Michał Włodarczyk
Treść zadania, Opracowanie

Piotr Smulewicz, Marek Sommer
Program

Dostępna pamięć: 64 MB.

OI, etap I, 6.10–3.11.2014

Łasuchy

Na uroczystej kolacji wieńczącej tegoroczny Zlot Bajtockich Miłośników Słodyczy przy okrągłym stole usiadło n łasuchów. Na stole postawiono n tortów. Torty różnią się między sobą wielkością, wyglądem i smakiem, ale dla łasuchów najważniejszą cechą charakteryzującą tort jest jego kaloryczność (i-ty tort ma kaloryczność c_i). Torty postawiono tak, że pomiędzy każdą parą sąsiadujących łasuchów znajduje się jeden tort. Każdy z łasuchów może wybrać, czy chce jeść tort znajdujący się po jego lewej stronie, czy ten znajdujący się po jego prawej stronie. Jeśli dwóch łasuchów wybierze ten sam tort, to dzielą się nim po połowie.

Każdy łasuch chce zmaksymalizować kaloryczność tortu (lub połówki tortu), który zje. Łasuch będzie niezadowolony, jeśli okaże się, że wybrał źle – czyli zjadłby więcej, gdyby wybrał drugi z dostępnych mu tortów (przy założeniu, że reszta łasuchów nie zmienia swojego wyboru). Pomóż łasuchom dokonać wyboru tak, aby żaden z nich nie był niezadowolony.

Wejście

Pierwszy wiersz standardowego wejścia zawiera liczbę całkowitą n ($2 \leqslant n \leqslant 1\,000\,000$), oznaczającą liczbę łasuchów (i zarazem liczbę tortów). Drugi wiersz zawiera ciąg n liczb całkowitych $c_1, c_2, \ldots, c_n$ ($1 \leqslant c_i \leqslant 1\,000\,000\,000$) pooddzielanych pojedynczymi odstępami; liczba c_i oznacza kaloryczność i-tego tortu. Zakładamy, że i-ty łasuch (dla $1 \leqslant i < n$) może wybrać tort i-ty lub $(i+1)$-szy, natomiast n-ty łasuch może wybrać tort n-ty lub pierwszy.

W testach wartych 50% punktów zachodzi $n \leqslant 1000$, a w podzbiorze tych testów wartym 20% punktów zachodzi $n \leqslant 20$.

Wyjście

Jeśli łasuchy nie mogą wybrać tortów tak, aby każdy z nich był zadowolony, w pierwszym i jedynym wierszu standardowego wyjścia powinno znajdować się jedno słowo `NIE`*. W przeciwnym wypadku, pierwszy i jedyny wiersz standardowego wyjścia powinien zawierać ciąg n liczb całkowitych pooddzielanych pojedynczymi odstępami; i-ta liczba ma oznaczać numer tortu wybranego przez i-tego łasucha. Jeśli jest więcej niż jedna poprawna odpowiedź, Twój program powinien wypisać dowolną z nich.*

Przykład

Dla danych wejściowych:

```
5
5 3 7 2 9
```

poprawnym wynikiem jest:

```
2 3 3 5 1
```

Testy „ocen”:

`1ocen`: *$n = 20$, torty o nieparzystych indeksach mają kaloryczność 7, a torty o parzystych indeksach mają kaloryczność 3; każdy łasuch będzie zadowolony, jeśli wybierze tort o kaloryczności 7;*

`2ocen`: *$n = 1\,000\,000$, kaloryczność każdego tortu jest liczbą wylosowaną z przedziału $[500\,000\,001, 1\,000\,000\,000]$; jeśli wszystkie łasuchy wybiorą tort z prawej strony lub jeśli wszystkie łasuchy wybiorą tort z lewej strony, to każdy z nich będzie zadowolony.*

Rozwiązanie

Zadania na Olimpiadzie Informatycznej można zazwyczaj zaklasyfikować do jednego z czterech rodzajów: zliczanie obiektów, optymalizacja, symulacja wydarzeń lub szukanie strategii wygrywającej w grze dwuosobowej. Zadanie *Łasuchy* odbiega od powyższego kanonu i dlatego na początku nie za bardzo wiadomo, jak się za nie zabrać. Jesteśmy proszeni o zaproponowanie dowolnego przyporządkowania tortów, w którym żaden łasuch nie będzie chciał zmieniać swojego przydziału. Choć na pierwszy rzut oka zagadnienie to wydaje się być trochę oderwane od rzeczywistości, okazuje się, że istnieje cała teoria matematyczna, blisko związana z ekonomią, badająca podobne problemy. Znajomość tej teorii nie jest niezbędna do rozwiązania zadania, niemniej jednak wygodnie będzie nam zaczerpnąć z niej kilka pojęć.

Krótki wstęp do niekooperacyjnej teorii gier

Definicja 1. *Grą* nazywamy obiekt matematyczny, składający się ze:

1. zbioru graczy $N = \{1, 2, \ldots, n\}$,
2. n zbiorów strategii – dla i-tego gracza jest to A_i,
3. rodziny funkcji $u_i : A_1 \times A_2 \times \ldots \times A_n \longrightarrow \mathbb{R}$ dla $i = 1, \ldots, n$, mówiących jakie *wypłaty* dostają gracze w zależności od decyzji podjętych przez wszystkich.

Każdy gracz może wybrać jedną strategię ze swojego zbioru A_i. Celem każdego z nich jest maksymalizacja własnego zysku, jednak gracze mogą sobie wzajemnie w tym przeszkadzać.

Definicja ta została wprowadzona przez słynnego matematyka Johna Nasha[1] i okazała się świetnym narzędziem do modelowania sytuacji konfliktu w ekonomii, naukach społecznych, ruchu drogowym czy sztucznej inteligencji. Najważniejszym narzędziem pomocnym w zrozumieniu mechaniki gry jest pojęcie *równowagi*.

Definicja 2. *Równowagą Nasha (czystą)* nazwiemy takie przyporządkowanie graczom strategii, w którym żadnemu z graczy nie opłaca się zmieniać strategii, przy założeniu, że pozostali gracze również pozostaną przy swoich wyborach.

[1] John Forbes Nash (1928 – 2015) – amerykański matematyk i ekonomista, współlaureat Nagrody Nobla w dziedzinie ekonomii w 1994 roku. Nash cierpiał na schizofrenię paranoidalną, o czym opowiada film *Piękny umysł*.

Oprócz równowagi *czystej* ważne jest też ogólniejsze pojęcie równowagi *mieszanej*, w której pozwalamy graczom na losowanie strategii.

Twierdzenie 1 (Nash, 1951). *Każda gra ze skończoną liczbą graczy i strategii posiada przynajmniej jedną mieszaną równowagę Nasha.*

Sytuacja się komplikuje, kiedy pytamy o istnienie czystej równowagi. Nie dość, że nie każda gra takową posiada (zachęcamy do wymyślenia przykładu – można ograniczyć się do dwóch graczy, z których każdy ma dwie strategie), to rozstrzyganie o jej istnieniu na podstawie funkcji wypłaty jest problemem NP-trudnym, co oznacza, że nie jest znany żaden algorytm odpowiadający na to pytanie w czasie wielomianowym. Oczywiście istnieją gry, w których poszukiwanie czystej równowagi jest prostsze, i na szczęście właśnie z taką grą będziemy mieć do czynienia.

W dalszej części opracowania ograniczamy się tylko do pojęcia równowagi czystej. Więcej o teorii gier można dowiedzieć się z internetowego skryptu `http://mst.mimuw.edu.pl/lecture.php?lecture=wtg`.

Równowaga przy stole

Nasze zadanie sprowadza się do znalezienia czystej równowagi Nasha w grze, w której graczami są łasuchy, każdy z nich ma dwie strategie (wzięcie lewego bądź prawego tortu), a wypłaty zależą od wybranego tortu oraz od decyzji podjętych przez sąsiadów.

Najprostsze rozwiązanie polega na sprawdzeniu wszystkich scenariuszy, jakie mogą wydarzyć się przy stole. Skoro każdy z graczy ma do wyboru dwie strategie, to złożoność obliczeniowa takiego algorytmu wynosi $O(2^n)$, co pozwala na zdobycie maksymalnie 20 punktów.

Spróbujemy innego podejścia. Wybierzmy pewne początkowe przyporządkowanie tortów i sprawdźmy, kto jest niezadowolony. Dopóki przynajmniej jeden gracz będzie niezadowolony, zmieniamy jego strategię, licząc na to, że w końcu dojdziemy do momentu, w którym wszyscy będą zadowoleni.

Taki pomysł może budzić wątpliwości: co w sytuacji, kiedy równowaga nie istnieje? Okazuje się, że autor zadania był trochę złośliwy, bo dla każdych danych wejściowych istnieje przydział, w którym wszyscy są zadowoleni[2]. Fakt ten stanie się oczywisty podczas analizy przedstawionych algorytmów.

No dobrze, ale jak dobrać stan początkowy oraz kolejność poprawiania łasuchów, aby zagwarantować, że po niewielkiej liczbie poprawek dojdziemy do stanu równowagi? Przecież hipotetycznie moglibyśmy wpaść w pętlę podczas zmieniania strategii graczy i napisać program, który nigdy się nie zakończy! Spokojnie, pod koniec opracowania udowodnimy, że nawet w najgorszej implementacji powyższego algorytmu taka sytuacja nie może nigdy mieć miejsca. Zaczniemy jednak od zaprezentowania kilku metod poprawiania stanu gry, które prowadzą do efektywnych rozwiązań.

[2]Na swoje usprawiedliwienie autor twierdzi, że niemożność wymyślenia przykładu z odpowiedzią *NIE* miała skłonić zawodników do zastanowienia się, dlaczego zawsze istnieje poprawny przydział, co mogło naprowadzić ich na trop algorytmu. Tym razem możemy mu wybaczyć.

Rozwiązanie wzorcowe

Pierwszy pomysł na początkowe ustawienie to przypisanie każdemu łasuchowi tortu po jego prawej stronie, czyli i-ty łasuch otrzyma tort o kaloryczności c_{i+1} (ustalamy, że indeksy rosną w prawo; ponadto dla prostoty opisu utożsamiamy tort o numerze $n+1$ z tortem o numerze 1). Zastanówmy się, kto może być niezadowolony przy takim ustawieniu. Jeśli i-ty gracz woli wybrać lewy tort, który musiałby dzielić z lewym sąsiadem, zamiast prawego tortu, którego nie musi współdzielić, to $c_i > 2c_{i+1}$. Zauważmy, że niezależnie od sytuacji w sąsiedztwie, i-ty gracz zawsze będzie preferował lewy tort, więc po pojedynczej zmianie już nigdy nie będziemy musieli poprawiać jego strategii.

Przypuśćmy, że zmodyfikowaliśmy wybór i-tego gracza. Mogło to nie spodobać się jego sąsiadom, którzy teraz również mogą chcieć zmienić swoje preferencje. Jednak jeśli gracz o numerze $i+1$ rezygnuje z całego tortu lub gracz o numerze $i-1$ wybiera obecnie tort, który będzie musiał współdzielić, to znowu żadne przyszłe wydarzenia nie zmienią już ich zdania. Spróbujmy sformalizować to rozumowanie.

Lemat 1. Rozpoczynając od przypisania każdemu łasuchowi tortu po jego prawej stronie i modyfikując wybory niezadowolonych łasuchów, nigdy nie znajdziemy się w sytuacji, w której pewien łasuch chciałby zmienić tort po raz drugi.

Dowód: Indukcja po liczbie graczy, którym zmieniliśmy przydział. Ustalmy niezadowolonego łasucha o numerze i, któremu zmieniamy przydział z prawego tortu na lewy. Rozpatrzymy cztery przypadki opisujące jego sąsiedztwo.

A *Żaden z sąsiadów nie był jeszcze modyfikowany.* Jak wcześniej zauważyliśmy, implikuje to $c_i > 2c_{i+1}$, zatem łasuch zawsze będzie preferował lewy tort.

B *Zmodyfikowaliśmy wcześniej tylko lewego sąsiada.* Zgodnie z założeniem indukcyjnym lewy sąsiad wybiera teraz lewy tort i już nigdy tego nie zmieni, natomiast prawy sąsiad wybiera prawy tort, ale może w przyszłości zmienić decyzję. Wynika z tego, że $c_i > c_{i+1}$ i niezależnie od preferencji prawego sąsiada i-ty gracz zawsze pozostanie przy lewym torcie.

C *Zmodyfikowaliśmy wcześniej tylko prawego sąsiada.* Rozumowanie analogiczne do przypadku B.

D *Obaj sąsiedzi zostali już zmodyfikowani.* Założenie indukcyjne mówi, że obaj sąsiedzi pozostaną przy swoich wyborach, więc nic nie przeszkodzi już i-temu łasuchowi w uczcie.

Pokazaliśmy, że obecnie poprawiany gracz zawsze pozostanie przy swoim nowym torcie, zatem udowodniliśmy tezę indukcyjną, co kończy dowód. ■

Możemy trzymać niezadowolonych łasuchów w kolejce, do której dodajemy łasuchy w momencie, w którym przestaje im odpowiadać ich decyzja. Jako że każdy może trafić do kolejki tylko raz, otrzymujemy rozwiązanie liniowe.

Jeszcze prostszy algorytm dostaniemy, wykorzystując fakt, że każdy początkowo niezadowolony łasuch może być źródłem dwóch ciągów poprawek: w lewo oraz

w prawo. Aby uwzględnić je wszystkie, można najpierw „obejść stół w prawo" sprawdzając zadowolenie kolejnych łasuchów, a następnie w lewo. Uwaga: nie wystarczy przejść od łasucha 1 do łasucha n i z powrotem, ponieważ gra toczy się na okręgu i pominiemy w ten sposób np. ciąg poprawek $(n-3, n-2, n-1, n, 1, 2, 3)$. Na szczęście dwukrotne obejście stołu w każdym z kierunków jest już wystarczające. Takie rozwiązanie zaimplementowano w pliku `las.cpp`.

Obie powyższe implementacje mogły liczyć na 100 punktów, natomiast algorytm sprawdzający łasuchów w kolejności od 1 do n do momentu, aż wszyscy będą zadowoleni, może działać w czasie $\Theta(n^2)$, przez co przechodzi jedynie testy warte 50 punktów.

Rozwiązanie alternatywne

Inny pomysł to rozpoczęcie od przypisania każdemu łasuchowi tortu o wyższej kaloryczności (w przypadku remisu wybieramy dowolny). Co ciekawe, tym razem wystarczy sprawdzić każdego łasucha tylko raz (w dowolnej kolejności) i zmienić strategię niezadowolonych. Poniższy lemat gwarantuje, że taka procedura doprowadzi do stanu równowagi.

Lemat 2. Rozpoczynając od przypisania każdemu łasuchowi tortu o wyższej kaloryczności, wystarczy poprawić strategię każdego łasucha co najwyżej raz, w dowolnej kolejności.

Dowód: Chcemy pokazać, że zmiana przydziału dla gracza o numerze i nie popsuje zadowolenia już sprawdzonych łasuchów. Przypuśćmy bez straty ogólności, że $c_i \geqslant c_{i+1}$, czyli omawiany łasuch początkowo wybierał lewy tort. Skoro obecnie jest on z niego niezadowolony, to oznacza, że i-ty tort jest współdzielony, zaś tort o numerze $i+1$ nie ma obecnie żadnego amatora. Zatem obaj jego sąsiedzi muszą wybierać torty po swojej prawej.

Kiedy zmienimy decyzję i-tego gracza, oba najbliższe torty będą przypisane pojedynczym graczom. Wypłata gracza $i-1$ zwiększy się dwukrotnie, natomiast gracz $i+1$, którzy nie był zainteresowany kalorycznością c_{i+1}, tym bardziej nie skusi się na kaloryczność $\frac{c_{i+1}}{2}$. Wobec tego potencjalna zmiana strategii i-tego gracza nie wpłynie na wcześniejsze ustalenia i po poprawieniu wszystkich niezadowolonych łasuchów znajdziemy się w stanie równowagi. ■

Implementację tego rozwiązania można znaleźć w pliku `las2.cpp`.

Dalsza analiza

Co jeżeli nie byliśmy wystarczająco sprytni i zaczęliśmy od innego przyporządkowania początkowego? W dalszym ciągu mamy szansę na przejście kilku testów, ponieważ nasz niekoniecznie efektywny algorytm zawsze znajdzie pewną czystą równowagę Nasha. Aby to pokazać, wprowadzimy przydatne pojęcie *potencjału*.

Definicja 3. Dla ustalonego przydziału tortów p, jego *potencjałem* nazwiemy wielkość

$$\Phi(p) = \sum_{i=1}^{n} \alpha_i^p c_i, \tag{1}$$

gdzie α_i^p zależy od p i równa się:

0: jeśli i-ty tort nie został nikomu przydzielony,

1: jeśli dokładnie jeden gracz chce zjeść i-ty tort,

$\frac{3}{2}$: jeśli dwóch graczy zdecydowało się na i-ty tort.

Przedstawiona definicja może wydawać się „wyciągnięta z kapelusza", chociażby ze względu na pojawiającą się znikąd stałą $\frac{3}{2}$. Pochodzenie wzoru (1) staje się jednak jasne w dowodzie kolejnego lematu.

Lemat 3. Jeśli przydział q otrzymujemy z p poprzez zmianę decyzji pewnego gracza, to wypłata tego gracza wzrasta o $\Phi(q) - \Phi(p)$.

Dowód: Przypuśćmy, że interesujący nas gracz zmienia swój tort z i-tego na j-ty. Z definicji ciągu (α_i^p) wynika, że jego nowa wypłata równa się $(\alpha_j^q - \alpha_j^p)c_j$, zaś poprzednia wynosiła $(\alpha_i^p - \alpha_i^q)c_i$. Jako że dla pozostałych tortów wartości α nie zmieniają się, to różnica pomiędzy tymi wypłatami równa się $\Phi(q) - \Phi(p)$. ■

Powyższa obserwacja pozwala na prosty dowód zapowiedzianego wcześniej twierdzenia.

Twierdzenie 2. *Algorytm modyfikujący decyzje niezadowolonych łasuchów zawsze znajduje pewną czystą równowagę Nasha po skończonej liczbie kroków.*

Dowód: Oznaczmy ciąg przydziałów generowany w trakcie działania ustalonego algorytmu przez $p_1, p_2, \ldots$ Zmiana decyzji niezadowolonego gracza prowadzi do wzrostu jego wypłaty, toteż z lematu 3 wnioskujemy, że dla każdego i zachodzi $\Phi(p_{i+1}) > \Phi(p_i)$.

Dodatnia różnica potencjałów nie może być mniejsza niż $\frac{1}{2}$, więc gdyby wygenerowany ciąg przydziałów był nieskończony, to ciąg $(\Phi(p_i))_{i=1}^{\infty}$ byłby nieograniczony. Nie jest to możliwe, ponieważ potencjał nie może przekroczyć $\frac{3}{2}\sum_{i=1}^{n} c_i$. Zatem algorytm w pewnym momencie znajduje przydział, w którym wszyscy są zadowoleni, i kończy działanie. ■

W praktyce taki algorytm mógł zdobyć sporo punktów, jeśli zaczynał od losowego przydziału i efektywnie znajdował niezadowolonych łasuchów.

Teoria gier a algorytmika

Funkcja potencjału wygląda na zbyt eleganckie narzędzie, by używać go tylko do badania przydziałów tortów. W rzeczywistości zostało ono wprowadzone do badania tzw. *routing games*, modelujących m.in. ruch drogowy albo przesyłanie pakietów danych po sieci. Każdy z uczestników takiej gry ma za zadanie przesłać pewien ładunek pomiędzy dwoma wierzchołkami grafu, a koszt przesyłania ładunku wzdłuż krawędzi zależy od tego, jak mocno jest ona eksploatowana przez innych graczy.

Oczywiście każdy gracz stara się minimalizować tylko własny koszt (albo własny czas przesyłu). Badanie równowagi Nasha w takich grach doprowadziło do odkrycia m.in. paradoksu Braessa, czyli przykładu sieci, w której dodanie nowej krawędzi w grafie prowadzi do pogorszenia sytuacji każdego z graczy. Co ciekawe, nie jest to jedynie teoretyczna koncepcja. Naukowcy zajmujący się zagadnieniami transportu drogowego wielokrotnie zaobserwowali sytuację, w której zamknięcie drogi w dużym mieście paradoksalnie doprowadziło do poprawienia przepustowości w całej sieci drogowej.

Analiza *routing games* jest częścią stosunkowo nowej teorii naukowej zwanej *algorytmiczną teorią gier*.

Jeszcze inne rozwiązania

Inny wniosek z lematu 3 jest taki, że każdy przydział maksymalizujący funkcję potencjału jest równowagą Nasha. Taki przydział można znaleźć przy pomocy programowania dynamicznego w czasie $O(n)$, co daje nam kolejne efektywne rozwiązanie.

Możliwe jest też bardziej bezpośrednie wykorzystanie programowania dynamicznego. Niech $i \geqslant 3$ oraz $t_i[A, B, C] = 1$, jeśli możliwe jest zadowolenie wszystkich łasuchów z przedziału $[2, i-1]$, gdy A koduje decyzję pierwszego łasucha, natomiast B, C – decyzje łasuchów o numerach $i-1, i$. W przeciwnym przypadku $t_i[A, B, C] = 0$. Dla każdego i musimy wyznaczyć tylko 8 wartości. Łatwo zauważyć, że możemy to zrobić, znając wartości $t_{i-1}[\cdot, \cdot, \cdot]$. Tak obliczona tablica t wystarcza do odtworzenia całego rozwiązania.

Testy

Testy zostały podzielone na trzy rodzaje:

1. Rosnący (lub malejący) ciąg tortów, w którym nie ma pary sąsiednich tortów, takich że jeden jest co najmniej dwukrotnie bardziej kaloryczny od drugiego. Wyjątek stanowi sytuacja obok pierwszego oraz ostatniego tortu. Całość jest przesunięta cyklicznie o losową wartość.

2. Wiele segmentów o kalorycznościach postaci $[x, x+1, x+2, \ldots]$ przedzielonych bardzo kalorycznymi tortami.

3. Testy całkowicie losowe.