Jest to zapis odczytu pod tym samym tytułem wygłoszonego na XXXV Szkole Matematyki Poglądowej, *Porządek*, sierpień 2005.

Działanie programu, do którego doszedł na koniec Autor, zaopatrzonego w odpowiednie spowolnienia, było demonstrowane na ekranie. Program ten można sobie ściągnać ze strony Ośrodka Kultury Matematycznej.

1729

Jarosław WRÓBLEWSKI, Wrocław

Wedle znanej powszechnie anegdoty, pewnego razu Hardy przyjechał do Ramanujana taksówką. Na powitanie powiedział:
– Jechałem taksówką o numerze bocznym 1729. To bardzo nieciekawa liczba, mam nadzieję, że to nie wróży nic złego.

Na to Ramanujan bez namysłu odrzekł:
– Wprost przeciwnie! Przecież 1729 to najmniejsza liczba rozkładająca się na sumę dwóch sześcianów na dwa sposoby.

Zastanówmy się, jak dla ustalonej liczby N w miarę efektywnie, przy pomocy komputera znaleźć rozwiązania równania

$$a^3 + b^3 = c^3 + d^3 \tag{1}$$

w liczbach całkowitych dodatnich $a,\ b,\ c,\ d \le N$, spełniających nierówności natury porządkowej

$$a \ge b \quad \text{oraz} \quad a > c \ge d. \tag{2}$$

Pomysł pierwszy (fatalny, ale od czegoś trzeba zacząć)

Przeglądamy wszystkie czwórki liczb (a, b, c, d) niewiększych od N spełniające warunek (2) i sprawdzamy, czy spełniają równanie (1). Takich czwórek jest $\binom{\binom{N+1}{2}}{2}$, czyli około $N^4/8$, a więc liczba operacji potrzebnych do wykonania programu rośnie wraz z czwartą potęgą zakresu poszukiwań. Można przy tym zaniedbać fakt, że wraz ze wzrostem N rośnie też czas wykonania pojedynczej operacji programu, gdyż operujemy na większych liczbach. W zakresie zmian N, który może nas interesować w praktyce, nie jest to bowiem istotne.

Pomysł drugi (o wiele lepszy, ale to też jeszcze nie jest to)

Przeglądamy trójki liczb (a, b, c) i sprawdzamy, czy liczba $a^3 + b^3 - c^3$ jest sześcianem liczby całkowitej dodatniej. Sprawdzenie, czy liczba x jest sześcianem, można wykonać na wiele różnych sposobów. Możemy stablicować sześciany liczb całkowitych w interesującym nas zakresie i za każdym razem sprawdzać, czy liczba x znajduje się w tablicy. Albo możemy badać reszty z dzielenia liczby x przez liczby pierwsze postaci $6n + 1$. Albo możemy obliczyć $\sqrt[3]{x}$ i sprawdzić, czy jest to liczba całkowita.

Nie będziemy opisywać szczegółów realizacji powyższych idei.

W każdym razie opisany algorytm będzie działał w czasie z grubsza proporcjonalnym do N^3, a więc znacznie szybciej niż w pomyśle pierwszym. Tu od razu się asekuruję przed specjalistami w branży: dla mnie $N^3 \log N$ to też „z grubsza N^3”.

Pomysł trzeci (to już prawie to, o co chodzi)

Tablicujemy wszystkie pary liczb (a, b) spełniających warunek $N \ge a \ge b$ wraz z sumami $a^3 + b^3$ i szukamy różnych par z tą samą sumą szescianów.

Pierwsza myśl: posortować pary według sum. Ale sortowanie jest dość kosztowne, a nas nie interesuje uporządkowanie sum sześcianów, a jedynie znalezienie powtórzeń.

Zastanówmy się, co byśmy zrobili, gdyby doszła do nas plotka, że podczas egzaminu pisemnego, który się właśnie zakończył, jeden ze studentów oddał 2 egzemplarze pracy, każda podpisana jego nazwiskiem. Wszystkie zebrane prace są dokumentnie pomieszane.

Możemy posortować prace według nazwisk. Ale to wydaje nam się zbyt czasochłonne.

Możemy rozdzielić prace na tysiące stosów. Dla każdego istniejącego w Polsce nazwiska utworzymy stos zawierający prace podpisane tym nazwiskiem. Fajny

pomysł, wymaga tylko paru hektarów powierzchni na utworzenie tysięcy stosów, z których prawie wszystkie pozostaną puste. Może zastosujemy go innym razem ;-)

Możemy wreszcie utworzyć stosy w liczbie z grubsza równej liczbie oddanych prac. Musimy mieć pewien deterministyczny algorytm przypisujący nazwiska do stosów w sposób, na pierwszy rzut oka, losowy. Chodzi o to, aby prace rozłożyły się na stosach w miarę równomiernie. Będzie trochę stosów pustych, trochę stosów z jedną pracą. Bez trudu sprawdzimy, czy nie powtarza się nazwisko na pracach w stosach zawierających więcej niż jedną pracę, bo stosy te nie będą zbyt liczne.

Zastosowanie powyższego pomysłu do zagadnienia równych sum sześcianów przybiera następującą postać:

dla ustalonego zakresu N naszych poszukiwań dobieramy liczbę pierwszą q; najlepiej, aby jej rząd wielkości był $N^2/2$, ale mogą zaistnieć przesłanki skłaniające nas do nieco innego wyboru.

Tworzymy w pamięci komputera (pomijamy programistyczną stronę zagadnienia) q stosów o umownych numerach od 0 do $q-1$. Wszystkie pary liczb (a, b), gdzie

$$N \geq a \geq b > 0,$$

będziemy zapisywać w pamięci, przy czym para (a, b) zostanie zapisana na stosie o numerze r, jeżeli r jest resztą z dzielenia liczby a^3+b^3 przez q.

Zapisując parę (a, b) na r-tym stosie, sprawdzamy, czy był on do tej pory pusty. Jeśli nie, porównujemy liczbę a^3+b^3 z liczbami c^3+d^3 dla wszystkich par (c, d), które już znajdują się na tym stosie. Jeżeli $a^3 + b^3 = c^3 + d^3$, drukujemy rozwiązanie. Zauważmy, że w ten sposób otrzymamy wszystkie rozwiązania równania (1) spełniające warunki (2).

Czas działania takiego programu jest z grubsza proporcjonalny do N^2. Ma on jednak pewną poważną wadę. Program potrzebuje bardzo dużo pamięci komputera na jednoczesne przechowywanie wszystkich par (a, b).

Przy $N = 17000$ program potrzebuje prawie 1700 MB pamięci RAM, a jego czas działania to zaledwie 2 minuty.

Chcąc zwiększyć zakres poszukiwań do $N = 34000$ musielibyśmy czekać na wynik 8 minut (żaden problem), ale musielibyśmy użyć prawie 7 GB pamięci RAM komputera, a to, jak na razie, nie jest powszechnie osiągalne.

Jak obejść tę trudność nie płacąc przy tym dramatycznym wydłużeniem czasu pracy programu?

Pomysł czwarty (ten właściwy)

Skoro nie możemy umieścić w pamięci naraz wszystkich par (a, b), podzielmy zbiór tych par na rozłączne podzbiory tak, aby spełnione były poniższe warunki:

(i) każdy podzbiór mieści się naraz w pamięci komputera,

(ii) jeśli $a^3 + b^3 = c^3 + d^3$, to pary (a, b) i (c, d) znajdują się w tym samym podzbiorze,

(iii) dla każdego podzbioru umiemy w miarę szybko wygenerować wszystkie pary liczb, które do niego należą.

Jakie ma więc być kryterium, wedle którego przypiszemy zbiór wszystkich par do poszczególnych podzbiorów?

Najbardziej naturalnym pomysłem jest umieszczanie w pamięci takich par (a, b), że liczba $a^3 + b^3$ mieści się w ustalonym przedziale. Jednak praktyczne próby wcielenia w życie tego pomysłu pokazują, że warunek (iii) jest nie najlepiej spełniony.

Dużo wydajniejszy jest pomysł następujący:

Ustalmy liczbę p. Najlepiej, jeśli p jest liczbą pierwszą postaci $6n + 5$.

Dla każdej liczby R, gdzie $0 \leq R < p$, umieścimy w pamięci komputera te pary (a, b), dla których

$$a^3 + b^3 \equiv R \pmod{p}.$$

Oto bliższy opis realizacji tego pomysłu:

Wybieramy liczbę pierwszą p postaci $6n + 5$. Liczba ta powinna być na tyle mała, na ile pozwoli nam dostępna pamięć komputera. Jednak pewne subtelności sprzętowo-programistyczne mogą czasami skłonić nas do wykorzystania tylko części pamięci.

Przed rozpoczęciem właściwych poszukiwań przygotowujemy tabelę T, dzięki której będziemy mogli obliczać pierwiastki sześcienne modulo p. Dokładniej, tabela ma p wierszy, a w r-tym wierszu znajdują się wszystkie liczby całkowite dodatnie $b \leq N$, dla których

$$b^3 \equiv r \pmod{p}.$$

W tym celu dla każdej liczby b obliczamy $b^3 \pmod{p}$, co mówi nam, w którym wierszu tabeli T należy umieścić liczbę b.

Dla każdej liczby R, gdzie $0 \leq R < p$, generujemy wszystkie pary (a, b) spełniające warunki $N \geq a \geq b$ oraz $a^3 + b^3 \equiv R \pmod{p}$. W tym celu dla każdej liczby a obliczamy $r = R - a^3 \pmod{p}$, a następnie w r-tym wierszu tabeli T znajdujemy odpowiednie wartości b, pamiętając, aby $a \geq b$.

Aha... powinniśmy też byli zawczasu wybrać liczbę q i zainicjować q stosów o numerach od 0 do $q-1$. Podobnie jak w poprzedniej wersji programu, para (a, b) zostanie zapisana na stosie o numerze $a^3 + b^3 \pmod{q}$. Podobnie jak poprzednio, zapisując parę (a, b) na stosie, sprawdzamy, czy był on do tej pory pusty. Jeśli nie, porównujemy liczbę a^3+b^3 z liczbami c^3+d^3 dla wszystkich par (c, d), które już znajdują się na tym stosie. Jeżeli $a^3 + b^3 = c^3 + d^3$, drukujemy rozwiązanie.