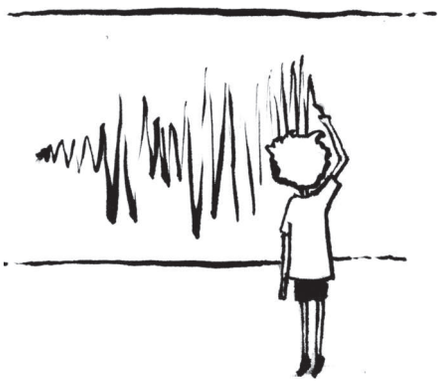
Rozpatrzmy zbiory

$$A = \{1 < i \leqslant n : a_{i-1} < 2p \leqslant a_i\} \quad \text{i} \quad B = \{1 \leqslant i < n : a_i \geqslant 2p > a_{i+1}\}.$$

Zauważmy, że $|A| - |B| = 1$, gdyż $a_1 < 2p$ i $a_n > 2p$. Doprowadzimy do sprzeczności, dowodząc, że jeśli $m > 2$ i p jest wystarczająco duże, to zbiór B jest dużo większy niż A.



Jak wykazali Lagarias, Rains i Sloane, liczby pierwsze pojawiają się w ciągu EKG w kolejności rosnącej, czyli przed pozycją n nie pojawiła się liczba pierwsza większa niż p. Jeśli q jest liczbą sterującą wyrazu $a_{n'}$ i $n' \leqslant n$, to q jest kandydatem na wyraz $a_{n'}$, więc pojawia się w ciągu na pozycji n' lub wcześniejszej. Wnioskujemy stąd, że wszystkie liczby sterujące do pozycji n są mniejsze niż p. Z podanego przed chwilą „kluczowego spostrzeżenia" wiemy, że dla każdej liczby pierwszej $q < p$ istnieje co najwyżej jeden taki indeks i, że $a_{i-1} < 2p \leqslant a_i$ i q jest liczbą sterującą a_i. Liczb pierwszych nie większych niż p jest nie więcej niż $cp/\log p$ dla pewnej stałej uniwersalnej c, czyli

$$|A| \leqslant cp/\log p.$$

Spójrzmy teraz na zbiór B. Niech q będzie liczbą sterującą wyrazu $a_n = mp$; oczywiście q jest dzielnikiem m. Wobec tego liczby kq dla $1 \leqslant k \leqslant (mp/q - 1)$ musiały pojawić się w ciągu wcześniej. Oznaczmy

$$B' = \{1 \leqslant i < n : 2q \mid a_i \text{ oraz } a_i > 2p\}.$$

Indeksy wszystkich parzystych wyrazów postaci kq (podzielnych przez 4, jeśli $q = 2$), większych od $2p$, należą do B', wobec tego

$$|B'| \geqslant \left\lfloor \frac{mp - 2p}{2q} \right\rfloor \geqslant \left\lfloor \frac{p}{6} \right\rfloor.$$

Z drugiej strony, jeśli $i \in B'$, to $2p$ jest potencjalnym kandydatem na wyraz a_{i+1}, czyli $a_{i+1} < 2p$. Mamy więc $B' \subseteq B$, zatem

$$|B| \geqslant \lfloor p/6 \rfloor.$$

Wobec tego dla odpowiednio dużych p mamy $|B| > |A|$.

Wyliczywszy dokładnie stałe w powyższym rozumowaniu, można dowieść, że $|B| > |A|$ już dla $p > 25\,000$. Teza dla mniejszych wartości p została sprawdzona numerycznie przez Lagariasa, Rainsa i Sloane'a, więc hipoteza została udowodniona dla wszystkich $p > 2$. Dowód zakończony.

Jak wyznaczać wyrazy ciągu EKG?

Jakub RADOSZEWSKI

Po przeczytaniu artykułu Marcina Pilipczuka trudno nie odnieść wrażenia, że nasz zasób wiedzy o zachowaniu ciągu EKG opiera się przede wszystkim na wynikach eksperymentów komputerowych, natomiast dowody otrzymanych w ten sposób hipotez pojawiają się z pewnym opóźnieniem. Co więcej, niektóre własności nie doczekały się jeszcze ścisłych dowodów, choć dane doświadczalne pokazują je na tyle wyraźnie, że ciężko byłoby nie wierzyć w ich prawdziwość. Widzimy tu, że informatyka zaczyna pełnić funkcję eksperymentalnej poddziedziny matematyki. Co ciekawe jednak, nie chodzi tu tylko o łatwiejsze stawianie hipotez – wyniki obliczeń mogą stanowić fragmenty *dowodu* odkrytych za ich pomocą własności! W przypadku ciągu EKG przejawiło się to w automatycznym sprawdzeniu prawdziwości hipotezy dla pewnej liczby początkowych wyrazów ciągu (dla pozostałych zadziałało rozumowanie czysto matematyczne), ale zdarzały się już w historii problemy, dla których cały dowód opierał się na komputerowej analizie przypadków – na przykład dowód faktu, że każdą mapę (graf planarny) można pokolorować czterema kolorami, tak aby żadne dwa sąsiednie państwa (wierzchołki) nie miały tego samego koloru. Niektórzy twierdzą, że takie rozumowanie, oparte w znacznej mierze na programach komputerowych i często obszernych wynikach ich działania, nie jest właściwie żadnym *dowodem*. Jest to argument poniekąd słuszny – ciężko ręcznie zweryfikować poprawność tego typu wywodu. Ten medal ma jednak dwie strony. Otóż przypomnijmy sobie chociażby liczne „dowody" Wielkiego Twierdzenia Fermata, publikowane bez mała przez trzy stulecia, które nie używały żadnych podejrzanych metod informatycznych, a mimo wszystko okazywały się nieprawdziwe (niepełne). Czy autorzy tych „dowodów"

wierzyli w słuszność przedstawianych wyników? Podejrzewam, że tak. Swoją drogą, zapewne większość Czytelników wierzy w prawdziwość dowodu Andrew Wilesa, mimo iż jest on tak długi i skomplikowany, że nie mieli ochoty go samodzielnie sprawdzić. Jeśli dołożymy do tego podobną konkurencję rozgrywaną w obecnych czasach – próby odpowiedzi na słynne pytanie: P = NP?, jedno z fundamentalnych w informatyce teoretycznej, których to prób – „dowodów" – były już dziesiątki (zainteresowany Czytelnik znajdzie ich systematycznie aktualizowaną listę na stronie `http://www.win.tue.nl/~gwoegi/P-versus-NP.htm`) – to zauważymy, że czysto matematyczne rozumowania wcale nie muszą być lepsze i mniej błędogenne od rozumowań opartych na wynikach badań eksperymentalnych.

W duchu powyższego, trochę przydługawego wstępu w tym artykule zastanowimy się, jak efektywnie wyznaczać kolejne wyrazy ciągu EKG. Nasz algorytm powinien być na tyle sprawny, aby dało się za jego pomocą sprawdzić poprawność wyników eksperymentu Lagariasa, Rainsa i Sloane'a, czyli powinien działać co najmniej dla indeksów około 50 000 (w tych okolicach powinna pojawić się największa liczba pierwsza mniejsza od 25 000). A im lepszy algorytm, tym lepiej pozwoli nam badać asymptotykę ciągu EKG – a nuż uda się postawić kolejne ciekawe hipotezy?

Początek nie będzie zbyt odkrywczy: otóż metoda wyznaczania wyrazów ciągu EKG (a_n) jest obecna *implicite* w tekście Marcina Pilipczuka. Faktycznie, jeśli znamy a_{n-1}, to wiemy, że jeden z dzielników pierwszych tej liczby jest liczbą sterującą wyrazu a_n. Jeśli właśnie liczba pierwsza q jest tą liczbą sterującą, to a_n jest najmniejszą wielokrotnością q niewystępującą wcześniej w ciągu. Aby wśród wszystkich dzielników pierwszych a_{n-1} zidentyfikować liczbę q, wystarczy, rzecz jasna, przejrzeć wszystkie te dzielniki i wybrać ten o najmniejszej pierwszej wolnej wielokrotności.

Aby uzupełnić ten opis postępowania, musimy wymyślić sposób efektywnego przejrzenia wszystkich dzielników pierwszych danej liczby naturalnej oraz sposób znalezienia najmniejszej wolnej wielokrotności każdego takiego dzielnika. Teraz zajmiemy się kolejno tymi dwiema lukami.

Część Czytelników miała szansę spotkać się z obszerną klasą problemów NP-zupełnych, które są powszechnie uznawane za nierozwiązywalne w czasie wielomianowym, a do tego mają tę zabawną cechę, że gdyby choć jeden z nich udało się rozwiązać w czasie wielomianowym, to wszystkie pozostałe również. Jednakże o problemie faktoryzacji nie wiemy, czy znajduje się w tej klasie, co oznacza mniej więcej tyle, że *tym bardziej* nie wiemy, czy da się go rozwiązać wielomianowo, czy też nie.

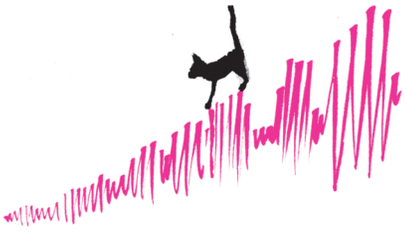
Problem rozkładu liczby naturalnej na czynniki pierwsze brzmi bardzo klasycznie: wydaje się, że powinien istnieć jakiś sympatyczny, efektywny algorytm dający sobie z nim radę. I faktycznie, jest to znany problem *faktoryzacji* liczby, lecz, niestety, w ogólności nie umiemy sobie z nim łatwo radzić – nie jest znany żaden wielomianowy algorytm rozwiązujący ten problem. Słowo *wielomianowy* oznacza tu algorytm o złożoności czasowej będącej jakimś wielomianem względem długości zapisu rozkładanej liczby, która to długość zapisu to mniej więcej logarytm z tej liczby.

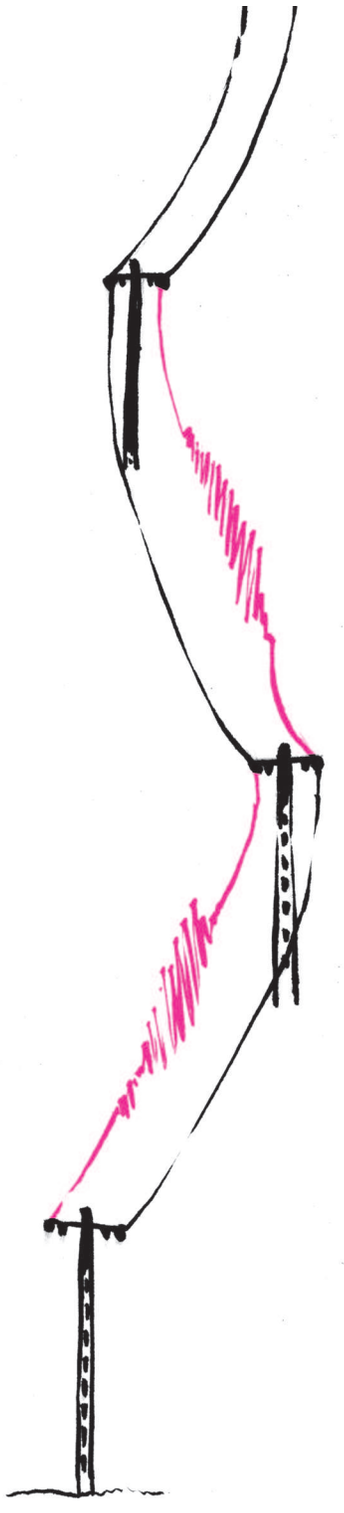
Na szczęście, w naszej sytuacji nie musimy uciekać się do ogólnych rozwiązań problemu faktoryzacji. Wystarczy, jeśli przypomnimy sobie, że rozkładane przez nas liczby nie są zbyt duże, co w teorii wynika z górnego oszacowania $14n$ na wartość n-tego wyrazu ciągu, a w praktyce z wykresu wartości pierwszych 10 000 elementów ciągu zawartego w artykule Marcina Pilipczuka.

W przypadku tak niedużych liczb możemy użyć metody *sita Eratostenesa*. Podstawowa wersja tego algorytmu, służąca do wyznaczania kolejnych liczb pierwszych, jest na tyle znana, że w tym miejscu ograniczymy się do przypomnienia jej w postaci pseudokodu:

```
for i := 2 to M do pierwsza[i] := true;
for i := 2 to M do
  if pierwsza[i] then begin
    j := i^2;
    while j ⩽ M do begin
      pierwsza[j] := false;
      j := j + i;
    end;
  end;
```

Obliczamy tu elementy boolowskiej tablicy $pierwsza[2..M]$, wykreślając wielokrotności kolejnych liczb pierwszych, tj. oznaczając je jako liczby złożone. Żeby było szybciej, przeglądanie wielokrotności liczby pierwszej i rozpoczynamy





od i^2, gdyż skądinąd wiemy, że wszystkie mniejsze musiały już zostać wcześniej wykreślone. Na końcu mamy zaznaczone (wartość **true**) wszystkie liczby pierwsze nie większe niż M.

Zapewne nie wszyscy wiedzą, że ten sam algorytm można wykorzystać także do rozkładania liczb na czynniki pierwsze. Wystarczy, mianowicie, dołożyć dodatkową tablicę $dzielnik[2\,..\,M]$, w której będziemy przechowywać najmniejszy dzielnik pierwszy każdej liczby złożonej z zakresu. Elementy tej tablicy inicjujemy na zera, a wypełniamy ją w momencie wykreślania liczb, tzn. wewnątrz pętli **while**, za pomocą instrukcji:

$$\textbf{if } dzielnik[j] = 0 \textbf{ then } dzielnik[j] := i;$$

Za pomocą tablicy *dzielnik* możemy już łatwo faktoryzować liczby nieprzekraczające M: dla danej liczby i jako pierwszy podajemy dzielnik $dzielnik[i]$, następnie $dzielnik[i/dzielnik[i]]$ i tak dalej, aż dojdziemy do jedynki. Takich kroków wykonamy nie więcej niż $\log_2 i$, gdyż za każdym razem i zmniejszamy co najmniej dwukrotnie. Dzięki temu, że w tablicy pamiętamy *najmniejsze* dzielniki pierwsze liczb, uzyskana w tym postępowaniu lista dzielników będzie zawsze niemalejąca, więc łatwo usuniemy z niej powtórzenia.

Stosując podobny argument (do tego z logarytmem), można wykazać, że złożoność czasowa sita Eratostenesa to $O(M \log M)$, co pozostawiamy Czytelnikowi (tak naprawdę złożoność tę można oszacować nawet przez $O(M \log\log M)$). Jeśli przypomnimy sobie, iż $M = O(n)$, przy czym n to indeks wyrazu ciągu EKG, który chcemy obliczyć, to słusznie zauważymy, że z problemem faktoryzacji poradziliśmy już sobie całkiem sprawnie.

Do zakończenia opisu całego algorytmu pozostała nam już tylko kwestia wyznaczania najmniejszych wolnych wielokrotności poszczególnych liczb pierwszych z rozkładu. Z tą częścią rozwiązania możemy poradzić sobie całkiem prosto, o ile zastosujemy odpowiednio sprytną strukturę danych, przy czym nie będzie to wcale struktura wysublimowana.

Znów skorzystamy z tego, że zakres interesujących nas liczb pierwszych jest nieduży. Dla każdej liczby pierwszej p nie większej niż M będziemy przechowywać, jako $wiel[p]$, jakąś wielokrotność liczby p, która wystąpiła już w ciągu. Zadbamy o to, aby w algorytmie jedynie powiększać te wartości, aktualizując je tylko w razie potrzeby. Początkowo $wiel[p] = 0$. Dodatkowo, w tablicy boolowskiej $użyte[0\,..\,M]$ będziemy pamiętali informację o tym, które liczby wystąpiły już w ciągu EKG. Początkowo, dla wygody, ustawiamy $użyte[0] =$ **true**, a resztę komórek na **false**. Okazuje się, że to już wystarczy – poniżej pseudokod wyznaczania najmniejszej wolnej wielokrotności liczby p, w którym, w gruncie rzeczy, nic się nie dzieje.

$$\textbf{while } użyte[wiel[p]] \textbf{ do } wiel[p] := wiel[p] + p;$$

Powyższa pętla może i wygląda bardzo nieoptymalnie, ale nie jest w rzeczywistości taka zła. Otóż wystarczy zauważyć, że dana liczba i, $i \leqslant M$, może wystąpić jako $wiel[p]$ dla nie więcej niż $\log_2 i$ liczb pierwszych p (będą to, oczywiście, dzielniki pierwsze liczby i). To oznacza, że łączna liczba obrotów powyższej pętli **while** z pewnością nie przekroczy $O(M \log M)$.

Możemy już podsumować cały algorytm wyznaczania n-tego wyrazu ciągu EKG. Używamy w nim zaledwie czterech tablic rozmiaru co najwyżej $M + 1$, przy czym dwie z nich (*pierwsza* i *dzielnik*) wypełniamy na samym początku w czasie $O(M \log M)$, a dwie pozostałe (*wiel* i *użyte*) uzupełniamy w trakcie obliczania kolejnych wyrazów ciągu EKG, również w łącznym czasie $O(M \log M)$. W takim samym czasie możemy wyznaczyć wszystkie dzielniki pierwsze wszystkich liczb z podanego zakresu. Stąd całe rozwiązanie działa w czasie $O(M \log M)$ i w pamięci $O(M)$, przy czym $M = O(n)$. Algorytm szybki, a zarazem prosty – i o to w tym sporcie chodzi.

Można by jednak spytać, czy nie da się lepiej. W szczególności, w podanej metodzie przy wyznaczaniu a_n musimy najpierw obliczyć wszystkie wcześniejsze wyrazy ciągu EKG – a może dałoby się obejść szybciej i bez tego? Podobnie jak Lagarias, Rains i Sloane, którzy w swojej pracy opisują algorytm zbliżony do powyższego, autor tego artykułu nie zna odpowiedzi na to pytanie, ale może Czytelnik ma jakiś pomysł? Jeśli tak, zachęcamy do podzielenia się nim z Redakcją.

Rozwiązanie zadania F 787.
Kondensator opisany w zadaniu jest układem dwóch połączonych równolegle kondensatorów o pojemnościach $C_1 = \varepsilon_0 S_1/d = \varepsilon_0 S(l - l_0)/(dl_0)$ i $C_2 = \varepsilon\varepsilon_0 S_2/d = \varepsilon\varepsilon_0 Sl/(dl_0)$, gdzie ε jest względną przenikalnością elektryczną dielektryka, a $S_{1,2}$ oznaczają pola okładek kondensatorów składowych, $S = S_1 + S_2$. Zatem

$$C = C_1 + C_2 = \varepsilon_0 \frac{[l_0 + l(\varepsilon - 1)]}{dl_0} S.$$