Informatyczny kącik olimpijski (58): Dwa przyjęcia

W niedawno wydanej książce *W poszukiwaniu wyzwań* – zbiorze zadań z konkursów programistycznych – Filip Wolski opisał rozwiązanie zadania *Dwa przyjęcia* z finału XII Olimpiady Informatycznej. W zadaniu tym występuje n osób, z których niektóre się znają (wiemy które). Chcemy podzielić ten zbiór na dwa rozłączne podzbiory (przyjęcia) w taki sposób, aby zmaksymalizować liczbę osób, które mają parzystą liczbę znajomych na przyjęciu, na którym przebywają. Przedstawiony w książce algorytm jest efektem indukcyjnego rozumowania o strukturze grafu zbudowanego na bazie relacji znajomości (krawędzie) pomiędzy osobami (wierzchołki) i pokazuje, że zawsze da się podzielić zbiór osób na takie dwie grupy, że każda osoba ma parzystą liczbę znajomych wewnątrz swojej grupy.

Wiedząc o tym, że szukany podział zawsze istnieje, można to zadanie rozwiązać zupełnie inaczej, bez stosowania teorii grafów. Zauważmy, że w zasadzie mamy do czynienia jedynie z wartościami binarnymi: są dwa przyjęcia, każdy gość musi trafić albo na pierwsze, albo na drugie z nich, wreszcie interesuje nas tylko parzystość liczby znajomych, a nie jej dokładna wartość. Sprowadzimy zatem oryginalny problem do znalezienia rozwiązania pewnego układu równań w ciele $\mathbb{Z}_2$.

Zaczniemy od przypisania każdej osobie niewiadomej x_i o następującym znaczeniu:

$$x_i = \begin{cases} 1 & \text{jeśli } i\text{-ta osoba przebywa na pierwszym przyjęciu,} \\ 0 & \text{jeśli } i\text{-ta osoba przebywa na drugim przyjęciu.} \end{cases}$$

Przyjmijmy na chwilę, że i-ta osoba ma znajomych o numerach $j_1, j_2, \ldots, j_q$ oraz że jest ich parzyście wielu (tj. $2 \mid q$). Przyjrzyjmy się takiemu oto równaniu:

$$x_{j_1} + x_{j_2} + \ldots + x_{j_q} \equiv 0 \pmod 2. \tag{1}$$

Jeśli znajdziemy wartościowanie niewiadomych $\{x_{j_1}, \ldots, x_{j_q}\}$ spełniające to równanie, to wtedy i-ta osoba będzie miała tak na pierwszym, jak i na drugim przyjęciu parzystą liczbę znajomych. W przeciwnym przypadku na obu przyjęciach będzie nieparzysta liczba znajomych i-tej osoby.

Dla osób o nieparzystej liczbie znajomych będziemy musieli lekko zmodyfikować nasze rozumowanie. Jeśli bowiem dla takiej osoby (x_i) zbudujemy analogiczne równanie

$$x_{j_1} + x_{j_2} + \ldots + x_{j_q} \equiv 0 \pmod 2,$$

to spełniające je wartościowanie niewiadomych $\{x_{j_1}, \ldots, x_{j_q}\}$ będzie oznaczało, że i-ta osoba ma parzyście wielu znajomych na pierwszym przyjęciu i nieparzyście wielu na drugim. Jeśli zamiast zera postawilibyśmy po prawej stronie równania jedynkę, uzyskalibyśmy analogiczną sytuację: i-ta osoba ma parzyście wielu znajomych na drugim przyjęciu i nieparzyście wielu na pierwszym. W każdym przypadku i-ta osoba może „przypadkowo” znaleźć się na przyjęciu z nieparzystą liczbą swoich znajomych.

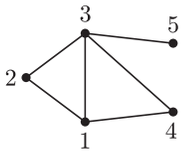
Aby poradzić sobie z osobami o nieparzyście wielu znajomych, zastosujemy pewną sztuczkę – włączymy takie osoby do równań z ich znajomymi:

$$x_i + x_{j_1} + x_{j_2} + \ldots + x_{j_q} \equiv 1 \pmod 2. \tag{2}$$

Żeby upewnić się, że taki pomysł ma sens, rozpatrzmy dwa przypadki:

1. Na pierwszym przyjęciu znajduje się parzyście wielu znajomych i-tej osoby. Wtedy, oczywiście, jest ich nieparzyście wielu na drugim przyjęciu, a zatem chcemy sprawić, aby i-ta osoba znalazła się na pierwszym przyjęciu. Ale w takiej sytuacji powyższe równanie będzie spełnione wtedy i tylko wtedy, gdy $x_i = 1$, czyli jest dobrze.
2. Na pierwszym przyjęciu znajduje się nieparzyście wielu znajomych i-tej osoby, a na drugim parzyście wielu. Wtedy chcemy wysłać tę osobę na drugie przyjęcie, więc $x_i = 0$ i równanie też zostanie spełnione.

Mamy już wszystkie składniki potrzebne do rozwiązania zadania. Konstruujemy układ n równań – po jednym dla każdej osoby. Jeśli i-ta osoba ma parzyście wielu znajomych, to równanie jest postaci (1), a w przeciwnym przypadku postaci (2). Na przykład:



$$\left(\begin{array}{ccccc|c} x_1 & + x_2 & + x_3 & + x_4 & & 1 \\ x_1 & & + x_3 & & & 0 \\ x_1 & + x_2 & & + x_4 & + x_5 & 0 \\ x_1 & & + x_3 & & & 0 \\ & & x_3 & & + x_5 & 1 \end{array}\right)$$

Do rozwiązywania układów równań liniowych służy algorytm eliminacji Gaussa. Zwykle jednak stosuje się go do „standardowych” układów równań w ciele liczb rzeczywistych, a nie w arytmetyce modularnej. Kluczowymi operacjami w eliminacji Gaussa są działania na wierszach rozszerzonej macierzy reprezentującej układ równań: mnożenie wiersza przez skalar oraz dodawanie wierszy. Za ich pomocą sprowadzamy macierz najpierw do postaci trójkątnej górnej z jedynkami na przekątnej, a następnie pozbywamy się wartości z górnej części macierzy, co daje nam rozwiązanie układu.

Aby przystosować eliminację Gaussa do działania w ciele $\mathbb{Z}_p$ (czyli w arytmetyce modularnej o podstawie p, gdzie p jest liczbą pierwszą), musimy umieć wykonywać analogiczne operacje. Zazwyczaj nie jest to duży problem: dodając wiersze, musimy jedynie pamiętać o braniu reszty z dzielenia przez p. Mnożenie przez skalar działa tak samo. Należy jednakże pamiętać, że w liczbach rzeczywistych mnożenie zwykle służy temu, aby doprowadzić do znalezienia się na przekątnej liczby 1, więc często mnożymy przez jakiś ułamek (*de facto* wykonujemy dzielenie). W arytmetyce modularnej o podstawie p odpowiednikiem takiego działania będzie przemnożenie x przez taką liczbę x^{-1}, że $x \cdot x^{-1} \equiv 1 \pmod p$. W znajdowaniu takich wartości może pomóc np. rozszerzony algorytm Euklidesa.

Na szczęście nasze zadanie jest dużo prostsze, wszak działamy w $\mathbb{Z}_p$ dla $p = 2$. W związku z tym nigdy nie wykonamy mnożenia ani dzielenia (gdyż jedynym niezerowym skalarem w $\mathbb{Z}_2$ jest jedynka). Czas i pamięć potrzebne na zbudowanie układu równań są rzędu $O(n^2)$. Łączna złożoność czasowa algorytmu wynosi jednak $O(n^3)$, bowiem w takim czasie działa eliminacja Gaussa.

Bartosz SZREDER
doktorant, Wydział Matematyki, Informatyki i Mechaniki,
Uniwersytet Warszawski