Obliczenia: rachunki, dowody i gry

Filip MURLAK[*]

Co da się obliczyć, a czego się nie da? Dziś informatycy pytają o to rutynowo w kontekście przeróżnych problemów, jednak pierwsza odpowiedź na to pytanie pojawiła się na dobrą sprawę, jeszcze zanim ktokolwiek zdążył je zadać, i wprawiła środowisko naukowe w zakłopotanie.

Na początku XX wieku matematycy wierzyli, że każdy problem matematyczny da się rozstrzygnąć za pomocą obliczeń. U szczytu tego optymizmu, w 1928 roku, David Hilbert postulował opracowanie uniwersalnej metody pozwalającej na obliczenie prawdziwości dowolnego stwierdzenia sformułowanego w języku logiki pierwszego rzędu – czyli za pomocą spójników logicznych *i*, *lub*, *nie* oraz kwantyfikatorów *istnieje* i *dla każdego*. Niecałe 10 lat później Alan Turing udowodnił, że taki algorytm nie istnieje. A więc są rzeczy, których obliczyć się nie da! Ale co to właściwie znaczy? Wydaje się, że wiemy, co to znaczy *obliczyć*. Jednak aby wykazać, że czegoś obliczyć się nie da, potrzebujemy więcej niż tylko nieformalnej intuicji. Potrzebujemy *definicji* obliczenia.

Praca Alana Turinga ukazała się w 1937 roku. Rok wcześniej taki sam wynik osiągnął Alonzo Church dla innej formalizacji pojęcia obliczenia, która potem okazała się równoważna.

Rachunek jako obliczenie. Aby sformułować definicję obliczenia, Alan Turing przeanalizował pracę rachmistrza, czyli człowieka przeprowadzającego rachunki. Rachmistrz, mówi Turing, prowadzi obliczenia na papierze. Papier jest wprawdzie dwuwymiarowy, ale podczas obliczeń nie czynimy z tego użytku. Przyjmiemy więc, że obliczenie prowadzimy na papierze jednowymiarowym, czyli *taśmie*. Taśma ta jest podzielona na *komórki*, odpowiadające kratkom na papierze. W każdej komórce można napisać *symbol*. Komórka ma ograniczoną powierzchnię, a oko ludzkie ma ograniczoną rozdzielczość, więc bez zmniejszenia ogólności możemy przyjąć, że symbole pochodzą ze *skończonego alfabetu*. Działania rachmistrza zależą od obserwowanych symboli i jego *stanu umysłu*. Umysł ludzki jest obiektem skończonym i liczba odróżnialnych stanów, które może przyjmować, jest skończona (choć wielka i trudna do oszacowania). Rachmistrz jest w stanie jednocześnie obserwować jedynie pewną niewielką liczbę komórek, powiedzmy K.

Jakie działania wykonuje rachmistrz? Spróbujmy je opisać w terminach możliwie najprostszych, niepodzielnych akcji. Rachmistrz może *napisać symbol* w obserwowanej komórce taśmy. Może też *zmienić obserwowaną komórkę* na dowolną inną, jeśli tylko potrafi ją natychmiast zidentyfikować. Jakie komórki mogą być natychmiast zidentyfikowane? Komórki, która jest bardzo daleko od wszystkich obserwowanych komórek, nie da się łatwo odróżnić od jej sąsiadów. Rachmistrz jest w stanie dokładnie określić odległość jednym rzutem oka tylko wtedy, gdy odległość ta nie jest zbyt duża, powiedzmy nie większa niż D. Przyjmijmy, że rachmistrz może przesunąć każdą obserwowaną komórkę o co najwyżej D pozycji w lewo lub w prawo. Ponadto, w czasie prowadzenia obliczeń rachmistrz *zmienia stan umysłu*. Reasumując, pojedynczy krok obliczenia obejmuje napisanie symbolu w niektórych obserwowanych komórkach, przesunięcie niektórych obserwowanych komórek o co najwyżej D pozycji oraz zmianę stanu umysłu.

Nasz rachmistrz jest profesjonalistą, który bezbłędnie opanował reguły prowadzenia rachunków, ale nie ma własnej inwencji: każdy kolejny krok jego rachunku jest całkowicie zdeterminowany przez zawartość obserwowanych komórek i stan umysłu. Reguły stosowane przez rachmistrza można opisać funkcją, która ciągowi symboli $a_1, a_2, \ldots, a_K$ znajdujących się w obserwowanych komórkach i stanowi umysłu p przyporządkowuje nowy ciąg symboli $b_1, b_2, \ldots, b_K$, ciąg przesunięć obserwowanych komórek

[*]Instytut Informatyki, Wydział Matematyki, Informatyki i Mechaniki, Uniwersytet Warszawski

$d_1, d_2, \ldots, d_K$ oraz nowy stan q. To znaczy, że rachmistrz w pierwszej obserwowanej komórce pisze symbol b_1 (jeśli $a_1 = b_1$, to rachmistrz nic nie pisze), a następnie zmienia obserwowaną komórkę na komórkę znajdującą się o d_1 pozycji w prawo, jeśli $d_1 > 0$, lub o $-d_1$ pozycji w lewo, jeśli $d_1 < 0$ (jeśli $d_1 = 0$, obserwowana komórka się nie zmienia). Następnie rachmistrz postępuje podobnie dla komórek $2, 3, \ldots, K$, po czym zmienia stan umysłu na q. Rachmistrz zaczyna obliczenie w pewnym ustalonym stanie umysłu, mając na taśmie zapisane dane początkowe i obserwując pierwszych K komórek taśmy. Potem wykonuje akcje wskazywane przez reguły. Obliczenie kończy się, gdy rachmistrz osiągnie odpowiedni stan umysłu, również z góry ustalony. Wynik obliczenia jest zapisany na taśmie.

Zwróćmy uwagę, że pracę rachmistrza może wykonać każdy, jeśli tylko dostanie odpowiednie instrukcje: listę symboli alfabetu, listę możliwych stanów, stan początkowy, stan końcowy oraz powyższą funkcję. A skoro tak, to dlaczego nie mogłaby tej pracy wykonywać maszyna? W ciągu 10 lat od ogłoszenia koncepcji maszyny Turinga pierwsze programowalne komputery zostaną skonstruowane w Niemczech, Wielkiej Brytanii i Stanach Zjednoczonych...

Wróćmy jednak do problemu obliczalności. Alan Turing przyjął, że uniwersalna metoda obliczania odpowiedzi na pytanie to po prostu taki zestaw instrukcji dla rachmistrza: wymagamy jedynie tego, żeby dla każdych danych początkowych rachmistrz kończył obliczenie i zapisywał na taśmie poprawną odpowiedź. W przypadku problemu postawionego przez Hilberta dane początkowe to stwierdzenie zapisane za pomocą symboli logicznych, a odpowiedź brzmi *tak* lub *nie*, zależnie od tego, czy dane stwierdzenie jest prawdziwe, czy fałszywe. Turing udowodnił, że istnieją problemy, dla których nie ma uniwersalnej metody obliczania odpowiedzi i że zagadnienie opisane przez Hilberta jest jednym z takich problemów.

O problemach takich mówimy, że są *nierozstrzygalne*. Innym słynnym problemem nierozstrzygalnym jest *problem stopu*, tzn. problem stwierdzenia, czy dla danego zestawu instrukcji i danych początkowych obliczenie zakończy się, czy też będzie trwało w nieskończoność.

Dowód jako obliczenie. Wyniki Turinga zdają się obnażać słabość matematyki: oto są rzeczy, których nie da się obliczyć. Ale przecież matematyka to nie tylko rachunki! Czym różni się praca matematyka próbującego udowodnić twierdzenie od pracy rachmistrza? Rachmistrz działa zgodnie z instrukcjami, jednoznacznie wskazującymi każdy kolejny krok obliczeń. Matematyk, prowadząc rozumowanie, również kieruje się pewnymi regułami, jednak reguły te nie mówią, jaki krok należy wykonać, ale raczej jakie są możliwe kroki. Spośród wielu możliwych kroków za każdym razem trzeba wybrać jeden i nie ma żadnej gwarancji, że akurat ten doprowadzi nas do celu. Aby uwzględnić to zjawisko w modelu maszyny Turinga, wystarczy zmodyfikować funkcję opisującą reguły obliczenia. Funkcja opisująca pracę rachmistrza wskazywała kolejną akcję na podstawie symboli w obserwowanych komórkach i bieżącego stanu umysłu. Funkcja opisująca pracę matematyka będzie wskazywała *zbiór możliwych akcji*. Taki model obliczeń nazywamy *niedeterministyczną* maszyną Turinga. Model opisany wcześniej to maszyna *deterministyczna*.

Dla uproszczenia rozważań przyjmijmy, że interesują nas wyłącznie pytania typu *tak*/*nie*. Nawet dla takich pytań różne ciągi wyborów maszyny niedeterministycznej mogą prowadzić do różnych odpowiedzi. Co zatem jest wynikiem obliczeń? Wróćmy na chwilę do zjawiska, które usiłujemy modelować, czyli do dowodzenia twierdzeń. Twierdzenie uznajemy za udowodnione, gdy odkryjemy ciąg kroków prowadzący do celu. Nie przejmujemy się tym, że wiele wyborów okazuje się złych i kolejne próby dowodu lądują w koszu... Wypada nam zatem przyjąć, że wynikiem obliczenia niedeterministycznej maszyny Turinga dla ustalonych danych początkowych jest *tak*, jeśli choć jeden ciąg wyborów prowadzi do odpowiedzi *tak*. Jeśli wszystkie ciągi wyborów prowadzą do odpowiedzi *nie*, wtedy przyjmujemy, że wynikiem obliczenia jest *nie*.

Innymi słowy, maszyna konstruuje dowód na to, że odpowiedź brzmi *tak*: należy odpowiedzieć *tak* wtedy, gdy istnieje choć jeden dowód.

Wydawać by się mogło, że niedeterministyczne maszyny potrafią rozwiązać więcej problemów niż deterministyczne. Okazuje się jednak, że tak nie jest: każdą niedeterministyczną maszynę Turinga $\mathcal{N}$ można zasymulować za pomocą maszyny deterministycznej $\mathcal{D}$. Symulacja ta nie jest trudna do wyobrażenia: wystarczy równolegle rozważać wszystkie możliwe wybory maszyny $\mathcal{N}$. Obliczenia obu maszyn rozpoczynają się tak samo: na taśmie zapisane są dane początkowe, maszyny znajdują się w stanie początkowym, obserwują pierwsze K komórek taśmy. Maszyna $\mathcal{N}$ w tym momencie wybiera jedną z kilku możliwych akcji. Maszyna $\mathcal{D}$ natomiast kopiuje zawartość taśmy tyle razy, ile możliwych akcji ma maszyna $\mathcal{N}$ (kolejne kopie oddziela dodatkowym symbolem, nieużywanym przez $\mathcal{N}$). Ponadto, w każdej kopii zaznacza pozycję komórek obserwowanych przez $\mathcal{N}$ oraz stan maszyny $\mathcal{N}$. Następnie, w każdej z tych kopii $\mathcal{D}$ symuluje inny wybór $\mathcal{N}$: modyfikuje zawartość taśmy, przesuwa obserwowane komórki i zmienia stan maszyny zgodnie z opisem akcji znajdującym się w zbiorze reguł maszyny $\mathcal{N}$. W ten sposób $\mathcal{D}$ ma na taśmie wszystkie możliwe rezultaty obliczenia $\mathcal{N}$ po jednym kroku. Dalej symulacja przebiega podobnie: dla każdego rezultatu obliczenia wykonujemy tyle kopii, ile maszyna $\mathcal{N}$ ma możliwości do wyboru, a potem w każdej kopii wykonujemy inną akcję. Po kolejnych rundach symulacji mamy na taśmie zapisane wszystkie możliwe rezultaty obliczenia $\mathcal{N}$ po $2, 3, 4, \ldots$ krokach. Jeśli w dowolnym momencie otrzymamy wynik *tak*, to kończymy symulację i udzielamy odpowiedzi *tak*. Jeśli natomiast dojdziemy do sytuacji, w której wszystkie rezultaty odpowiadają zakończonemu obliczeniu $\mathcal{N}$ z wynikiem *nie*, to kończymy obliczenie z odpowiedzią *nie*.

Czytelnik Dociekliwy spyta, czy mamy gwarancję, że nasza symulacja zawsze się zakończy. Jeśli istnieje ciąg wyborów $\mathcal{N}$ prowadzący do wyniku *tak*, to w końcu się na niego natkniemy. Ale co, jeśli taki ciąg wyborów nie istnieje? Wtedy wiemy, że każdy ciąg wyborów maszyny $\mathcal{N}$ prowadzi do odpowiedzi *nie* w skończenie wielu krokach. Czy z tego wynika, że nasza symulacja też się skończy po skończenie wielu krokach? Okazuje się, że tak. Gwarancji udziela nam następujący lemat Königa: *jeśli każdy wierzchołek w drzewie ma skończenie wiele dzieci i każda gałąź jest skończona, to całe drzewo też jest skończone.*

Gra jako obliczenie. Wiemy już, że dodanie niedeterministycznych wyborów do naszego modelu obliczeń nie pozwala na rozwiązywanie nowych problemów. Spróbujmy więc czegoś więcej. Wyobraźmy sobie, że matematyk próbuje przekonać swoją sceptyczną koleżankę, że potrafi udowodnić pewne twierdzenie. Przypuśćmy, że matematyk mówi: „Aby udowodnić twierdzenie, dowodzę lemat A, potem lemat B, a z nich otrzymuję twierdzenie jako wniosek C". Koleżanka mogłaby odpowiedzieć: „Dobrze, ale jak dowodzisz lemat B?". Matematyk na to: „Robię najpierw obserwację B.1, potem korzystam z ogólnego faktu B.2 i dostaję lemat jako wniosek B.3". Koleżanka: „W porządku, pokaż, jak dostajesz wniosek B.3". Matematyk podaje dowód wniosku. Po takiej wymianie zdań sceptyczna koleżanka przyjmuje, że matematyk faktycznie zna dowód twierdzenia, choć nie zobaczyła go w całości. Koleżanka wierzy bowiem, że matematyk nie tylko potrafił odpowiedzieć na pytania, które usłyszał, ale że ma kompletną *strategię* odpowiedzi na wszystkie możliwe pytania.

Taki sposób weryfikacji istnienia dowodu modelujemy za pomocą *alternujących* maszyn Turinga. Stany maszyny alternującej są podzielone na dwa zbiory: stany egzystencjalne, kontrolowane przez matematyka, oraz stany uniwersalne, kontrolowane przez jego sceptyczną koleżankę. Poza tym maszyna alternująca wygląda tak jak niedeterministyczna, tzn. obliczenie wymaga dokonywania wyborów spośród dostępnych kroków. Jeśli bieżący stan jest egzystencjalny, to matematyk wybiera kolejny krok obliczenia. Jeśli jednak stan jest uniwersalny, to kolejny krok wybiera sceptyczna koleżanka, a matematyk musi umieć poradzić sobie niezależnie od wyboru koleżanki. Maszyna udziela odpowiedzi *tak*, jeśli matematyk ma strategię prowadzącą do wyniku *tak*, niezależnie od wyborów koleżanki.

Czy w tak poszerzonym modelu da się rozwiązać więcej problemów? Okazuje się, że odpowiedź znowu jest negatywna. Dość łatwo jest symulować maszynę alternującą $\mathcal{A}$ za pomocą maszyny niedeterministycznej $\mathcal{N}$. Symulacja jest

Rozwiązanie zadania F 843.
Zgodnie z prawem Faradaya siła elektromotoryczna jest równa

$$\mathcal{E} = \frac{\Delta\phi}{\Delta t},$$

gdzie ϕ to strumień wektora indukcji magnetycznej. Z warunków zadania mamy $\phi = ktS$, a więc $\mathcal{E} = kS$, gdzie S jest polem powierzchni ograniczonej obręczą. Ze względu na symetrię zagadnienia wartość pola elektrycznego w każdym punkcie obręczy jest taka sama. Ponieważ wartość liczbowa siły elektromotorycznej indukcji jest z definicji równa wykonanej przez zewnętrzne źródło energii pracy potrzebnej na jednokrotny obieg obwodu przez jednostkowy ładunek elektryczny, więc zachodzi związek $\mathcal{E} = 2\pi r E$. Stąd

$$E = \frac{\mathcal{E}}{2\pi r}$$

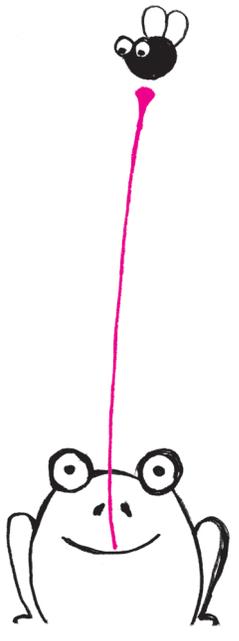
i ostatecznie

$$E = k\frac{\pi r^2}{2\pi r} = \frac{kr}{2}.$$

podobna do poprzedniej: na taśmie trzymamy częściowe rezultaty obliczeń $\mathcal{A}$, ale tym razem tylko niektóre. Mianowicie, jeśli w rozważanym częściowym rezultacie maszyna $\mathcal{A}$ jest w stanie egzystencjalnym, to maszyna $\mathcal{N}$ niedeterministycznie wybiera jeden z możliwych ruchów maszyny $\mathcal{A}$. Jeśli natomiast $\mathcal{A}$ jest w stanie uniwersalnym, to $\mathcal{N}$ wykonuje wszystkie możliwe ruchy w oddzielnych kopiach częściowego rezultatu, podobnie jak wcześniej. Maszyna $\mathcal{N}$ akceptuje, jeśli wszystkie częściowe rezultaty odpowiadają ukończonemu obliczeniu maszyny $\mathcal{A}$ z wynikiem *tak*.

Pożytki z niedeterminizmu i alternacji. Pokazaliśmy, że zarówno maszyny niedeterministyczne, jak i alternujące rozwiązują dokładnie te same problemy co maszyny deterministyczne. Czy to znaczy, że z niedeterminizmu i alternacji nie ma żadnego pożytku? Dotychczas interesowaliśmy się wyłącznie tym, czy odpowiedź na pytanie da się obliczyć, czy też nie. Jednak dużo istotniejsze jest pytanie, czy odpowiedź da się obliczyć szybko. Rozważmy następujący problem: spośród 400 uczniów należy wybrać 100, którzy będą spali w jednej dużej sali podczas wycieczki szkolnej, należy jednak przestrzegać reguły, aby pewne pary uczniów nie znalazły się jednocześnie w tej sali; lista takich zakazanych par jest również częścią danych początkowych. Jak można taki problem rozwiązać? Można rozważyć po kolei wszystkie możliwe podzbiory 100 uczniów i dla każdego zbioru sprawdzić, czy zawiera zakazaną parę, czy nie. Ale czy ta metoda jest praktyczna? Wymaga ona rozważenia $\binom{400}{100} \approx 2{,}2 \cdot 10^{96}$ podzbiorów. Ta liczba na papierze wygląda niegroźnie, ale wedle szacunków kosmologów wielokrotnie przekracza liczbę atomów w obserwowalnym wszechświecie. Trudno więc sobie wyobrazić, że tą metodą można uzyskać wynik nawet przy użyciu bardzo szybkiego komputera...

Z drugiej strony, niedeterministyczna maszyna Turinga mogłaby po prostu przejrzeć listę uczniów i niedeterministycznie zaznaczyć 100 spośród nich, a następnie sprawdzić, czy wybrany podzbiór nie zawiera żadnej zakazanej pary. Ta metoda wymaga przejrzenia listy uczniów raz dla każdej zakazanej pary, a więc nie więcej niż $\binom{400}{2} = 79800$ razy. Większości współczesnych komputerów nie zajęłoby to nawet sekundy. Kłopot polega na tym, że komputery są realizacją deterministycznej maszyny Turinga, a więc nie mogą po prostu niedeterministycznie wygenerować podzbioru uczniów.



Problem ustalenia, czy $\mathsf{P} = \mathsf{NP}$, jest jednym z kilku najważniejszych problemów otwartych współczesnej matematyki i prawdopodobnie najistotniejszym pytaniem w teoretycznej informatyce.

Pytanie brzmi więc: jak szybko można symulować maszyny niedeterministyczne (lub alternujące) za pomocą maszyn deterministycznych. Obie opisane przez nas symulacje są wykładnicze: jeśli maszyna niedeterministyczna o Q stanach wykonuje N kroków, to symulująca maszyna deterministyczna wykonuje mniej więcej Q^N kroków. Na przykład dla niedeterministycznej maszyny rozwiązującej problem wyboru 100 uczniów taka symulacja zajęłaby więcej czasu niż bezpośrednie przejrzenie wszystkich możliwych podzbiorów. Czy da się to zrobić lepiej? Konkretnie, czy da się zasymulować obliczenie niedeterministyczne długości N za pomocą deterministycznego obliczenia długości N^d, dla pewnej stałej d? Na to pytanie do dziś nikomu nie udało się udzielić odpowiedzi. Odpowiedź twierdząca oznaczałaby, że problemy rozwiązywalne w wielomianowo wielu krokach na maszynie niedeterministycznej da się rozwiązać w wielomianowo wielu krokach na maszynie deterministycznej, czyli że $\mathsf{P} = \mathsf{NP}$. Odpowiedź przecząca nie dawałaby bezpośrednich wniosków dotyczących problemu $\mathsf{P} = \mathsf{NP}$, ale byłaby nie mniej sensacyjna. Problemy rozwiązywalne na maszynie alternującej w wielomianowo wielu krokach tworzą tzw. klasę PSPACE (nazwa pochodzi od równoważnej charakteryzacji przez maszyny deterministyczne używające wielomianowej liczby komórek taśmy). O tej klasie również nie wiadomo, czy jest równa klasie NP, czy też jest od niej większa. Choć pytanie to jest mniej słynne, rozstrzygnięcie tego problemu byłoby również wielkim przełomem.