Co może maszyna Turinga, czyli o algorytmach (I)

Prof. dr Andrzej MOSTOWSKI, członek rzeczywisty PAN

Według *Małej Encyklopedii Powszechnej* (PWN, Warszawa 1952, s. 18) algorytm jest to „określona metoda postępowania w celu rozwiązania danego zadania rachunkowego, np. algorytm znajdowania pierwiastków kwadratowych".
Słowo „algorytm" jest zniekształconym nazwiskiem astronoma Muhammeda ibn Musa Alchwarizmi, pochodzącego z Uzbekistanu; żył on w pierwszej połowie IX wieku, a działał na dworze kalifa w Bagdadzie. W roku 830 napisał on książkę pt. *Hisāb al-djabr wa'l-mukābala* (czyli nauka o redukcji i przeniesieniach), w której wyłożył znane wówczas wiadomości o tym, co — pod wpływem jego książki — nazwano potem algebrą. Książka ta odegrała w historii matematyki dużą rolę: dzięki niej odkrycia matematyków hinduskich, a w szczególności używane powszechnie dzisiaj cyfry oraz system dziesiętny zapisywania liczb naturalnych, przyjęły się w matematyce arabskiej i w konsekwencji — europejskiej.
Umiejętność wykonywania prostych czynności rachunkowych rozwijała się wolno. Znane i powszechnie dziś używane algorytmy, takie jak algorytm dodawania lub algorytm mnożenia liczb zapisanych w układzie dziesiętnym, przyjęły się dopiero w wieku piętnastym.
Dziś nie zaliczamy tych umiejętności do poważnej matematyki. Uczymy się ich w bardzo młodym wieku i traktujemy je jako zupełnie oczywiste. W czasach Alchwarizmiego było inaczej. Formułowanie prostych algorytmów było wtedy poważną i potrzebną działalnością matematyka, głównie dlatego, że system dziesiętny nie był rozpowszechniony.
Niektóre algorytmy są łatwe i znane każdemu, jak np. algorytmy dodawania i mnożenia w układzie dziesiętnym. Łatwo je przeformułować tak, by stosowały się do układu dwójkowego lub jakiegokolwiek innego. Inne algorytmy są trochę trudniejsze, jak na przykład algorytm obliczania pierwsiastka kwadratowego albo algorytm pozwalający z rozwinięcia dziesiętnego przechodzić do rozwinięcia dwójkowego. Ale proszę się zastanowić nad algorytmem obliczania pierwiastka kwadratowego z liczby zapisanej w układzie dwójkowym. Na pewno trzeba się trochę pomęczyć, zanim się taki algorytm znajdzie (jest on opisany w książce I. Floresa, *Arytmetyka maszyn cyfrowych*, Warszawa 1970, WNT, ale kto ma poczucie humoru i zna angielski, niech lepiej zajrzy do książki: Ian Synge, *Kandelman's Krim, A realistic fantazy*, London 1957, Jonathan Cape, s. 104).
A oto jeszcze jeden niebanalny przykład algorytmu: postępowanie pozwalające wyznaczać kolejne cyfry rozwinięcia dziesiętnego liczby π. W zasadzie algorytm taki znalazł już Archimedes.

W wielu przypadkach nie wiemy, czy dane zadanie rachunkowe daje się rozwiązać przy pomocy algorytmu, czy istnieje „określona metoda postępowania", pozwalająca wyznaczać szukane wielkości. Następujący przykład, w którym taka metoda zapewne istnieje, pochodzi od słynnego matematyka holenderskiego L. E. J. Brouwera (1881–1966): wyznaczyć kolejne liczby n, takie że w rozwinięciu dziesiętnym liczby π stoi jedna za drugą n cyfr 7. Nie wiemy, czy np. 7 jest taką liczbą, i nie mamy pojęcia, jak na to pytanie odpowiedzieć.
Jakkolwiek przypuszczamy, że nie ma algorytmu, który pozwalałby na wyznaczanie tych liczb, to dowodu na to nie posiadamy. Jeśli chcemy na serio odpowiedzieć na pytanie, czy jakieś zadanie daje się rozwiązywać przy pomocy algorytmu, musimy najpierw podać ścisłą matematyczną definicję algorytmu. Tym problemem zajmowano się intensywnie w trzydziestych latach obecnego wieku i zaproponowano szereg definicji, które zresztą wszystkie okazały się równoważne. Naszkicujemy tu pierwszą i chyba najbardziej znaną definicję, pochodzącą od angielskiego matematyka A. M. Turinga (1912–1954).
Przede wszystkim musimy uświadomić sobie, że każdy algorytm jest przepisem na przekształcanie wyrażeń. Na przykład algorytm dodawania liczb zapisanych w układzie dwójkowym pozwala na uzyskanie z dwóch ciągów zer i jedynek nowego takiego ciągu; podobnie jest dla wszystkich innych algorytmów wspomnianych wyżej. Dlatego Turing opisując abstrakcyjne pojęcie algorytmu wyobraża sobie dwustronnie nieskończoną taśmę podzieloną na pola, z których każde zajęte jest przez pewien symbol. Dopuszczamy tylko skończoną ilość symboli $s_0, s_1, \ldots, s_N$, przy czym s_0 możemy uważać za symbol „pusty", tj. brak jakiegokolwiek symbolu na polu uważamy za równoznaczne z tym, że na polu tym figuruje symbol s_0. W każdej chwili tylko skończona ilość pól na taśmie jest zajęta przez symbole różne od s_0.
Taśma jest częścią urządzenia, nazywanego „maszyną Turinga". Maszyna ta

działa, wykonując kolejno poszczególne kroki; w każdej jednostce czasu maszyna wykonuje jeden krok.

Turing zakłada, że w każdej chwili maszyna znajduje się w pewnym stanie, przy czym ilość możliwych stanów jest skończona. Oznaczymy te stany symbolami q_0, q_1, ..., q_P. W konkretnej realizacji technicznej stany maszyny są wyznaczone przez wzajemne położenia jej części, a więc dźwigien, kół etc., lub też przez stany elektryczne albo magnetyczne tych jej części, które działają na zasadach elektromagnetycznych. Przy opisie abstrakcyjnym nie musimy się zastanawiać, czym konkretnie są stany. Wystarczy wiedzieć, że w każdej chwili maszyna znajduje się w pewnym stanie.

Zakładamy dalej, że maszyna jest wyposażona w urządzenie, zwane głowicą, która w każdej chwili obserwuje jedno z pól taśmy. Pole to jest w tej chwili wyróżnione. Maszyna jest w stanie wykonywać następujące czynności: (L) — przesunąć taśmę o jedno pole w lewo; (P) — przesunąć taśmę o jedno pole w prawo; (D_j) — usunąć symbol znajdujący się na polu wyróżnionym i zastąpić go symbolem s_j $(j = 0, 1, \ldots, N)$. Przy czynnościach (L) i (P) następuje zmiana wyróżnionego pola, natomiast przy czynnościach (D_0), ..., (D_N) to samo pole jest wyróżnione po dokonaniu czynności, jakie były wyróżnione przed jej wykonaniem.

Jakie czynności maszyna będzie wykonywać i w jakim porządku, zależy od instrukcji, które musimy ustalić przed uruchomieniem maszyny. Instrukcja składa się z dwu części: warunku, który ustala, kiedy instrukcję można wykonać, oraz instrukcji właściwej. Warunek ma zawsze postać: „jeśli jesteś w stanie q_i, a na polu wyróżnionym jest symbol s_h", instrukcja właściwa ma jedną z trzech postaci: „wykonaj czynność (L) i przejdź do stanu q_k" albo „wykonaj czynność (P) i przejdź do stanu q_k", albo „wykonaj czynność (D_j) i przejdź do stanu q_k". Ostatecznie więc każdą instrukcję możemy zapisać w jednej z następujących postaci: $q_i s_h P q_k$, $q_i s_h L q_k$, $q_i s_h s_j q_k$, przy czym dwa pierwsze symbole $q_i s_h$ określają warunek wykonalności, a dwa ostatnie tworzą instrukcję właściwą.

Działanie maszyny jest określone przez program, to jest skończony zbiór instrukcji. Instrukcje te nie mogą być zupełnie dowolne, nie możemy bowiem udzielać maszynie dwu sprzecznych instrukcji. Gdybyśmy w programie mieli na przykład instrukcje $q_1 s_1 L q_2$ oraz $q_1 s_1 P q_2$, to maszyna znajdująca się w stanie q_1 z symbolem s_1 na polu wyróżnionym nie wiedziałaby, czy ma wykonać czynność (P), czy (L). Żądamy zatem, aby w programie nie znalazły się dwie różne instrukcje o tych samych warunkach.

Możemy teraz opisać działanie maszyny. Umieszczamy najpierw na taśmie dowolne symbole spośród s_0, s_1, ..., s_N, tak aby tylko skończona liczba symboli była różna od s_0. Ciąg tych symboli tworzy tzw. początkowy stan taśmy, albo krótko: „wejście". Przyjmiemy umownie, że stanem maszyny w chwili początkowej jest q_0 i że polem wyróżnionym jest ostatnie pole po prawej stronie taśmy, zajęte przez symbol różny od s_0.

Z chwilą uruchomienia maszyny wyszukuje ona w programie instrukcję, którą może zastosować, tj. taką, że jej warunki zgadzają się ze stanem, w jakim maszyna się znajduje, i z symbolem na polu wyróżnionym.

Po wykonaniu czynności wskazanej przez instrukcję i przejściu do nowego stanu, wyznaczonego przez instrukcję, maszyna znów wyszukuje instrukcję, która daje się zastosować, i powtarza to postępowanie tak długo, jak to jest możliwe.

Może się zdarzyć, że po wykonaniu pewnej ilości poruszeń maszyna nie znajdzie już instrukcji, którą mogłaby wykonać. W tym przypadku maszyna zatrzymuje się. Ciąg symboli figurujący na taśmie w końcowej chwili działania maszyny, czyli tzw. „wyjście", jest wynikiem algorytmu określonego przez maszynę, zastosowanego do wejścia. Jeśli oznaczymy przez M maszynę, a przez w jej wejście, to wyjście oznaczamy przez $M(w)$.

Druga możliwość jest taka, że maszyna zawsze znajduje instrukcję dającą się zastosować. Nie zatrzymuje się ona wtedy nigdy. Mówimy, że algorytm opisany przez maszynę nie daje się zastosować do wejścia, albo że $M(w)$ jest nieokreślone.

Jako prosty przykład określimy maszynę opisującą algorytm dodawania jedynki do liczby zapisanej w rozwinięciu dwójkowym. Algorytm ten jest, jak wiemy, następujący: jeśli ostatnią cyfrą jest 0, to zastępujemy ją przez 1, poprzednich zaś cyfr nie zmieniamy; jeśli ostatnią cyfrą jest 1, to zmieniamy ją na 0 i powtarzamy opisane postępowanie z przedostatnią cyfrą. Ten proces kontynuujemy aż do wyczerpania wszystkich cyfr.

Aby opisać ten algorytm przy pomocy maszyny Turinga, musimy mieć 3 symbole: s_0, 0, 1 oraz 3 stany: q_0 (stan, w którym maszyna zmienia cyfrę na polu wyróżnionym), q_1 (stan, w którym maszyna przesuwa taśmę nie mając jedynki do „przeniesienia"), q_2 (stan, w którym maszyna przesuwa taśmę mając jedynkę do „przeniesienia").

Rozwiązania — gry

1. Jeśli strategie minimaksowe są w równowadze, to istnieje taka wypłata, która jest jednocześnie najmniejszą w wierszu i największą w kolumnie. Przypuśćmy, że jest nią a. Wtedy $c < a < b$. Jeśli teraz $d > c$, to B_1 jest dominująca (pamiętamy, że wypłaty dla B mają przeciwne znaki). Jeśli $c > d$, to tym bardziej $b > d$ i A_1 jest dominująca. W pozostałych przypadkach — analogicznie.

2. Jeśli $a+d = b+c$, to albo $a < b$ i $c < d$, albo $a > b$ i $c > d$. W pierwszym przypadku B_1 jest dominująca, w drugim — B_1 jest dominowana, a z założenia nie ma takich strategii.

3. $w(p, y) = ap+c(1-p)y+$ $+(bp+d(1-p))(1-y)$.
Łatwo sprawdzamy, że $ap+c(1-p) = bp+$ $+d(1-p) = v$, stąd $w(p, y) = v \cdot y+v(1-y)$ $= v$. Dla $w(x, q)$ — analogicznie.

4. Jeśli B stosuje jakąkolwiek strategię, to A stosując strategę $(p, 1-p)$ może zapewnić sobie wypłatę v i nie może zapewnić wypłaty większej. Z drugiej strony B, stosując strategię $(q, 1-q)$, zapewnia sobie to, że A nie otrzyma wypłaty większej niż v.

Program maszyny składa się z 6 instrukcji:

$$q_0 s_0 1 q_2, \quad q_0 01 q_1, \quad q_0 10 q_2, \quad q_1 0 P q_1, \quad q_1 1 P q_1, \quad q_2 0 P q_0.$$

Przypuśćmy np., że wejściem maszyny jest ciąg 1010. Stany maszyny i symbole na taśmie są wówczas następujące („tłusta" czcionka oznacza pole wyróżnione):

(1) $1010\, q_0$, (2) $101\mathbf{1} q_1$, (3) $10\mathbf{1}1 q_1$, (4) $1\mathbf{0}11 q_1$, (5) $\mathbf{1}011 q_1$, (6) $s_0\, 1011 q_1$.

Jeśli natomiast wejściem jest ciąg 1111, to działanie maszyny przebiega następująco:
(1) $1111 q_0$, (2) $111\mathbf{0} q_2$, (3) $11\mathbf{1}0 q_0$, (4) $1\mathbf{1}00 q_2$, (5) $\mathbf{1}100 q_0$,
(6) $\mathbf{1}000 q_2$, (7) $1000 q_0$, (8) $\mathbf{0}000 q_2$, (9) $s_0\, 0000 q_0$, (10) $\mathbf{1}0000 q_2$.

Podany tu przykład jest niezwykle prosty i nie daje pełnego świadectwa tego, co maszyna Turinga może naprawdę zdziałać. Czytelnik, który chciałby się nauczyć, jak z prostych czynności wykonywanych przez maszyny Turinga można składać działania coraz bardziej złożone, musiałby zwrócić się do łatwo zresztą dostępnych opracowań poważniejszych (zob. np. B. A. Trahtenbrot, *Algorifmy i wyczislitelnyje awtomaty*, Moskwa 1974). Jakkolwiek jednak złożone będzie działanie wykonywane przez maszynę Turinga, będzie ono miało zawsze charakter algorytmiczny: wykonanie go nie będzie wymagało inteligencji, lecz tylko uwagi i ścisłego przestrzegania instrukcji.
W dotychczasowej części artykułu staraliśmy się być bardzo dokładni. W dalszej części, w której nie możemy już prowadzić wykładu tak ściśle, postaramy się opowiedzieć, jak można skonstruować zadanie rachunkowe, dla którego nie istnieje rozwiązanie przy pomocy maszyny Turinga.

Równania różniczkowe

Dr Henryk KOŁAKOWSKI

Każdy z czytelników spotkał się niejednokrotnie z równaniami algebraicznymi postaci:

$$f(x, y) = 0, \tag{1}$$

gdzie f jest funkcją rzeczywistą określoną na prostokącie $\Omega = \{(x, y): a < x < b, c < y < d\}$, tzn. funkcją, która każdej parze liczb $(x, y) \in \Omega$ przyporządkowuje liczbę rzeczywistą $f(x, y)$. Rozwiązaniem równania (1) nazywa się w szkole każdą parę liczb $(p, q) \in \Omega$ spełniającą warunek:

$$f(p, q) = 0.$$

Można na to spojrzeć inaczej: rozwiązaniem równania (1) będziemy nazywali funkcję

$$y = y(x)$$

określoną na pewnym zbiorze $I \subset (a, b)$, spełniającą warunki:

$$\text{jeśli} \quad x \in I, \quad \text{to} \quad (x, y(x)) \in \Omega,$$
$$\text{jeśli} \quad x \in I, \quad \text{to} \quad f(x, y(x)) = 0.$$

Równaniem typu (1) jest na przykład równanie liniowe

$$Ax + By + C = 0,$$

gdzie A, B, C są liczbami rzeczywistymi i $B \neq 0$. Wówczas jedynym rozwiązaniem jest funkcja

$$y = \frac{-Ax - C}{B}.$$

Rozpatrzmy jeszcze jeden przykład:

$$x^2 + y^2 - 1 = 0, \quad x \in (-1, 1), \quad y \in (-1, 1).$$

Spośród rozwiązań określonych na $I = (-1, 1)$ ciągłe są dwa :

$$y = \sqrt{1 - x^2} \quad \text{oraz} \quad y = -\sqrt{1 - x^2}.$$

Po tym krótkim wstępie przejdziemy do omówienia najprostszych równań różniczkowych.
Definicja: *Równaniem różniczkowym zwyczajnym rzędu pierwszego* nazwiemy równanie:

$$\frac{dy}{dx} = f(x, y), \tag{2}$$

gdzie f jest funkcją określoną na prostokącie Ω.
Rozwiązaniem równania (2) nazywamy każdą funkcję $y = y(x)$ różniczkowalną w przedziale I, której wykres leży w zbiorze Ω i która spełnia warunek (2), to znaczy:

$$\text{jeśli} \quad x \in I, \quad \text{to} \quad \frac{dy(x)}{dx} = f(x, y(x)).$$

Krzywą $y = y(x)$, $x \in I$ nazywamy krzywą całkową równania (2).