Czy losowe bity pomagają?

*Marcin PILIPCZUK**

* Wydział Matematyki, Informatyki i Mechaniki, Uniwersytet Warszawski

Nagroda Abela to, obok medalu Fieldsa, jedno z największych wyróżnień w świecie matematyki. Przyznawana corocznie od 2002 roku stanowi matematyczny odpowiednik Nagrody Nobla. W 2021 roku nagrodę tę otrzymali László Lovász i Avi Wigderson za fundamentalny wkład w rozwój informatyki teoretycznej i matematyki dyskretnej oraz wiodącą rolę w przekształceniu tych dziedzin w kluczowe dyscypliny matematyki współczesnej. O ile pewnie większość Czytelników spotkała się z bardzo użytecznymi wynikami pierwszego laureata – choćby tzw. lokalnym lematem Lovásza – dorobek drugiego laureata wydaje się mniej znany. W tym krótkim artykule naszkicuję jeden z najciekawszych kierunków badań, w którym istotną rolę odegrał Avi Widgerson, mianowicie *derandomizację* algorytmów.

Zacznijmy od znanego wielu Czytelnikom przykładu problemu *testu pierwszości*: mając daną liczbę naturalną n, chcemy sprawdzić, czy jest ona pierwsza, czy złożona. Od lat 70. ubiegłego wieku znamy kilka prostych i wydajnych testów używających losowości; w pewnym uproszczeniu sprowadzają się one do sprawdzania, czy zachodzi teza Małego Twierdzenia Fermata dla n i losowo wybranej podstawy $1 \leqslant a < n$, tj. czy a^{n-1} daje resztę 1 z dzielenia przez n.

Przez ponad dwadzieścia lat od powstania tych metod nie znaliśmy jednak *deterministycznego* testu, tj. algorytmu, który w czasie wielomianowym od rozmiaru wejścia (tj. od $\lceil \log_2(n+1) \rceil$, bo liczba n na wejściu podana jest w zapisie binarnym) rozstrzyga, czy n jest pierwsze, czy złożone, bez używania bitów losowych. Dopiero w 2002 roku udało się *zderandomizować* test pierwszości: powstał algorytm deterministyczny działający z grubsza w tym samym czasie (wielomianowym od rozmiaru wejścia).

Drugim, trochę mniej znanym przykładem problemu, dla którego istnieje naturalny algorytm korzystający z losowości, jest *test równości wielomianów* (*Polynomial Identity Testing*, PIT): mając dane dwa wielomiany wielu zmiennych, pytamy, czy są one równe. Przyjmujemy tutaj, że wielomiany dane są jako obwody arytmetyczne i są wielomianami nad ciałem skończonym. W tym przypadku test z użyciem losowości jest jeszcze łatwiejszy: jeśli wejściowe wielomiany są różne (i rozmiar ciała jest istotnie większy od stopni obu wielomianów), to lemat Schwartza-Zippela mówi, że z dużym prawdopodobieństwem wartości obu wielomianów dla losowego argumentu są różne, co jest jednoznacznym dowodem, że wielomiany też są różne. Do dziś jednak nie umiemy satysfakcjonująco zderandomizować tego testu.

Istotny dla dalszej opowieści będzie jeszcze problem szacowania wagi obwodu (*Circuit Approximation Problem*, CAP): mając dany obwód logiczny C o N wejściach i jednym wyjściu, oszacuj – z dokładnością do 10% – dla jakiej frakcji możliwych wejść obwód daje wynik 1. Innymi słowy, jeśli potraktujemy C jako funkcję $C : \{0,1\}^N \to \{0,1\}$, to chcemy podać liczbę c spełniającą

$$\left| \frac{|\{y \in \{0,1\}^N : C(y) = 1\}|}{2^N} - c \right| \leqslant 10\%.$$

Tutaj znów algorytm korzystający z losowości jest prosty i naturalny. Wylosujmy pewną (dość niewielką, np. 1000) liczbę argumentów y i obliczmy, dla jakiej frakcji wylosowanych argumentów y mamy $C(y) = 1$; pokazanie, że z dużym prawdopodobieństwem wynik naszych obliczeń jest liczbą bliską $|\{y \in \{0,1\}^N : C(y) = 1\}|/2^N$, jest elementarnym ćwiczeniem z rachunku prawdopodobieństwa. I znów, nie wiemy, jak rozwiązać problem CAP deterministycznie w czasie wielomianowym.

Generatory pseudolosowe

Naiwnym pomysłem na derandomizację algorytmu korzystającego z bitów losowych jest uruchomienie go na wszystkich możliwych kombinacjach bitów losowych. To rozwiązanie jest zazwyczaj bardzo kosztowne: np. w przypadku problemu PIT, sprowadza się to do przetestowania równości wartości wielomianów dla wszystkich argumentów.

Rozwiązanie zadania F 1039.
Filmowcy będą rejestrowali przebieg zdarzenia, w kórym wszystkie odległości będą k razy mniejsze niż w rzeczywistości, ale przyspieszenie g pozostanie niezmienione. Rzeczywisty spadek o odcinek s będzie odpowiadał spadkowi o $s_m = s/k$ modelu pociągu. Mamy: $s = \frac{1}{2}gt^2$ oraz $s_m = \frac{1}{2}gt_m^2$, gdzie t i t_m oznaczają, odpowiednio, czas spadku rzeczywistego pociągu i spadku jego modelu. Otrzymujemy:

$$k = \frac{s}{s_m} = \frac{t^2}{t_m^2}.$$

Oznacza to, że przebieg sfilmowanej katastrofy będzie „realistyczny”, gdy $t_m = t/\sqrt{k}$, czyli filmująca model kamera powinna rejestrować $f_m = f \cdot \sqrt{k}$ klatek na sekundę. Modele kolejek są standardowo produkowane w skali $k = 87$, a więc kamera filmowców amatorów powinna rejestrować $f_m = 24\sqrt{87} \approx 224$ klatki na sekundę. Czytelnik Dociekliwy łatwo udowodni, że taka szybkość filmowania pozwoli na realistyczne przedstawienie ruchu obrotowego spadających elementów pociągu.

Rozwiązanie zadania F 1040.
W warunkach podanych w zadaniu gazy wchodzące w skład powietrza (w tym para wodna), z dobrym przybliżeniem, zachowują się jak gazy doskonałe. Ciśnienie mieszaniny gazów doskonałych jest sumą ciśnień jej składników (prawo Daltona). Tlen i azot występują w postaci cząsteczek dwuatomowych, a argon jednoatomowych. Masa m_S jednego mola powietrza suchego (mieszaniny azotu, tlenu i argonu) wyniesie więc:

$$m_S = (0{,}78 \cdot 28 + 0{,}21 \cdot 32 + 0{,}01 \cdot 40)\,\mathrm{g}.$$

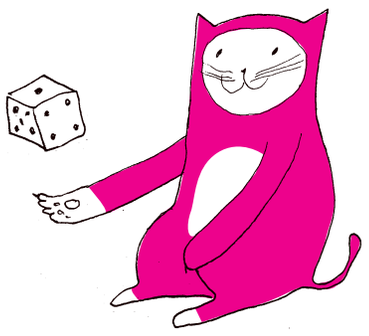
W jednym molu powietrza mokrego znajduje się ułamek u mola nasyconej pary wodnej równy stosunkowi ciśnienia pary p_p do całkowitego ciśnienia mieszaniny p, a więc

$$u = 2{,}3 \cdot 10^3/10^5 = 0{,}023.$$

Masa mola pary wodnej to 18 g. Masa 1 mola powietrza mokrego wynosi więc

$$m_m = (1 - 0{,}023) \cdot m_S + 0{,}023 \cdot 18\,\mathrm{g}.$$

Otrzymujemy $m_m/m_S \approx 0{,}9913$. Jest to także stosunek gęstości powietrza mokrego i suchego. Mokre powietrze ma mniejszą gęstość niż suche, co powoduje jego unoszenie się i gromadzenie w postaci chmur.



Pomysłem na ominięcie tego problemu jest zastosowanie tzw. generatorów pseudolosowych (*pseudorandom generator*). Taki generator to funkcja $G : \{0,1\}^n \to \{0,1\}^N$ dla pewnych $n \ll N$ taka, że jeśli wylosujemy z rozkładem jednostajnym $x \in \{0,1\}^n$ i obliczymy $G(x)$, to wynik „będzie wyglądał", jakby był wybrany jednostajnie ze zbioru $\{0,1\}^N$.

Co to znaczy „będzie wyglądał"? Tutaj najbardziej użyteczna okazuje się definicja „informatyczna": funkcja G jest generatorem pseudolosowym, jeżeli nie ma prostego algorytmu, który umiałby rozróżniać wartości wylosowane jednostajnie ze zbioru $\{0,1\}^N$ od wartości $G(x)$ dla losowo wybranego $x \in \{0,1\}^n$. Bardziej precyzyjnie powiemy, że obwód logiczny C o N wejściach *łamie* generator G, jeśli

- $\Pr_{x\in\{0,1\}^n}[C(G(x)) = 1] \geqslant 51\%$, ale
- $\Pr_{y\in\{0,1\}^N}[C(y) = 1] \leqslant 49\%$.

Zauważmy, że trudność generatora jest dobrze zdefiniowana, bo zawsze istnieje jakiś obwód, który go łamie. W szczególności, istnieje obwód wielkości mniej więcej 2^n, łamiący G: jest to obwód wyliczający alternatywę, po wszystkich $x \in \{0,1\}^n$, sprawdzeń, czy wejście y jest równe $G(x)$ (łamie G, jeżeli $N \geqslant n + 2$). Obwód ten można zakodować za pomocą 2^n kopii obwodu wyliczającego G oraz niewielkiej liczby dodatkowych bramek.

Trudnością generatora nazwiemy najmniejszą wielkość obwodu, który go łamie. Użyteczne generatory pseudolosowe to takie, które mają wysoką trudność.

W jaki sposób można użyć generatorów pseudolosowych do derandomizacji? Przypomnijmy sobie problem szacowania wagi obwodu CAP i algorytm losowy losujący pewną liczbę argumentów (stałą, np. 1000) i zwracający, dla jakiej frakcji wylosowanych argumentów obwód dał wynik 1. Załóżmy, że mamy wejście C dla problemu CAP (tj. obwód logiczny o N wejściach i jednym wyjściu). By algorytm losowy naiwnie zderandomizować, obliczylibyśmy $C(x)$ dla wszystkich 2^N wejść. Załóżmy jednak, że mamy jeszcze dany generator pseudolosowy $G : \{0,1\}^n \to \{0,1\}^N$. Jeśli G jest dobrym generatorem, spodziewamy się, że

$$(*) \qquad \frac{|\{x \in \{0,1\}^n : C(G(x)) = 1\}|}{2^n} \sim \frac{|\{y \in \{0,1\}^N : C(y) = 1\}|}{2^N}.$$

To dałoby algorytm losowy losujący pewną liczbę wartości $x \in \{0,1\}^n$ i zwracający, dla jakiej frakcji wylosowanych wartości x mamy $C(G(x)) = 1$, oraz jego derandomizację obliczającą bezpośrednio $|\{x \in \{0,1\}^n : C(G(x)) = 1\}|/2^n$ za pomocą 2^n obliczeń wartości obwodu C i funkcji G.

Mamy więc dwa przypadki: Jeśli $n \ll N$ i szacowanie $(*)$ zachodzi, daje nam to bardzo dobry algorytm deterministyczny dla problemu CAP na obwodzie C. Jeśli $(*)$ *NIE* zachodzi, to C zachowuje się istotnie inaczej na zbiorze wartości G niż na całej przeciwdziedzinie $\{0,1\}^N$. Okazuje się, że to wystarczy, by użyć C jako lewarka do skonstruowania małego (tj. wielkości porównywalnej z C) obwodu łamiącego G i wykazania, że trudność G jest tak naprawdę niska i G nie jest dobrym generatorem.

Dzięki pracom Aviego Wigdersona i współautorów wiemy, że dobre generatory pseudolosowe możemy skonstruować z problemów trudnych obliczeniowo. W szczególności, wraz z Russellem Impagliazzo pokazał on, że jeśli istnieje problem rozwiązywalny algorytmem działającym w czasie 2^n, którego nie można rozwiązać za pomocą obwodów wielkości podwykładniczej (dodajmy, że istnienia takich problemów się „spodziewamy"), to zachodzi równość klas BPP = P. Klasa P to klasa problemów, dla których istnieje deterministyczny wielomianowy algorytm. Z kolei klasa BPP to klasa problemów, dla których istnieje wielomianowy algorytm, który może korzystać z bitów losowych, ale daje niepoprawną odpowiedź z prawdopodobieństwem co najwyżej 1/3. Nasz algorytm dla problemu testowania równości wielomianów PIT spełnia ten warunek, czyli należy do BPP. Równość klas BPP = P oznacza zatem, że PIT i wiele podobnych problemów możemy rozwiązać w deterministycznym czasie wielomianowym.

Avi Wigderson i Russell Impagliazzo pokazali, że jeżeli istnieje problem $f \in \mathrm{DTIME}(2^n)$, dla którego istnieje stała $\delta > 0$, oraz że obwód obliczający f dla danych n-bitowych ma wielkość co najmniej $2^{\delta n}$ dla każdego n, to zachodzi BPP = P.

Odwracając kota ogonem, wynik Impagliazzo i Wigdersona oznacza, że jeśli wymieniony prosty algorytm dla PIT nie da się zderandomizować, to w świecie algorytmów wykładniczych istnieje bardzo nieoczekiwana rozbieżność między siłą „zwykłych" algorytmów a siłą obwodów logicznych. Tego raczej się nie spodziewamy, ale na razie nie umiemy wykluczyć.