Skąd wiadomo, czy procesor jest dobry

W 1993 roku firma *Intel* wprowadziła do sprzedaży procesor *Pentium*. Dziś w takie właśnie procesory wyposażona jest większość sprzedawanych na całym świecie komputerów osobistych. W tym samym roku Thomas Nicely, profesor matematyki w prowincjonalnym Lynchburg College (Virginia, USA), postanowił poprawić dostępne numeryczne oszacowania wartości tzw. *stałej Bruna B*.

Aby powiedzieć, co to jest stała Bruna, trzeba najpierw przypomnieć jedną definicję. O dwóch liczbach pierwszych mówimy, że są *bliźniacze*, jeśli różnią się o 2 (jak np. 3 i 5, albo 17 i 19, albo $p_1 = 824\,633\,702\,441$ i $p_2 = 824\,633\,702\,443$). Do dziś nie wiadomo, czy takich par jest skończenie, czy też nieskończenie wiele. Największe, które obecnie znamy, to $697\,053\,813 \cdot 2^{16\,352} \pm 1$. Wkrótce po pierwszej wojnie światowej norweski matematyk, Viggo Brun, udowodnił jednak, że suma odwrotności wszystkich liczb pierwszych bliźniaczych

$$B = \left(\tfrac{1}{3} + \tfrac{1}{5}\right) + \left(\tfrac{1}{5} + \tfrac{1}{7}\right) + \left(\tfrac{1}{11} + \tfrac{1}{13}\right) + \cdots,$$

to liczba skończona. Dzięki obliczeniom różnych pasjonatów wiadomo dziś, że $B \approx 1{,}90216057778$ (i jest to wiedza całkowicie bezużyteczna).

Nie to jednak jest najciekawsze. Gdy w czerwcu 1994 roku Nicely analizował pierwsze wyniki swych obliczeń, stwierdził ze zdziwieniem, że uzyskane przezeń wartości $\pi(n)$ (liczby liczb pierwszych nie przekraczających n) są dla niektórych n inne od opublikowanych. Po licznych poszukiwaniach błędu w oprogramowaniu, znalezieniu usterki w kompilatorze języka C++ firmy Borland itd. okazało się w końcu, że przyczyna niezgodności jest prozaiczna: procesor *Pentium* nie potrafi po prostu dzielić, a dokładniej – wartości $1/p_1$ i $1/p_2$ dla podanych wyżej dwunastocyfrowych bliźniaczek p_1 i p_2 daje z dokładnością jedynie 9 miejsc po przecinku (zamiast obiecanych przez producenta 19).

Firma *Intel* początkowo nie zareagowała na wieści Nicely'ego i dopiero rozpropagowanie informacji o usterkach *Pentium* w Internecie zmusiło ją do rozpoczęcia wymiany procesorów na życzenie klientów. Nicely zaś, który w przedziale $[1, 10^{14}]$ znalazł w końcu 135 780 321 665 par liczb pierwszych bliźniaczych, używa teraz do *każdego* rachunku przynajmniej dwóch różnych komputerów.

Historia Nicely'ego to jedynie przypadek. Niemniej jednak programy do poszukiwania dużych liczb pierwszych (albo znajdowania dalekich cyfr rozwinięcia dziesiętnego π) są używane często przez samych producentów sprzętu do testowania własnych produktów. Np. gazetowa wiadomość z września 1996 roku o tym, że $2^{1\,257\,787} - 1$ jest największą (wówczas znaną) liczbą pierwszą, zawdzięczała swoje pochodzenie testom nowego CRAYa.

P.S.

Pogawędka o semantyce języków programowania

Skomplikowane konstrukcje języka programowania wysokiego poziomu są zwykle objaśniane przez odwoływanie się do pojęć translatora (programu tłumaczącego) danego języka. Na przykład łamigłówka

```
var a,b,c,:integer;
procedure p(var x1,x2,x3:integer;
      procedure f(par1:integer;var par2,par3:integer));
  var loc:integer;
  procedure q(y1:integer;var y2,y3:integer);
    begin
     y1:=y3;y3:=y2;y2:=loc;
     f(y2,y3,y1)
    end;
  begin
    x1:=x1+1;loc:=x1;
    if x1<6 then p(x1,x2,x3,q) else f(x1,x2,x3);
  end;
procedure r(z1:integer;var z2,z3:integer);
  begin
  end;
begin
 a:=1;b:=2;c:=3;
 p(a,b,c,r);
 write(a,b,c);
end
```

jest często objaśniana w terminach stosu rekordów aktywacji, dowiązania statycznego, adresu kodu procedury itd. itp. Co bardziej dociekliwi mogą zatem zapytać: „Po co język wysokiego poziomu, skoro gdy chcemy go naprawdę zrozumieć, musimy i tak zejść na niższy poziom?". Co więcej, szybko okazało się, że np. PASCAL w wersji jednego translatora nie musi być tym samym, co PASCAL w wersji innego. W końcu, jaki ten PASCAL jest naprawdę? Czy, na przykład, deklaracje zmiennych mają znaczenie tylko w kontekście następujących po nich instrukcji, czy też mają jakieś znaczenie niezależnie od kontekstu, jako rzeczy same w sobie?

Stworzenie niezależnych od komputera metod definiowania znaczenia poszczególnych konstrukcji języka programowania było traktowane na przełomie lat 60/70 jako poważne wyzwanie intelektualne. Pierwsze zadowalające rozwiązanie (tzw. semantyka denotacyjna) zostało zaproponowane w latach 70. przez informatyków i matematyków z Uniwersytetu w Oxfordzie.

Czy można udowodnić w jakimś sformalizowanym systemie wnioskowania – bez odwoływania się do pojęć translatora – że program powyżej wypisze to i to? Takie formalne systemy są. Wykazanie ich niesprzeczności jest bardzo trudne i prowadzi do bardzo subtelnych semantyk języków programowania, a także do zagadnień związanych z określaniem typu obiektów. Nawet mocno wyabstrahowane wersje pojawiających się tu problemów są na tyle trudne, że do ich badania używane są metody z teorii dziedzin, rachunku λ, logiki intuicjonistycznej, teorii kategorii, teorii gier, czy teorii złożoności obliczeń.

Michał GRABOWSKI