am determinat valorile A și B , putem calcula foarte simplu valoarea $f(x)$ pentru fiecare valoare a lui x . Așadar, complexitatea soluției optime este $O(N + X_{MAX})$.