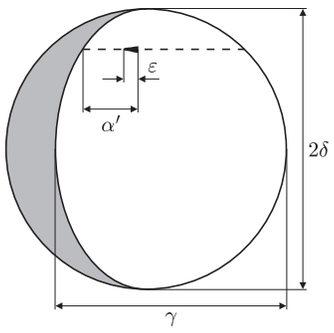

Rys. 5. Kąty, które należy zmierzyć, aby wyznaczyć wysokość wzniesienia. Linia przerywana łączy punkty jednakowo odległe od płaszczyzny wyznaczonej przez położenie obserwatora, środka Księżyca i środka Słońca.

Dzięki temu do wyznaczenia wysokości wzniesienia nie jest potrzebna znajomość kątowej skali obrazu Księżyca obserwowanego w lunecie bądź kątowej skali fotografii Księżyca. Otrzymane zależności są, oczywiście, prawdziwe dla dowolnego kulistego obiektu oświetlonego odległym, niemal punktowym źródłem światła.

* * *

Jeśli R uznamy za wielkość znaną, to do wyznaczenia wysokości wzniesienia konieczne będzie zmierzenie czterech kątów: ε, α', γ, δ (rys. 5). Można je zmierzyć w trakcie bezpośredniej obserwacji wizualnej lub wykorzystując do pomiarów zdjęcie Księżyca.

Zrobienie zdjęcia Księżyca, umożliwiającego wykonanie niezbędnych pomiarów z rozsądną dokładnością, wymaga użycia obiektywu o ogniskowej zbliżonej do 1 m. Rolę takiego obiektywu spełnia zazwyczaj obiektyw lunety lub zwierciadło teleskopu. Decydujący wpływ na dokładność pomiaru ma wielkość i ostrość obrazu Księżyca. Ze względu na drgania układu fotografującego powodowane powiewami wiatru i turbulencją atmosferyczną czas naświetlania nie powinien przekraczać 1/30 sekundy. Ponieważ jedną z mierzonych wielkości jest promień tarczy Księżyca, a wielkość tę można wyznaczyć najdokładniej, mierząc średnicę tarczy, zdjęcie powinno obejmować całą oświetloną część tarczy.

Bezpośredni (wizualny) pomiar kątów ε, α', γ, δ będzie wymagał użycia lunety lub teleskopu umożliwiającego osiągnięcie ponad stokrotnego powiększenia. Typując wzniesienia przewidziane do pomiaru, należy wybierać takie, których otoczenie wydaje się w miarę płaskie i poziome. Jedyną wskazówką, umożliwiającą ocenę stopnia spełnienia tego warunku, jest światłocieniowy obraz otoczenia. Jeśli zależy nam na zmierzeniu wysokości konkretnego wzniesienia, należy poczekać na wieczór, w którym znajdzie się ono w pobliżu terminatora.

Rozwiązanie zadania M 1353.
Zauważmy, że

$$\frac{1}{n} = \frac{1}{n-1} - \frac{1}{n(n-1)},$$

ale

$$\frac{1}{k} - \frac{1}{l+1} = \left(\frac{1}{k} - \frac{1}{k+1}\right) + \left(\frac{1}{k+1} - \frac{1}{k+2}\right) + \ldots + \left(\frac{1}{l} - \frac{1}{l+1}\right) = \frac{1}{k(k+1)} + \frac{1}{(k+1)(k+2)} + \ldots + \frac{1}{l(l+1)},$$

zatem wystarczy przyjąć $k = n-1$, $l = n(n-1)-1$.

Test na liczbę pierwszą

*Wojciech CZERWIŃSKI**

Chyba wszyscy lubimy liczby pierwsze. Szczególne wrażenie robią te naprawdę duże, wydają się skrywać w sobie jakąś nadzwyczajną tajemnicę: dlaczego akurat one stały się swego rodzaju wybrańcami spośród innych liczb i mają tak niezwykłe właściwości?

Matematycy od dawna zastanawiają się, jak sprawdzać, czy liczba jest pierwsza. Dawniej robili to tylko (a może aż) z ciekawości i poczucia doniosłości zadania. W dzisiejszych czasach mają także bardziej praktyczne motywacje. Przykładowo, na potrzeby algorytmu RSA chcielibyśmy umieć sprawdzać, czy liczba mająca około 500 cyfr jest pierwsza, czy też złożona.

Więcej szczegółów o algorytmie RSA i o teście Millera–Rabina można przeczytać m.in. w książkach *Wprowadzenie do algorytmów* T. Cormena, R. Leisersona, R. Rivesta i C. Steina oraz *Kryptografia. W teorii i w praktyce* D. Stinsona.

Powszechnie stosowany i w praktyce najszybszy jest test pierwszości Millera–Rabina (w skrócie test MR), wymyślony w 1980 roku. Wykorzystuje on losowość i opiera się na następującym pomyśle. Powiedzmy, że chcemy sprawdzić, czy liczba n jest pierwsza. Okazuje się, że jeżeli n jest złożona, to co najmniej połowa liczb ze zbioru $\{1, 2, \ldots, n-1\}$ jest *świadkami złożoności* tej liczby (to zresztą bardzo mało dokładne oszacowanie). Nazwa bierze się stąd, że jeżeli dla pewnej liczby n istnieje liczba $1 \leqslant a \leqslant n-1$ będąca jej świadkiem złożoności, to wiadomo, że n jest liczbą złożoną.

Stosunkowo łatwo jest sprawdzić, czy dana liczba a jest świadkiem złożoności dla n, ale nie będziemy teraz wchodzić w szczegóły, co to znaczy i jak się to robi. Co więcej, odpowiednie obliczenia można wykonać szybko – jeżeli liczba n ma k cyfr, to algorytm sprawdzający, czy a jest świadkiem złożoności dla n, wykonuje mniej więcej k^3 operacji. To znaczy, że dla n będącej liczbą pięćsetcyfrową wykona on mniej więcej 500^3, czyli 125 milionów operacji. Nasz domowy komputer potrzebuje na to mniej więcej jednej setnej sekundy – czyli nie jest źle.

*doktorant, Instytut Informatyki, Uniwersytet Warszawski

Zatem aby sprawdzić, czy liczba n jest pierwsza, postępujemy tak: losujemy a z przedziału od 1 do $n-1$ i sprawdzamy, czy a jest świadkiem złożoności dla n. Jeżeli jest, to wiemy, że n jest złożona. Jeśli nie, to n przeszła pierwszą próbę i sprawdzamy dalej. Ponieważ dla dowolnego n świadkowie złożoności to przynajmniej połowa liczb z przedziału od 1 do $n-1$, więc prawdopodobieństwo, że liczba złożona n przejdzie pojedynczą próbę, jest nie większe niż $1/2$. Losujemy więc kolejne a i znów sprawdzamy, czy jest ono świadkiem złożoności dla n. Jeżeli liczba n jest złożona, to prawdopodobieństwo przejścia 30 testów bez znalezienia świadka złożoności jest mniejsze niż $1/2^{30}$ (czyli również mniejsze niż $1/10^9$). Po 100 testach prawdopodobieństwo tego, że błędnie uznamy liczbę złożoną za pierwszą, jest już mniejsze niż szansa na to, że w ciągu następnej minuty wielki meteoryt uderzy w Ziemię i zakończy wszelkie nasze matematyczne dywagacje. Czyli mamy pewność zupełnie wystarczającą. Przeprowadzenie 100 testów zajmie na domowym komputerze nie więcej niż sekundę – to bardzo niewiele.

Matematykom jednak problem testowania pierwszości nadal nie dawał spokoju. Chcieli mieć metodę dającą pewność absolutną, obliczającą wynik w sensownym czasie, a nie tylko wystarczającą do celów praktycznych.

Powiemy, że algorytm jest *wielomianowy*, jeżeli dla danych rozmiaru k wykonuje co najwyżej $W(k)$ operacji, gdzie W to pewien wielomian. Czyli, na przykład, test MR jest algorytmem wielomianowym, gdyż dla liczby o k cyfrach wykonuje co najwyżej $C \cdot k^3$ operacji dla pewnej stałej C. Algorytmy wielomianowe są umownie uważane za szybkie, a te o większej złożoności – za wolne. W informatyce teoretycznej dla danego problemu bardzo ważnym pytaniem jest, czy da się skonstruować algorytm wielomianowy rozwiązujący go, który podaje poprawną odpowiedź zawsze, a nie tylko z dużym prawdopodobieństwem. Dla problemu testowania pierwszości pytanie to do niedawna było problemem otwartym. Test MR nie jest rozwiązaniem, ponieważ zawiera element losowości.

Zwróćmy uwagę, że jeśli liczba n ma k cyfr, to k jest rzędu $\log n$. Zatem test MR wykonuje *de facto* rzędu $\log^3 n$ operacji. Jednak naturalną miarą wielkości liczby danej na wejściu jest właśnie liczba jej cyfr. Dlatego wymagamy, by algorytm wielomianowy rozwiązujący problem testowania pierwszości wykonywał rzędu $\log^i n$ operacji; n^i to za dużo.

Chociaż przez wiele lat nie udawało się znaleźć odpowiedzi, większość matematyków wierzyła w istnienie wielomianowego testu pierwszości. Aż wreszcie w roku 2002 trzech naukowców z Uniwersytetu w Kampurze ogłosiło znalezienie pierwszego takiego algorytmu. Nazwano go algorytmem AKS, od pierwszych liter nazwisk twórców: Agrawal, Kayal, Saxena. To był wielki sukces – kwestia trudności testowania pierwszości została ostatecznie rozstrzygnięta. Autorzy otrzymali za to osiągnięcie m.in. nagrodę Gödla – najbardziej prestiżowe wyróżnienie w dziedzinie informatyki teoretycznej. Co ciekawe, test AKS działa w praktyce wolniej niż test MR. Po pewnych poprawkach dla liczby o k cyfrach wykonuje rzędu k^6 operacji, a oryginalny algorytm jest nawet jeszcze nieco wolniejszy. To nie umniejsza jednak doniosłości wyniku.

Opis algorytmu mieści się na zaledwie kilku stronach, co jest zupełnie niespotykane dla osiągnięć matematycznych tej trudności. Co więcej, używa (po paru poprawkach) jedynie elementarnej matematyki. Opiera się na kilku bardzo sprytnych spostrzeżeniach. Korzystając z okazji, przyjrzyjmy się temu algorytmowi z lotu ptaka. Głównym pomysłem jest następujące spostrzeżenie.

Lemat. *Niech $a, n \in \mathbb{N}$, $n \geqslant 2$ oraz $\mathrm{NWD}(a,n) = 1$. Wówczas n jest pierwsza wtedy i tylko wtedy, gdy*

$$(x+a)^n \equiv x^n + a \pmod{n}. \tag{1}$$

Dowód. Kongruencję z lematu rozumiemy jako równość modulo n współczynników przy odpowiednich potęgach x^i. W wyrażeniu $(x+a)^n$ współczynnik przy x^i to $\binom{n}{i}a^{n-i}$. Wystarczy więc zastanowić się, kiedy n dzieli $\binom{n}{i}$ dla $1 \leqslant i \leqslant n-1$ oraz czy $a^n \equiv a \pmod{n}$. Gdy n jest pierwsza, to łatwo sprawdzić, że liczby $\binom{n}{i}$ oraz $a^n - a$ są podzielne przez n (ten ostatni fakt to małe twierdzenie Fermata), więc teza jest prawdziwa.

Przypuśćmy teraz, że n jest złożona i liczba pierwsza q dzieli n. Niech m będzie takim maksymalnym wykładnikiem, że $q^m \mid n$. Wówczas jednak mamy

$$q^m \nmid \binom{n}{q} = \frac{n \cdot (n-1) \cdot \ldots \cdot (n-q+1)}{q!},$$

gdyż q dzieli licznik w m-tej potędze, a mianownik w pierwszej. Liczba q jest względnie pierwsza z a, czyli $q \nmid \binom{n}{q}a^{n-q}$, a zatem $n \nmid \binom{n}{q}a^{n-q}$, co dowodzi tezy również dla liczb złożonych. □

Rozwiązanie zadania M 1352.
Wykażemy najpierw, że

$$\sum_{k=i}^{i+(n-r)} \binom{k}{i}\binom{n-k}{r-i} = \binom{n+1}{r+1}.$$

Rozważmy w tym celu $(r+1)$-elementowe podzbiory zbioru $\{1,\ldots,n+1\}$, dla których $t_{i+1} = k+1$. Wtedy $i \leqslant k \leqslant n-(r-i)$. Zatem elementy $t_1,\ldots,t_i$ i $t_{i+2},\ldots,t_{r+1}$ możemy wybrać na $\binom{k}{i}\binom{n-k}{r-i}$ sposobów. Sumując te możliwości po dozwolonych wartościach k, dostajemy żądany wzór.

Rozważmy teraz r-elementowe podzbiory zbioru $\{1,\ldots,n\}$. Takich podzbiorów, dla których $t_i = k$, jest $\binom{k-1}{i-1}\binom{n-k}{r-i}$, gdzie $i \leqslant k \leqslant n-r+i$. Zatem średnia arytmetyczna liczb t_i wynosi

$$\frac{1}{\binom{n}{r}} \sum_{k=i}^{n-r+i} k\binom{k-1}{i-1}\binom{n-k}{r-i}.$$

Ale $\binom{k-1}{i-1} = \frac{i}{k}\binom{k}{i}$, więc ta średnia jest równa

$$\frac{i}{\binom{n}{r}} \sum_{k=i}^{n-r+i} \binom{k}{i}\binom{n-k}{r-i} =$$

$$= \frac{i}{\binom{n}{r}}\binom{n+1}{r+1} = i \cdot \frac{n+1}{r+1}.$$

By rozstrzygnąć, czy n jest pierwsze, wystarczy więc sprawdzić, czy wielomiany $(x+a)^n$ i x^n+a mają wszystkie współczynniki przystające modulo n. Gdybyśmy jednak sprawdzali je po kolei przy każdej potędze x, to wykonalibyśmy przynajmniej rzędu n operacji. To za dużo, więc trzeba obliczać współczynniki sprytnie.

Kolejny pomysł to sprawdzanie równości (1) modulo wielomian postaci x^r-1 dla pewnego $r\in\mathbb{N}$, czyli

$$(x+a)^n \equiv x^n + a \pmod{n, x^r-1}. \tag{2}$$

Może się jednak okazać, że niektóre liczby złożone spełniają równość (2), mimo że nie spełniają równości (1). Rozwiązaniem tego problemu jest sprawdzanie (2) dla odpowiednio dobranego r i dla wszystkich dodatnich wartości a, nie większych niż $\lfloor\sqrt{\phi(r)}\log n\rfloor$ (gdzie $\phi(r)$ oznacza liczbę liczb mniejszych od r i względnie pierwszych z r). Okazuje się, że liczba n, spełniająca wszystkie takie równości, musi być pierwsza. Właśnie w wykazaniu prawdziwości tego spostrzeżenia leży główna trudność techniczna dowodu.

Równość wielomianów $W(x)$ i $V(x)$ modulo pewien wielomian $Z(x)$ oznacza, że reszty z dzielenia wielomianów W i V przez wielomian Z są takie same. Zapis

$$(x+a)^n \equiv x^n + a \pmod{n, x^r-1}$$

interpretujemy tak: najpierw obliczamy reszty $r_1(x)$ i $r_2(x)$ z dzielenia wielomianów $(x+a)^n$ i x^n+a przez x^r-1, a następnie porównujemy współczynniki wielomianów r_1 i r_2 – sprawdzamy, czy są przystające modulo n. Inaczej, możemy sprawdzić, czy reszta z dzielenia wielomianu $(x+a)^n - x^n - a$ przez x^r-1 jest wielomianem, którego wszystkie współczynniki są podzielne przez n.

Niech $o_r(n)$ będzie taką najmniejszą liczbą k, że $n^k \equiv 1 \pmod r$. Algorytm AKS działa według następującego schematu:

1. jeśli $n = a^b$ dla $a\in\mathbb{N}$, $b>1$, to zwróć ZŁOŻONA
2. znajdź taką najmniejszą liczbę r, że $o_r(n) > \log^2 n$
3. jeśli $1 < \mathrm{NWD}(a,n) < n$ dla jakiegoś $a \leqslant r$, to zwróć ZŁOŻONA
4. jeśli $n \leqslant r$, to zwróć PIERWSZA
5. dla a od 1 do $\lfloor\sqrt{\phi(r)}\log n\rfloor$ wykonuj:
   jeśli $(x+a)^n \not\equiv x^n + a \pmod{n, x^r-1}$, to zwróć ZŁOŻONA
6. zwróć PIERWSZA

Bez wielkiej trudności można stwierdzić, że jeśli algorytm zakończy działanie w jednym z pierwszych pięciu kroków, to odpowiedź jest poprawna. Jeżeli zwróci wynik w punkcie 1, to zrobi, oczywiście, słusznie. Jeżeli znajdzie w punkcie 3 taką liczbę $k = \mathrm{NWD}(a,n)$, że $1<k<n$, to k jest nietrywialnym dzielnikiem n, czyli istotnie n jest złożona. Z kolei jeżeli w punkcie 4 okaże się, że $n\leqslant r$, to znaczy, że w punkcie 3 sprawdziliśmy wszystkie $a\leqslant n$ i każda była względnie pierwsza z n, czyli faktycznie n jest pierwsza. Jeśli w punkcie 5 równość modulo n nie jest spełniona, to nie zachodzi również zależność (1), czyli istotnie n jest złożona. Pozostaje jedna wątpliwość co do poprawności algorytmu: czy wynik zwracany w punkcie 6 jest poprawny?

Dowód opiera się na analizie funkcji f oraz liczb $m\in\mathbb{N}$ spełniających równanie

$$f(x)^m \equiv f(x^m) \pmod{x^r-1, p}$$

dla pewnego konkretnego czynnika pierwszego p liczby n. Jeśli liczba n przeszła testy w punkcie 5, to dla każdej funkcji postaci $f(x)=x+a$ oraz $m=n$ powyższa kongruencja jest prawdziwa. Można wyprowadzić analogiczne zależności dla innych funkcji f i liczb m i, idąc tym tropem, udowodnić, że algorytm słusznie zwraca wynik PIERWSZA w punkcie 6. To rozumowanie pozwala zakończyć dowód poprawności algorytmu, ale wymaga trochę pracy nad szczegółami.

Czytelnik Uważny chciałby pewnie zapytać, czy istotnie algorytm AKS jest wielomianowy. Tę sprawę rozwiązuje lemat, z którego wynika, że dla $n>2$ istnieje r nie większe niż $\lceil\log^5 n\rceil$ spełniające własność z drugiego kroku algorytmu. Analizując dokładnie dalsze kroki (do czego przydaje się pewne doświadczenie w szacowaniu złożoności), zauważymy, że ten fakt rzeczywiście pozwala na oszacowanie czasu działania algorytmu przez wielomian zależny od $\log n$. Dobrym zadaniem na początek jest znalezienie sposobu wykonania pierwszego kroku w czasie wielomianowym od $\log n$.

Czytelnik zainteresowany szczegółami bez problemu znajdzie w Internecie artykuł autorów algorytmu *Primes is in P*, napisany naprawdę przystępnie i przejrzyście.

Te wyniki nie oznaczają bynajmniej, że cała sprawa liczb pierwszych doczekała się ostatecznego zamknięcia. Istnieje naprawdę ogromnie dużo nierozwiązanych problemów dotyczących liczb pierwszych. Wiele z nich ma także znaczenie praktyczne. Chyba najważniejszym takim problemem jest zagadnienie faktoryzacji liczb, czyli znajdowania ich rozkładu na czynniki pierwsze. Do dziś nie ma dla tego problemu żadnych algorytmów działających w zadowalającym czasie, choćby takich, które wykorzystują losowość, albo nawet takich, które wydają się poprawne, ale na razie nie umiemy tego udowodnić.