FILOZOFIA I NAUKA
Studia filozoficzne i interdyscyplinarne
Tom 1, 2013

Andrzej Wilk
Instytut Filozofii i Socjologii PAN

PRAWDA O PRZYSZŁOŚCI I POJĘCIE OBLICZEŃ

STRESZCZENIE

Tekst jest poświęcony problemowi wykorzystania teorii obliczalności w naukach empirycznych, które kierują swoją uwagę na zdarzenia przyszłe. Podstawowy rozważany problem jest taki: jak połączyć intuicyjne pojęcie nieprzewidywalności ze ścisłym pojęciem obliczalności? Centralną dla tej linii myślenia jest jednak następująca kwestia: czy świat realny można modelować na komputerze? Autor zakłada, po pierwsze, że wiedza o przyszłości magazynowana jest w zdaniach, po drugie, że jeden ze sposobów jej uzyskania polega na wykorzystaniu matematycznych formuł opisujących ewolucję w czasie, to jest równań ruchu (zdania o przyszłości są więc zdaniami o położeniach obiektów w przyszłości). Cel jest ujęty trywialnie: pokazać, że zdania o przyszłości mogą być formułowane w sposób, który wymaga zaangażowania matematyki niealgorytmicznej. Bardziej precyzyjnie: pokazać, że dla każdego równia ruchu i wszelkich możliwych warunków początkowych nie istnieje program, który odpowiada tak/nie na pytanie, czy równanie posiada okresowe rozwiązania, czy nie. Schemat rozumowania jest następujący: zakładamy (*reductio od absurdum*), że maszyna Turinga wyposażona jest w program, który wycina okresowość w rozwiązaniach równań ruchu. Postulat testowania okresowości bierze się stąd, że jej obecność świadczy o efektywności operacji/funkcji. Jest jasne, że warunki początkowe muszą być obliczalne/rekurencyjne. Z tego względu, że zbiór liczb rzeczywistych jest nieskończony/nieprzeliczalny możemy dopuścić, że któraś operacja (funkcja) z ich udziałem będzie w końcu nieefektywna (rezultaty Banacha/Mazura, Turinga, Pour-El/Richardsa, Chaitina, Battermana). To upoważni nas do wykorzystania „twierdzenia o stopie" dla maszyny Turinga i w rezultacie do stwierdzenia, że klasa równań ruchu jest nierozstrzygalna.

1. WSTĘP

W myśleniu o przyszłości obecne jest silne poczucie dyskomfortu, które ma źródło w tym, że na różne sposoby próbujemy przyszłość przewidzieć i najczęściej się okazuje, że cel nie został osiągnięty. Obowiązuje też reguła,

że mylimy się tym bardziej, im dalej wybiegamy w przyszłość, czyli gdy predykcje dotyczą dłuższych okresów czasu. Dlatego mówimy, że przyszłość jest z gruntu nieprzewidywalna. Moim celem jest dodanie do intuicyjnego rozumienia nieprzewidywalności przyszłości tych znaczeń, które wiążą się z pojęciem nieefektywnych obliczeń w sensie deterministycznej maszyny liczącej. Zakładam, rzecz jasna, że dwie płaszczyzny refleksji nad nieprzewidywalnością, intuicyjna i obliczeniowa, nie pozostają ze sobą w sprzeczności, a wręcz przeciwnie, uzupełniają się. Skupiając się na obliczeniowej stronie fenomenu będę starał się pokazać, że:

> Formułując zdania o przyszłości w ten sposób, że narzędziem rozstrzygania ich prawdziwości jest maszyna matematyczna wraz z wpisanymi w nią równaniami opisującymi zmianę położeń obiektów w czasie trzeba uwzględnić logiczną nieefektywność.

Wspomnieć tu należy o następującym problemie. Otóż niektórzy filozofowie i logicy są zdania, że rozważania o „prawdzie przyszłej" należy odseparować od ontologii, ponieważ prowadzenie ich równolegle ze sporem determinizm-indeterminizm może inspirować rewizję logiki klasycznej (dla Łukasiewicza wartość ½ ma właśnie wyraźny „metafizyczny posmak", bo ilustruje rzeczywistość pozajęzykową, czyli zjawisko niezdeterminowania zdarzeń).[1] Podejście takie warte jest uwagi choćby z tego powodu, że preferowane tu spojrzenie na przyszłość zakładające wykorzystanie formuł matematycznych opisujących zmianę położeń obiektów w czasie podpowiada stosowanie logiki klasycznej, a ta jest tradycyjnie dwuwartościowa i autor będzie się tego trzymać. Z drugiej strony, równania ruchu wykorzystujemy do opisu świata fizycznego, a w tym kontekście ucieczka od refleksji metafizycznej może być kwestionowana. Tylko, że czyniąc zadość ostatniej sugestii, narażamy się znów na niebezpieczeństwo, że zbyt długie pozostawanie pod jej wpływem pociągnie próbę zreformowania logiki. Wyjściem z kłopotów może być izolacja tej części metafizycznej spekulacji, która ma niewygodne dla logiki klasycznej implikacje.

Odnośnie prawdy i metod jej rozstrzygania, to przyjęciu założenia, że zdanie o przyszłości jest już dzisiaj prawdziwe bądź fałszywe, towarzyszyć będzie przekonanie, że można podważyć efektywność pewnego typu procedur, które uruchamiamy usiłując te wartości rozpoznać. Dokładnie rzecz biorąc, idzie tu o ten rodzaj mechanicznych postępowań, które skutkują nieobliczalnością w rozumieniu deterministycznej maszyny Turinga. Zaistnienie operacji o takiej charakterystyce będzie w rezultacie prowadziło do stwierdzenia, że

> dla każdego równania ruchu i wszystkich możliwych warunków początkowych nie istnieje uniwersalny algorytm, który odpowiada `tak/nie` na pytanie, czy równanie (układ równań) posiada okresowe rozwiązania, czy nie.

---

[1] G. Malinowski, *Logiki wielowartościowe*, PWN, Warszawa 1990, s. 32.

Rozumowanie, które ma to pokazać, jest następujące: zakładamy (*reductio ad absurdum*), że maszyna Turinga rozwiązując dowolne równanie ruchu, wyposażona jest w program, który wycina okresowość w rozwiązaniach. Postulat testowania okresowości bierze się stąd, że jej obecność świadczy o obliczalności operacji (okresowość rozwinięcia liczby rzeczywistej jest spektakularnym przykładem obliczalności). Wiadomo, że warunki początkowe kodowane w maszynie muszą być obliczalne. Od niealgorytmicznych ciągów symboli binarnych maszyna z definicji nie startuje. Z tego znów powodu, że jest ich nieprzeliczalnie wiele, jest pewność, że któraś operacja z ich udziałem będzie w końcu nieefektywna. W artykule *Obliczalność a świat realny*[2] w tym miejscu rozumowania pojawiły się wyniki z obszaru tak zwanej obliczalnej analizy, Banacha i Mazura[3] oraz Pour-El i Richardsa.[4] Teraz potrzebną nieobliczalność wyprowadzimy z teorii chaosu, *via* teoria algorytmicznej informacji. To pozwoli nam wykorzystać twierdzenie o (rekurencyjnej) nierozwiązywalności problemu stopu dla maszyny Turinga i w konsekwencji orzec, że klasa równań ruchu jest nierozstrzygalna. W odniesieniu do co najmniej jednego równania nie istnieje bowiem program stwierdzający, czy posiada okresowe rozwiązania, czy nie.

Efekty przedsięwzięcia polegającego na wykazaniu, że zdania o przyszłości mają często postać niealgorytmiczną, mogą stać się głosem w sporze między realizmem a antyrealizmem semantycznym. Może się bowiem okazać, że w niektórych przypadkach łatwo daje się określić tak zwane warunki prawdziwości matematycznej wypowiedzi o przyszłości, ale nie sposób tej wypowiedzi dowieść, czyli efektywnie zrealizować operacji (obliczania), którą ta wypowiedź zakłada. Fakt istnienia analitycznej nieobliczalności powinien też prowadzić do wniosku, że tak zwany świat realny zawiera zadziwiający „bezprawny" obszar, który opiera się cyfrowej reprezentacji. Taki stan rzeczy jest oczywiście motywem poszukiwania niestandardowych sposobów liczenia, głównie komputingu analogowego, choćby w postaci *neural nets*. Trzeba jednak podkreślić, że oczekiwane sukcesy obliczania analogowego nie przeszkadzają w sformułowaniu generalnego wniosku, że efektywna cyfrowa komputacja stoi na przecięciu wszystkich możliwych modeli *Uniwersum*. Dziedzina, w której daje się zrealizować, jest rozległa, a decydujące jest to, że w matematycznych postępowaniach, które odsłaniają anomalię nieefektywności punktem wyjścia są zawsze obiekty podlegające obliczalnej cyfrowej deskrypcji.

---

[2] Artykuł ten zostanie opublikowany w następnym tomie czasopisma FILOZOFIA A NAUKA.

[3] S. Mazur, *Computable Analysis*, Rozprawy Matematyczne XXIII, PWN, Warszawa 1963.

[4] M.B. Pour-El, J. I. Richards, *Computability in Analysis and Physics*, Springer, Berlin–Heidelberg 1989.

2. REALIZM SEMANTYCZNY

Refleksja nad sporem realizm-antyrealizm semantyczny dotyczy jego matematycznej wersji, ponieważ odpowiada to postawionemu tu zadaniu. Przypomnijmy: jest nim wykazanie, że zdania o przyszłości można formułować z użyciem deterministycznego komputera przy zakodowanych w nim formułach opisujących ewolucję w czasie, wraz z warunkami początkowymi. Oznacza to, że korzystamy z maszyny matematycznej o określonym zbiorze (rekurencyjnych/deterministycznych) instrukcji, wraz z (rekurencyjnymi) liczbami rzeczywistymi reprezentującymi warunki początkowe. Innymi słowy, chcemy efektywnie *obliczyć/dowieść* przy pomocy maszyny, że przyszłość będzie taka a taka. Celowo pomijamy tu tę odmianę realizmu/antyrealizmu semantycznego, która akcentuje empiryczno-weryfikacyjną stronę zagadnienia. Można przypuszczać, że jeżeli wyżej określone zadanie zostanie zrealizowane, to będzie głosem w sporze. Co ważne, będzie nim głównie wtedy, gdy się okaże, że przyszłości nie można efektywnie obliczyć.

Jak wiemy, intuicjonizm zaproponował

> ...nową odpowiedź na pytanie, czym jest rozumienie zdania matematycznego; nie ma być ono znajomością tego, na czym polega prawdziwość owego zdania, niezależnie od możliwości ustalenia przez nas jego prawdziwości czy fałszywości, lecz znajomością tego, co jest wymagane, by je dowieść. Ujmując to we wcześniej wprowadzonych terminach, zdania matematyczne należy interpretować jako deklaracje, a nie jako afirmacje.[5]

Antyrealiści rozszerzyli dziedzinę, w której powinna funkcjonować intuicjonistyczna dyrektywa znaczeniowa, o zdania niematematyczne. Nie muszą to być wyłącznie zdania o faktach; mogą to być zdania o przedmiotach intencjonalnych, a nawet sprzecznych. Składową matematyczną antyrealizmu dobrze ilustrują słowa Heytinga (ucznia Brouwera, twórcy intuicjonizmu):

> ...Dowód jakiegoś stwierdzenia jest konstrukcją matematyczną, którą z kolei można traktować w sposób matematyczny. Intencja takiego dowodu prowadzi zatem do nowego zdania. Jeżeli oznaczyć przez $+p$ zdanie: "zdanie p jest dowodliwe", to $+$ jest funkcją logiczną, mianowicie dowodliwością. Stwierdzenia (Behauptungen) $> p$ i $> +p$ mają to samo znaczenie. Istotnie, jeśli p jest udowodnione, to udowodniona jest też dowodliwość p, a gdy udowodnione jest $+p$, to została spełniona intencja [podania] dowodu p, to znaczy p zostało udowodnione. Mimo to zdania p i $+p$ nie są identyczne, co najlepiej pokazać na przykładzie. Przy obliczaniu stałej Eulera C może się zdarzyć, że pewna liczba wymierna, powiedzmy A, zawarta jest nadzwyczaj długo

[5] M. Dummett, *Realism and Anti-realism*, w: M. Dummett, The Seas of Lanquage, Clarendon Press, Oxford 1993, s. 462–478, przekład: *Realizm i antyrealizm*, w: *Filozofia brytyjska u schyłku XX wieku*, red., P. Gutowski, T. Szubka, TN KUL, Lublin 1998, s. 77.

w przedziale, w którym przybliżamy coraz bardziej C, tak że dochodzimy w końcu do przypuszczenia, że C jest równe A, to znaczy oczekujemy, że przy dalszych obliczeniach ciągle znajdować się będziemy w naszym przedziale A. Przypuszczenie takie nie jest jednak żadną miarą dowodem, że tak zawsze będzie. Zdanie $+(C = A)$ mówi więcej niż zdanie $(C = A)$.[6]

Co innego zatem znaczy „wiedzieć, na czym polega prawdziwość zdania matematycznego", a co innego znaczy „znać/przedstawić jego dowód". Należy odróżnić tak zwane warunki prawdziwości od tak zwanych warunków stwierdzalności. Dla antyrealisty przedstawienie (konstruktywnego) dowodu zdania matematycznego stanowi coś więcej niż postulowanie jego prawdziwości bez możliwości jej ustalenia. Dla realisty prawda przekracza znów możliwości jej rozpoznania. Jeżeli uwzględnimy racje antyrealisty, to otrzymamy wniosek, że matematycy przez trzysta lat nie rozumieli słynnej nierówności Fermata, bo zdanie to zostało udowodnione dopiero w latach dziewięćdziesiątych XX wieku. Przykład ten jest też ciekawy z tego względu, że przed podaniem pełnego dowodu istniały dowody cząstkowe. Wszyscy zainteresowani mieli przy tym jakieś zrozumienie luk, które trzeba zapełnić. Wracając do problemu algorytmiczności zdań o przyszłości formułowanych przy pomocy maszyny, to może się okazać, że zapętlenie w które wpada program komputerowy uniemożliwia efektywne obliczenie przyszłości. Fakt taki byłby zatem równoważny brakowi konstruktywnego dowodu (dowodu, że przyszłość będzie tak a nie inaczej skonfigurowana). Stąd musimy liczyć się z tym, że prawdziwość niektórych zdań o przyszłości nie zostanie nigdy rozstrzygnięta. Chyba, że empirycznie, gdy uzyskają one w końcu kwalifikacje zdań o teraźniejszości.

3. PRAWDA O PRZYSZŁOŚCI I POJĘCIE MECHANICZNEJ PROCEDURY

Ogólnie ujmując, mechaniczna procedura to zbiór reguł, które porządkują jakąś aktywność gwarantując przy tym jej zakończenie. Łatwo zauważyć, że żądanie zakończenia, czyli postulat efektywności, ma tu podstawowe znaczenie. Procedury nieefektywne, więc ciągnące się w nieskończoność, nie są procedurami w sensie właściwym. To samo powiedzieć trzeba o algorytmach. Uważa się, że intuicyjnie oczywiste pojęcie mechanicznej procedury/algorytmu nie daje się zadowalająco sformalizować. Próbą jego formalizacji był, jak wiadomo, projekt maszyny matematycznej Turinga. Można podzielić się jednak refleksją, że pojęcie mechanicznej procedury jest z pewnością nieformalne i nieprecyzyjne, ale jedynie w kanonicznym (rekurencyj-

[6] A. Heyting, *Die intuitionistische Grundlegung der Mathematik*, Erkenntnis, 1931, t. 2, s. 106–115; tłum., *Intuicjonistyczne podstawy matematyki*, w: *Współczesna filozofia matematyki*, red., R. Murawski, Wydawnictwo Naukowe PWN, Warszawa 2002, s. 68.

nym) matematycznym sensie. Model Turinga nie zawiera w gruncie rzeczy nic nieprecyzyjnego: mamy pola z symbolami, czytnik, stan aktywny, stan pasywny, zbiór deterministycznych instrukcji... Co najważniejsze, żadnego marginesu dla dowolności. Nie ma więc mowy o jakiejś przednaukowej „intuicji", czy przeświadczeniu, tylko raczej o zapisie, który nie jest podręcznikowo rekurencyjny. Może być jednak i tak, że dla następnych pokoleń reprezentacja maszynowo-algorytmiczna będzie czymś bardziej naturalnym niż rekurencyjna.

3.1. Algorytm

Nieformalnie rzecz biorąc, algorytm to zbiór zasad niezbędnych do wykonania zadania obliczeniowego. A bardziej precyzyjnie:

Algorytm to skończonego rozmiaru abstrakcyjny przepis zarządzający procesem, który może być przeprowadzony mechanicznie przez człowieka lub maszynę liczącą w skończonym czasie.

Efektywne obliczenia albo efektywna procedura C jest mechaniczną procedurą, która może być zastosowana dla pewnej klasy symboli wejściowych s, i która ewentualnie dostarczy dla nich symboliczne wyjście t. Proces ten można opisać wzorem $C(s) = t$. Algorytm nie musi być zdefiniowany dla wszystkich wartości wejściowych. W takim przypadku procedura ciągnie się w nieskończoność. Zbiega się to z faktem, że istnieją funkcje częściowo-rekurencyjne. Chociaż sekwencje wejściowe i wyjściowe, jak również rozmiar pamięci wewnętrznej muszą być ograniczone, to nie ma specjalnych restrykcji odnośnie ich rzeczywistego rozmiaru. To samo stosuje się do długości deskrypcji algorytmu oraz czasu jego egzekucji; zdarza się, że relatywnie „małe" programy wymagają „dużego" czasu wykonania i produkują „olbrzymie" wyjście. Przedstawione powyżej definicje algorytmu i efektywnych obliczeń są oczywiście nieformalne. Kwestia, czy istnieją formalne (matematyczne) pojęcia korespondujące z tymi heurystycznymi przybliżeniami, wydaje się być kluczowa. Szczególnie idzie tu o termin „mechaniczna procedura", który powinien być ściśle formalnie zdefiniowany. Dodajmy, że „ściśle" w sensie zastanych matematycznych konwencji. Pozytywną odpowiedź daje teza Churcha-Turinga stanowiąc, że klasa funkcji określonych przy pomocy intuicyjnego pojęcia „efektywnej mechanicznej procedury" jest identyczna z klasą funkcji (częściowo) rekurencyjnych. Najbardziej znaną mechaniczną procedurą jest algorytm Euklidesa, przy pomocy którego po wykonaniu skończonej ilości operacji można znaleźć na przykład największy wspólny dzielnik dowolnej pary liczb naturalnych. Po każdym kolejnym kroku oczywistym jest tu krok następny. Oczywiste jest również zakończenie procedury. Daje się ją przy tym opisać w skończony sposób jako skończony zbiór dyrektyw, mimo że stosuje się do wszystkich liczb naturalnych.

3.2. Prawda o przyszłości i równania ruchu

W prowadzonych tu rozważaniach przyjęto, że wzory matematyczne opisujące ewolucję w czasie, czyli równania ruchu, odpowiadają za transmisję prawdy w czasie. Oparto się zatem na dość trywialnej konstatacji, że jeżeli ktoś oblicza położenia obiektu w przyszłości, to w istocie buduje zdanie o przyszłości. Zdanie, które okaże się prawdziwe wtedy, gdy dobrze dokona pomiaru warunków początkowych oraz działań, a fałszywe wtedy, gdy przynajmniej jedna z tych czynności będzie niepoprawna. Sama formuła opisująca dynamikę jest z założenia prawdziwa, to jest adekwatnie odzwierciedla porządek rzeczy (kształt zmian). Prawda peregrynuje więc od zdań o warunkach początkowych do zdań o przyszłych położeniach obiektów. Podobnie z fałszem. Zdania o przyszłości mogą mieć postać

S_1 – obiekt o_1 znajduje się w miejscu p_1 w czasie t_1

S_2 – obiekt o_2 znajduje się w miejscu p_2 w czasie t_2

S_n – obiekt o_n znajduje się w miejscu p_n w czasie t_n

a ich zbiór $(S_1,....,S_n)$ jest z pewnością nieskończony i nieprzeliczalny. Tak samo jak zbiór liczb rzeczywistych, które reprezentują warunki początkowe. Mogą mu jednak przysługiwać jeszcze inne cechy, gdyby się na przykład okazało, że klasa równań ruchu nie jest rozstrzygalna.

4. ROZSTRZYGALNOŚĆ (*ENTSCHEIDUNGSPROBLEM* HILBERTA)

Zagadnienie rozstrzygalności związane jest z nazwiskiem niemieckiego matematyka D. Hilberta i jego programem rekonstrukcji matematyki w terminach czysto syntaktycznych. Charakterystycznym dla tego programu było takie rozumienie prawdy matematycznej, że funkcjonuje ona jedynie w granicach systemu formalnego, który podlega deskrypcjom finitarnym.[7] Każdy dowód matematyczny miał być skończoną manipulacją na liście symboli (w szczególności dotyczyć to miało postulowanego dowodu niesprzeczności matematyki).[8] Formalizacja, która zajmowała centralną pozycję w programie, miała eliminować, z jednej strony, konieczność operowania pojęciem znaczenia zdania matematycznego, z drugiej strony, intuicję.

System formalny S jest systemem symboli wraz z zasadami ich użycia. Symbole indywidualne są elementami alfabetu. Formuły są sekwencjami symboli. Należy zdefiniować klasę formuł, które nazywa się formułami dobrze uformowanymi, oraz klasę dobrze uformowanych formuł, które nazywa

[7] Za finitarne Hilbert uznawał te rozwiązania, które dziś nazywamy efektywnymi, algorytmicznymi.
[8] Hilbert żądał, by twierdzenia, które wcześniej zostały udowodnione środkami niekonstruktywnymi uzyskały dowód w znaczeniu konstruktywnym (finitarnym).

się aksjomatami (może ich być skończona albo nieskończona ilość). Dalej, trzeba określić listę zasad, które nazywa się zasadami inferencji. Jeżeli zasadę inferencji nazwiemy R, to będzie ona definiowała relację bezpośredniej konsekwencji R' między zbiorem dobrze uformowanych formuł $M_1,....,M_n$, czyli założeniami, a formułą F, czyli konkluzją (twierdzeniem).

Jeżeli użyjemy języka nauk komputerowych, to system formalny powinien być implementowany jako skończonych rozmiarów program komputerowy, który na wyjściu dostarcza wszystkie prawdziwe twierdzenia matematyczne. Z pojęciem hipotetycznego programu komputerowego, który generuje wszystkie prawdy matematyczne, blisko związanie jest pojęcie uniwersalnej procedury rozstrzygania, czy konkretna sekwencja symboli posiada dowód, czy nie. Aby przybliżyć to drugie pojęcie, zacytujmy Tarskiego:

> Przez procedurę rozstrzygania dla danej sformalizowanej teorii T rozumiemy metodę, która pozwala nam rozstrzygnąć w każdym poszczególnym przypadku, czy dane zdanie, sformułowane w języku teorii T, da się udowodnić przy użyciu środków dostępnych w T (lub ogólniej, może być rozpoznane jako ważne w T). Problem rozstrzygalności dla T to problem udzielenia odpowiedzi na pytanie, czy istnieje procedura rozstrzygania dla T (i ewentualnie kwestia pokazania takiej procedury). Teorię T nazywamy rozstrzygalną bądź nierozstrzygalną w zależności od tego, czy problem rozstrzygalności dla niej ma rozwiązanie pozytywne, czy negatywne.[9]

Tarski nie używa tu określeń „uniwersalna", ani „mechaniczna", ale mając na uwadze to, że Hilbert chciał zmechanizować całą matematykę, mamy podstawowe znaczenie pojęcia „uniwersalnej procedury rozstrzygania". Sądzi się, że istnienie uniwersalnej procedury dowodowej znosiłoby zasadność aplikacji uniwersalnej procedury rozstrzygania. Na pozytywny aspekt badań nad rozstrzygalnością zwraca uwagę Tarski przedstawiając następujący przypadek: mamy maszynę do dowodów oraz maszynę do rozstrzygnięć. Maszyna do dowodów produkuje zdania dowodliwe w czasie skończonym, ale nie dającym się z góry oszacować, maszyna do rozstrzygnięć odpowiada `tak/nie`, czy dane zdanie jest dowodliwe, w czasie skończonym i dającym się z góry oszacować. Wyobraźmy sobie teraz sytuację, że dysponujemy wyłącznie maszyną do dowodów. W rezultacie:

> ...dla dowolnego danego twierdzenia, jesteśmy pewni, że w skończonym okresie czasu, maszyna ta wyprodukuje to zdanie lub jego negację i przez to odpowie nam, czy to zdanie jest dowodliwe, czy nie. Jednakże nie da się z góry oszacować czasu potrzebnego na to. W ten sposób, spotykamy się z przypad-

---

[9] A. Tarski, A. Mostowski, R. M. Robinson, *A General Method in Proofs of Undecidability*, w: *Undecidable Theories*, North-Holland Publishing Comp., 1–30, Amsterdam 1953; tłum.: *Ogólna metoda dowodów nierozstrzygalności*, w: A. Tarski, *Pisma logiczno-filozoficzne*, t. 2, *Metalogika*, red. J. Zygmunt, Warszawa 2001.

kiem, w którym nie jest jasne, czy mamy pozytywne rozwiązanie problemu rozstrzygalności.[10]

Tarski akcentuje tu trudności realizacji, ale co innego oznacza dysponować procedurą, która dostarcza dowód dla każdej formuły danej teorii, a co innego oznacza korzystać z metody, która dla danej teorii sprawdzi, czy każda formuła posiada dowód, czy nie. Można sprawdzać rozstrzygalność danej teorii/klasy nie oferując zarazem metody dowodzenia dla każdego zdania tej teorii/klasy. Przykładem jest proponowany sposób testowania rozstrzygalności klasy równań ruchu, który nie jest przecież sposobem dowodzenia tych równań. Ogólnie rzecz ujmując, istnieją zadania, które nie ograniczają się do udzielenia odpowiedzi tak/nie. Należy choćby podać wartościowanie, które spełnia formułę logiczną, a nie tylko odpowiedzieć, że ono istnieje. Pierwsze zadanie nazywa się zadaniem funkcyjnym, drugie zaliczyć należy do grupy zadań decyzyjnych. Zadania decyzyjne zastępują zadania funkcyjne jedynie w przypadku wyników negatywnych. Łatwiej też pokazać, że jakaś teoria jest nierozstrzygalna niż, że jest rozstrzygalna. W pierwszym przypadku trzeba udowodnić, że zbiór jej twierdzeń jest rekurencyjnie przeliczalny, w drugim, że jest rekurencyjny. Nierozstrzygalna może być teoria, problem, bądź zdanie. Teorią nierozstrzygalną jest np. arytmetyka liczb naturalnych, z mnożeniem i dodawaniem. Problemem nierozstrzygalnym jest problem równań diofantycznych, czy też problem stopu dla maszyny Turinga. Zdaniem nierozstrzygalnym jest jedno z równań diofantycznych, albo jedno ze zdań Gödla. Krótko, jeżeli w obszarze pewnej klasy zagadnień istnieje przynajmniej jedna kwestia, co do której nie można udzielić odpowiedzi tak/nie, to klasa jest nierozstrzygalna. Odnośnie klasy równań ruchu również daje się skonstruować pytanie testujące jej rozstrzygalność/nierozstrzygalność. Może być ono takie: czy dla każdego równania ruchu i wszelkich możliwych warunków początkowych istnieje program separujący przypadki, gdy rozwiązania są okresowe od pozostałych? Należy podkreślić, że nie jest to pytanie o dowód dla równania ruchu. Jeżeli będzie miało się odnosić do własności posiadania dowodu, to będzie brzmieć tak: czy można dowieść, że dowolne równanie ruchu przy dowolnych warunkach początkowych nie posiada okresowych rozwiązań?

4.1. Liczby nieobliczalne (rozumowanie diagonalne)

Nieformalnie ujmując, liczba rzeczywista jest obliczana wtedy, gdy zbliża się do pewnego stopnia precyzji dzięki zadanemu z góry programowi. Liczba π jest obliczalna, bo istnieje skończony algorytm generujący jej rozwinięcie.

[10] A. Tarski, *Remarks of Alfred Tarski*, Revue Internationale de Philosophie, 27–8, s. 16–0; przedruk w: A. Tarski, *Collected Papers*, v. 1–2, rd. by J. McKenzie and Givant S., Birkhäuser Verlag, Basel 1986; cytat z: J. Woleński, *Epistemologia*, PWN, Warszawa 2005, s. 265.

Jeżeli pożądany jest bardzo wysoki stopień precyzji, to obliczanie może trwać ekstremalnie długo, lecz algorytm się nie zmienia. Aby wprowadzić nieobliczalne liczby rzeczywiste, trzeba wcześniej wprowadzić liczby rzeczywiste. Jedna z metod opiera się na spostrzeżeniu, że każda liczba rzeczywista r jest granicą pewnego ciągu $\{r_n\}$ liczb wymiernych. Utożsamia się więc liczby rzeczywiste z ciągami do nich zbieżnymi. Są dwa aspekty efektywizacji konwergencji:

1. Ciąg liczb wymiernych musi być obliczalny. Czyli, że musi być obliczalny przez skończony zbiór instrukcji zadany z góry.

2. Zbieżność tego ciągu do granicy musi być efektywna.

Funkcję (stałą lub zmienną) traktuje się jako ciąg wtedy, gdy przedmiotem zainteresowania są jej własności dla argumentów będących liczbami naturalnymi. Teoria ciągów liczb rzeczywistych jest więc teorią funkcji w twierdzeniach której wszystkie zmienne występujące jako argumenty w wyrażeniach funkcyjnych są zrelatywizowane do N. Funkcja α dla argumentów naturalnych może dać ciąg rosnący, malejący, monotoniczny, zbieżny w sensie Cauchy'ego, mający granicę, itd. Dowodząc istnienia liczb rzeczywistych korzysta się z metod niekonstruktywnych (argument diagonalny). Przykładem zastosowania metody niekonstruktywnej jest dowód następującego twierdzenia:

TWIERDZENIE 1. Istnieją liczby niewymierne $x, y \in R - Q$ takie, że $x^y \in Q$.

(R – zbiór liczb rzeczywistych, Q – zbiór liczb wymiernych)

Dowód: przypadek 1. $\sqrt{2}^{\sqrt{2}} \in Q$; przypadek 2. $\sqrt{2^{\sqrt{2}}} \notin Q$, wtedy $\sqrt{2}^{\sqrt{2}^{\sqrt{2}}} = 2 \in Q$.

Kwestia, który przypadek zachodzi, czyli która liczba jest wymierna, pozostaje nierozstrzygniętą. Istnienie nieobliczalnej liczby rzeczywistej, czyli liczby której rozwinięcie nie jest obliczalne sukcesywnie w wyniku efektywnych obliczeń, można wykazać dysponując wpierw definicją obliczalnej liczby rzeczywistej.

DEFINICJA 1. Liczba rzeczywista x jest obliczalna wtedy, gdy istnieje obliczalny ciąg liczb wymiernych, który jest efektywnie zbieżny do x.

DEFINICJA 2. (efektywna zbieżność). Ciąg $\{r_n\}$ liczb wymiernych jest efektywnie zbieżny do liczy rzeczywistej x wtedy, gdy istnieje funkcja rekurencyjna $\varepsilon : \mathrm{N} \to \mathrm{N}$ taka, że dla każdego $n \in \mathrm{N}$
$k \geq \varepsilon(n)$ pociąga $|r_k - x| \leq 2^{-n}$

DEFINICJA 3. (obliczalny ciąg liczb wymiernych). Ciąg $\{r_n\}$ liczb wymiernych jest obliczalny wtedy, gdy istnieją trzy funkcje rekurencyjne $a(n), b(n), c(n)$ z N do N takie, że

$$r(n) = (-1)^{c(n)} \frac{a(n)}{b(n)}$$

TWIERDZENIE 2. Niech x będzie obliczalna liczbą rzeczywistą. Jeżeli $x > 0$, to istnieje efektywna procedura, która to pokazuje. Podobnie dla $x < 0$. Jeżeli $x = 0$, to nie istnieje efektywny sposób pokazania tego.[11]

Istnieją przypadki, gdy konwergencja nie jest efektywna, wtedy potrzebne są dwa preliminaryjne fakty:

TWIERDZENIE 3. Niech $a : \mathrm{N} \to \mathrm{N}$ będzie jedno-jednoznaczną funkcją rekurencyjną generującą rekurencyjnie przeliczalny non-rekurencyjny zbiór A. Niech $w(n)$ denotuje „czas oczekiwania"

$$w(n) = \max\{m : a(m) \le n\}$$

wtedy nie istnieje funkcja rekurencyjna c taka, że $w(n) \le c(n)$ dla każdego n.[12]

TWIERDZENIE 4. Niech $a : \mathrm{N} \to A \subset \mathrm{N}$ będzie jedno-jednoznaczną funkcją rekurencyjną generującą rekurencyjnie przeliczalny ale non-rekurencyjny zbiór A. Rozważmy szereg

$$s_k = \sum_{m=0}^{k} 2^{-a(m)}$$

i niech $x = \lim_{k \to \infty} S_k$. Wtedy czas oczekiwania $w(n)$ jest najmniejszą liczbą całkowitą taką, że $k \ge w(n)$ implikuje $x - s_k \le 2^n$.

To pociąga, że wcześniej zdefiniowana liczba rzeczywista x jest obliczalnym ciągiem liczb wymiernych $\{s_k\}$, który jest zbieżny nieefektywnie. Fakt ten jest zgodny z następującymi wnioskami:

(i) Istnieją nieobliczalne liczby rzeczywiste korespondujące z nieefektywną konwergencją obliczalnego ciągu liczb wymiernych. Chociaż ciąg s_k jest obliczalny, to nie można powiedzieć, że granica x jest obliczalna, bo zbieżność jest nieefektywna. Chaitin pokazał *LISP* algorytm do obliczania Ω, którego konwergencja jest „bardzo słaba". Tylko w takim sensie (w granicy

---

[11] Dowód w: M. B. Pour-El, J. I. Richards, *Computability in Analysis and Physics*, Springer, Berlin–Heidelberg 1989, s. 14.

[12] Dowód twierdzenia. w: Pour-El, Richards, op. cit., s. 15–16.

nieskończonego czasu) jest możliwe obliczanie ciągu o nieograniczonej złożoności, w szczególności ciągu przypadkowego.[13]

(ii) Istnieją funkcje *signum* x oraz *integer part* x, które wyprowadzają poza klasę ciągów liczb rzeczywistych obliczalnych.[14]

(iii) Maksymalna/minimalna wartość funkcji jest obliczalna, ale punkt/punkty gdzie to minimum/maksimum występuje, nie musi być obliczalny. Koniecznym warunkiem jest obecność nieskończenie wielu punktów max/min. Jeżeli funkcja obliczalna przybiera lokalne max/min w izolowanym punkcie, to punkt jest obliczalny.[15]

(iv) Różniczkowe równania ruchu mogą mieć nieobliczalne rozwiązania (słabe) z obliczalnych wartości początkowych.[16]

(iiv) Ograniczone liniowe operatory w przestrzeni Banacha zachowują obliczalność, ale nieograniczone nie zachowują obliczalności.[17]

Teraz można przeprowadzić (diagonalne) rozumowanie, które jest zarazem dowodem na istnienie nieobliczalnej liczby rzeczywistej.

TWIERDZENIE 5. Zbiór rekurencyjnych liczb rzeczywistych nie jest rekurencyjnie przeliczalny. Nie istnieje efektywna enumeracja wszystkich liczb rzeczywistych.

Dowód przez diagonalizację. Zakładamy (nie wprost), że istnieje efektywnie obliczalna enumeracja rekurencyjnych liczb rzeczywistych [0,1].

$$r_1 = 0, r_{11} r_{12} r_{13} r_{14} \ldots$$
$$r_2 = 0, r_{21} r_{22} r_{23} r_{24} \ldots$$
$$r_3 = 0, r_{31} r_{32} r_{33} r_{34} \ldots$$
$$r_4 = 0, r_{41} r_{42} r_{43} r_{44} \ldots$$

Utwórzmy teraz liczbę z diagonalnych elementów $0, r_{11} r_{22} r_{33} r_{44} \ldots$ Następnie zmieńmy każdą z tych cyfr w taki sposób, aby uniknąć 0 i 9. Jest to konieczne, ponieważ dwie liczby rzeczywiste o różnych rozwinięciach są identyczne, gdy jedna kończy się nieskończoną sekwencją 9, a druga 0, na przykład 0,09999...= 0,10000... Liczba $r' = r_1', r_2', r_3', r_4', \ldots$, przy $r' \neq r_{nn}$, różni się od każdej innej z listy na przynajmniej jednej (diagonalnej) pozycji. Dlatego istnieje co najmniej jedna liczba, która wypada z oryginalnej enumeracji. Skoro założenie wyjściowe jest poprawne, a wszystkie operacje, które zawiera argument diagonalny są obliczalne, to skonstruowana liczba powinna również być obliczalna i pojawić się na liście. Ale wcześniej otrzymaliśmy

---

[13] G. J. Chaitin, *Algorithmic Information Theory*, Cambridge Univ. Press, Cambridge 1987.
[14] S. Mazur, *Computable Analysis*, Rozprawy Matematyczne XXIII, PWN, Warszawa 1963.
[15] Pour-El, Richards, op. cit., s. 42.
[16] Ibidem, s. 73.
[17] Ibidem, rozdz. 3.

wiadomość, że nie jest zawarta w oryginalnej enumeracji. Mamy zatem sprzeczność. Konkluzja płynie więc jedna: nie istnieje efektywnie obliczalna enumeracja rekurencyjnych liczb rzeczywistych. Zbiór rekurencyjnych liczb rzeczywistych nie jest rekurencyjnie przeliczalny. Rozumowanie przekątniowe wykorzystał Cantor dowodząc, że nie ma jedno-jednoznacznej relacji między liczbami naturalnymi a liczbami rzeczywistymi. Tych pierwszych jest przeliczalnie wiele a tych drugich jest nieprzeliczalnie wiele.

4.2. Równania diofantyczne (obliczanie = wielomianowe równanie diofantyczne)

Dziesiąty problem z listy Hilberta dotyczył istnienia uniwersalnej algorytmicznej procedury pozwalającej rozstrzygnąć, czy równanie diofantyczne posiada rozwiązania, czy nie. Równania te są z jedną lub z wieloma niewiadomymi, o współczynnikach całkowitych, rozwiązań szuka się wśród liczb całkowitych. Można tych rozwiązań nie znaleźć, można znaleźć skończoną ilość, można znaleźć nieskończenie wiele.

DEFINICJA 4. $A(n)$ wielomianowe (wykładnicze) równanie diofantyczne $L(x_1,...,x_n) = R(x_1,...,x_n)$ jest zbudowane z nieujemnych całkowitych zmiennych $x_1,...,x_n$, i z niecałkowitych stałych, przy użyciu operacji dodawania $(A + B)$, mnożenia $(A \cdot B)$, potęgowania (A^B).

DEFINICJA 5. Predykat $P(a_1,...,a_n)$ jest rekurencyjnie przeliczalny wtedy, gdy istnieje algorytm, który dla danych nieujemnych liczb całkowitych $a_1,...,a_n$ będzie (poprzez wygenerowanie wszystkich n-tek spełniających P) odkrywał, że liczby te posiadają własność P. Predykat P jest rekurencyjny wtedy, gdy w dodatku istnieje algorytm, który będzie odkrywał, że liczby te nie posiadają własności P. Predykat P jest wielomianem (wykładniczym) diofantycznym wtedy, gdy P stwierdza, że istnieją nieujemne liczby całkowite $x_1,...,x_m$ takie, że $L(a_1,...,a_n,x_1,...,x_m) = R(a_1,...,a_n,x_1,...,x_m)$

TWIERDZENIE 6. Predykat jest wielomianowym/wykładniczym równaniem diofantycznym wtedy, gdy jest rekurencyjnie przeliczalny.[18]

Obliczanie może być zakodowane jako wielomianowe (wykładnicze) równanie diofantyczne, a precyzyjnie, jako konstrukcja wielomianów, która generuje/reprezentuje zbiór liczb pierwszych (wniosek ten można rozciągnąć na liczby całkowite dodatnie).[19] Mamy zatem arytmetyzację obliczania, a przy tym pewną *summę* teorii liczb, bo (metaforycznie) liczby pierwsze stanowią

---

[18] M. Davis, H. Putnam, J. Robinson, *The decision problem for exponential diophantine equations*, Annals of Mathematics 74, 1961, s. 425–436.

[19] J. P. Jones, *Diophantine representation of the set of prime numbers*, Amer. Math. Monthly 83, 1976, s. 449-464, również: G. J. Chaitin, *Algorithmic Information Theory*, Cambridge Univ. Press, Cambridge 1987.

„cegiełki" z których składają się inne liczby. W 1970 r. rosyjski matematyk Matijasiewicz kierując się wynikami uzyskanymi przez Robinson, Davisa i Putnama udowodnił, że nie istnieje program komputerowy systematycznie odpowiadający tak/nie na pytanie, czy dowolny układ równań diofantycznych posiada rozwiązania, czy nie.[20] Jeżeli wykorzystamy następującą definicję:

DEFINICJA 6. Zbiór $A \subset \mathrm{N}$ jest rekurencyjnie przeliczalny wtedy, gdy $A = \emptyset$, bądź A jest zakresem funkcji (częściowo) rekurencyjnej. Zbiór $A \subset \mathrm{N}$ jest rekurencyjny wtedy, gdy zarówno on jak i jego dopełnienie $(\mathrm{N} - A)$ jest rekurencyjnie przeliczalne.

oraz uwzględnimy rezultat Matijasiewicza, to musimy przyjąć, że zbiory diofantyczne są rekurencyjnie przeliczalne, ale nie są rekurencyjne. Zdanie „Nie istnieje rozwiązanie równania diofantycznego D" spełnia też własność zdania G w sensie I-go twierdzenia Gödla o niezupełności. Dodatkowo jest prawdziwym zdaniem teorio-liczbowym, a nie metamatematycznym jak zdanie G. Praktyczne sformułowanie tego zdania jest jednak niewykonalne z uwagi na ogromne współczynniki.[21] Przykładem wykładniczego równania diofantycznego jest słynne równanie Fermata

$$(x+1)^{n+3} + (y+1)^{n+3} = (z+1)^{n+3}$$

gdzie szukamy liczb całkowitych dodatnich w miejsce zmiennych i wykładników. Dodać trzeba, że negatywne rozwiązanie X problemu Hilberta jest równocześnie negatywnym rozwiązaniem ogólnego problemu rozstrzygalności Hilberta. Jeżeli bowiem nie istnieje uniwersalny program (mechaniczna procedura) odpowiadający tak/nie dla wszystkich równań diofantycznych, to nie istnieje również uniwersalny program odpowiadający tak/nie dla wszystkich zagadnień matematycznych.

Hilbert uważał, że „...tak długo jak dziedzina nauki oferuje obfitość problemów, tak długo jest żywa; niedostatek problemów znaczy obumarcie lub ustanie niezależnego rozwoju (przeł.. *A.W*)".[22] Spoglądając na zestaw problemów do rozwiązania, można odczytać to tak, że głównym problemem jest skonstruowanie procedury, która rozwiąże wszystkie problemy matematyczne. Mimo niepowodzenia program Hilberta miał kolosalny wpływ na matematykę. Jeden z kierunków oddziaływania wiąże się ze sferą motywacji – postulaty Hilberta były inspiracją dla wielu matematyków, głównie Gödla

[20] J. V. Matijasiewicz, *Diofantonost pierieczyslimych mnożestv,* Dokl. Akad. SSSR, 191, 2, 1970; dow w j. pol: Z. Adamowicz, P. Zbierski, *Logika matematyczna*, PWN, Warszawa 1991.

[21] Zob. S. Krajewski, *Twierdzenie Gödla a filozofia*, Studia Filozoficzne, 6/7, 1988.

[22] H. Weyl, *David Hilbert and his mathematical work*, Bulletin of the American Mathematical Society, 50, 1944, s. 612–654; cytowane za: K. Trzęsicki, *From the Idea of Decidability to the Number* Ω *,* w: *Issues of Decidability and Tractability*, University of Bialystok, 2006, s. 80.

i Turinga. Drugi koresponduje z ideą zmechanizowania dedukcji, która znajduje zastosowanie w naukach komputerowych. Maszyna Turinga to nic innego jak abstrakcyjny model matematyka, który pracuje zgodnie z formalistycznym programem Hilberta.

Pojawia się obecnie też taka myśl, że w przeciwieństwie do matematyki, której dziedzina nie jest skończenie aksjomatyzowalna, w obrębie fizyki daje się argumentować w ten sposób, że chociaż nie znamy wszystkich praw natury, to jest ich skończona liczba. Dlatego można wstrzymać eksperymentowanie – bo jesteśmy w posiadaniu ostatecznych i prawdziwych wzorów – i tylko wyprowadzać ich konsekwencje. Co jest już syntaktycznym zadaniem dedukcji. Formalistyczna utopia Hilberta, nie do pomyślenia w matematyce, byłaby zatem do zrealizowania w fizyce.

4.3. „Program" Laplace'a

Laplace pisał, że

> ..., gdyby dopuścić na chwilę myśl o inteligencji, która byłaby w stanie objąć rozumem wszystkie siły ożywiające przyrodę i odpowiadające im sytuacje istot składających się na nią – inteligencji wystarczająco pojemnej, by poddać te dane analizie – ujęłaby w jeden wzór zarówno ruchy największych ciał, jak i najlżejszego atomu; nic nie byłoby dla niej niepewne, a przyszłość jawiłaby się jej przed oczami wyraźnie jak przeszłość.[23]

Używając dzisiejszego języka, „sytuacje istot składających się na nią [przyrodę]" denotują warunki początkowe otrzymane w pomiarach. Wyraźne widzenie przyszłości polega znów na tym, że się umieszcza te warunki początkowe w rekurencyjnym wzorze i oblicza przyszłe położenia. Pomysł Laplace'a byłby do zrealizowania, gdyby nie to, że warunki początkowe można ustalić wyłącznie ze skończoną dokładnością, a skończone części rozwinięć liczb rzeczywistych są zawsze obliczalne. Nie da się więc wykluczyć, że „rzeczywiste" warunki początkowe są nieobliczalne, tylko skończone, czyli obliczalne części liczb otrzymanych w pomiarach nic o tym nie mówią. Tym samym poza polem analizy pozostaje nieprzeliczalna ilość potencjalnych danych. Z jednej zatem strony, nie sposób zidentyfikować, czyli ująć liczbowo nieobliczalności warunków początkowych, z drugiej, jeżeli nawet imitując nieobliczalność wykorzystamy maszynę liczącą posiadającą generator liczb losowych, to nie ma gwarancji, że symulacji startowej coś realnie odpowiada. Pozostaje startować od obliczalności. Ale i tu, jak się okazuje, plan Laplace'a pada, bo istnieją nieobliczalne funkcje analityczne. Dobrym przykładem sytuacji, która najwyraźniej dostarcza logiczną nieefektywność jest problem trzech i więcej ciał. Dotyczy on ruchu ciała o nieznacznej masie, które poru-

---

[23] P. S. de Laplace, *A Philosophical Essay on Probability*, Dover, New York 1951; francuski oryginał: P. S. de Laplace, *Essai philosophique sur les probabilites*, 1814.

sza się w polu grawitacyjnym dwóch ciał o dużej masie. W 1892 roku Poincaré odkrył, że ruch małego ciała może być bardzo dziwny, nieregularny. Obecnie panuje przeświadczenie, że jeżeli nawet można pomyśleć rozwiązanie problemu trzech ciał, to w terminach standardowych, lecz „bardziej wyszukanych funkcji".[24] Ilustracją problemu mogą być pozycje trzech idealnych planet w przypadku spełniającym następujące równanie różniczkowe:

$$\partial_{tt} z[t] == -z[t]/z[t]^2 + (1/2(1+eSin[2\pi t]))^2)^{3/2}$$

gdzie e jest ekscentrycznością orbity eliptycznej planet. Pomijając sytuację gdy $e = 0$, równanie nie ma rozwiązania w terminach standardowych funkcji (rekurencyjnych). Kreisel uważał, że problem kolizji związany z problemem trzech ciał można traktować jako potencjalne źródło nieobliczalności, a dokładnie, jako „...sytuację do analogowej komputacji funkcji nierekurencyjnych."[25]

Trzeba dodać, że współczesne nauki komputerowe oddzielają pojęcie efektywnej obliczalności/rozstrzygalności od pojęcia wykonalnej obliczalności (*feasible computability*). Wykonalną obliczalność można rozumieć jako „obliczalność w praktyce", albo „obliczalność w świecie realnym".[26]

4.4. Samoodniesienie

Samoodniesienie stawia matematyków w trudnym położeniu. Raz bowiem „szkodzi", innym razem „pomaga". W jednym przypadku odpowiada za sprzeczność w traktowanej jako fundament matematyki teorii mnogości (antynomia Russella), w drugim zaś jest wykorzystywane w dowodzeniu (dowód twierdzenia Gödla). Generalnie, mamy z nim do czynienia wtedy, gdy symbol (słowo, zdanie, wypowiedź, język) odnosi się do własnej semantyki, znaczenia, interpretacji. Sądzi się, że występowanie paradoksów związane jest z faktem, że istnieją obiekty, które nie mogą być zdefiniowane w pewnym skończonym języku formalnym.[27] Można to pokazać w prosty sposób: załóżmy, że istnieje ekwiwalentna skończonej maszynie Turinga tak zwana *UNIWERSALNA MASZYNA PRAWDY (UMP)*. Maszyna ta na wejściu otrzymuje dowolne zdania, których prawdziwość musi ocenić. Na wyjściu produkuje zaś proste komunikaty: *PRAWDA* lub *FAŁSZ*. Załóżmy teraz, że na wejściu *UMP* pojawia się zdanie „*UMP* nie rozstrzygnie, że to zdanie jest prawdziwe". W rezultacie mamy ten sam problem jak w przypadku antynomii kłamcy: *UMP* nie może odpowiedzieć *PRAWDA* lub *FAŁSZ* bez popada-

---

[24] S. Wolfram, *New Kind of Science*, Wolfram Media Inc., 2002, s. 972.

[25] G. Kreisel, *Church's Thesis: a kind of reducibility axiom for constructive mathematics,* w: *Intuitionism and Proof Theory*, North-Holland Publ. Comp., Amsterdam, London 1970, s. 143.

[26] R. Murawski, *The Present State of Mechanized Deduction, and Present Knowledge of its Limitations*, Studies in Logic, Grammar and Rhetoric, 9 (22), 2006, s. 48.

[27] A. Tarski, *Pojęcie prawdy w językach nauk dedukcyjnych,* Towarzystwo Naukowe Warszawskie, Warszawa 1933.

nia w sprzeczność. Ale „ktoś z zewnątrz", nie będący częścią maszyny, może rozpoznać, że zdanie „*UMP* nie rozstrzygnie, że to zdanie jest prawdziwe" jest prawdziwe. Oznacza to, że operuje „zewnętrznym" pojęciem prawdy. Mocniejszym niż to, którym dysponuje *UMP*. Nie zawsze też samozwrotność wywołuje sprzeczność. Zdanie „To zdanie jest prawdziwe" nie niesie paradoksalnych konsekwencji.

5. MASZYNA TURINGA

Już Leibniz chciał, by wszelkie spory rozstrzygane były nie w ramach werbalnego starcia między adwersarzami, ale przez precyzyjną manipulację na symbolach. Podobna do arytmetycznej mechaniczna kalkulacja znaleźć miała poprawne stanowiska w kontrowersyjnych kwestiach, a koniecznym warunkiem jej przeprowadzenia było istnienie prerekwizytu – ujednoliconego i sformalizowanego języka.[28] W dwudziestym wieku idea zmechanizowania rozumowania pojawiła się pod postacią maszynowego dowodzenia. Argumentowano, że maszyna oprócz operacji na liczbach może dokonywać operacji boolowskich. W rzeczywistości komputery raczej pomagają w dowodzeniu niż same konstruują dowody twierdzeń. Zdarza się, że przy pomocy komputera można znaleźć kontrprzykład.

5.1. Komputery uniwersalne

Komputery uniwersalne to klasa automatów, w których jest możliwa implementacja funkcji (częściowo) rekurencyjnych. Maszyna Turinga i automat komórkowy są elementami tej klasy. Trzeba odróżnić maszyny deterministyczne (maszyna Turinga, automat komórkowy) od maszyn nondeterministycznych (non-deterministyczna maszyna z wyrocznią *oracle*). Te drugie są zdolne do rozwiązywania problemów nierozwiązywalnych dla maszyn pierwszego rodzaju, choćby problemu stopu. Teraz przeprowadzimy rozumowanie, które dostarczy definicję uniwersalnego algorytmu.

Uniwersalny algorytm. Niech P_x będzie zbiorem instrukcji związanym z numerem gödlowskim $x = \#(P_x)$. Niech φ_x będzie funkcją związaną z P_x. Przeprowadźmy teraz następujące postępowanie: dla dowolnie wybranych liczb $x, y \in \mathrm{N}$ znajdźmy P_x, na przykład poprzez enumerację wszystkich zbiorów instrukcji aż do $(x+1)$-go miejsca. Dalej, wykorzystując P_x dla wejścia y obliczmy $\varphi_x(y)$. Jeżeli $\varphi_x(y)$ będzie miało wartość, to weźmy tą

[28] Tendencję do ujednolicenia języka i redukcji pojęć da się zauważyć w matematyce od Kartezjusza. Zredukowano geometrię do analizy. Zarytmetyzowano analizę używając pojęcia liczby, funkcji, zbioru. Liczbę rzeczywistą zdefiniowano przy pomocy pojęcia liczby naturalnej, ciągu, granicy. Samo pojęcie liczby naturalnej sprowadzono do pojęć teoriomnogościowych. Przykładem rekonstrukcji matematyki w ramach jednolitej teorii (mnogości) jest *Principia Mathematica* Russella i Whiteheada.

wartość do obliczenia $u(x, y)$. Jeżeli to zrealizujemy, to otrzymamy efektywnie obliczalny algorytm (związany z u), który produkuje $\varphi_x(y)$ przy wejściu x, y. Przy tej definicji u, $u(x, y)$ efektywnie imituje $\varphi_x(y)$.

Przez tezę Churcha istnieje $z \in \mathrm{N}$ takie, że algorytm u koresponduje z funkcją częściowo- rekurencyjną $\varphi_z(x, y)$. W tym momencie możemy podać definicję komputera uniwersalnego:

DEFINICJA 7 (komputer uniwersalny). System fizyczny albo inne urządzenie techniczne, w którym może być implementowana uniwersalna funkcja $u = \varphi_z$ nazywa się uniwersalnym komputerem bądź uniwersalną maszyną liczącą. Uniwersalny komputer może obliczać wszystkie funkcje obliczalne.

Wyrażenie $U(p, s) = t$ będzie od tej pory używane do oznaczania uniwersalnego komputera U z programem p, wejściem s i wyjściem t. Ø denotuje puste wejście lub wyjście. $s_1 = s_{11}, s_{12}, s_{13}, \ldots, s_{1i}$ oznacza ciąg wejściowy.

5.2. Maszyna Turinga[29]

Najbardziej znanym uniwersalnym komputerem jest maszyna Turinga zawierająca skończoną pamięć na potencjalnie nieskończonej taśmie. Taśma składa się z kwadratów. Informacja jest na tych kwadratach zapisywana lub z nich odczytywana. Projekt maszyny sporządzony został na długo przed powstaniem pierwszych maszyn elektronicznych, a słowo „komputer" stosowało się w oryginale bardziej do osoby wykonującej obliczenia niż do urządzenia technicznego. Wiele wskazuje na to, że Turing stworzył model/przepis działania umysłu ludzkiego.

DEFINICJA 8 (maszyna Turinga). Przyjmijmy dyskretne cykle czasowe oznaczone przez 0,1,2,3,.... Maszyna Turinga jest automatem, który posiada następujące cechy:

1. skończona liczba stanów wewnętrznych $a_1, \ldots, a_k$ tworzy zbiór $A = \{a_i\}$; a_0 jest zwane stanem pasywnym, stany $a_1, \ldots, a_k$ nazywa się stanami aktywnymi.

2. składająca się z kwadratów taśma przesuwać się może do przodu i do tyłu w taki sposób, że w każdym momencie skanowany może być tylko jeden kwadrat.

3. z każdego kwadratu można odczytać oraz w każdym kwadracie można zapisać skończoną liczbę symboli $s_0, s_1, \ldots, s_l$ gdzie s_0 jest niezapisanym symbolem; $l + 1$ oznacza możliwe położenia kwadratu.

W każdym momencie czasowym sytuacja maszyny jest zdefiniowana przez:

a. szczególny stan maszyny a_i

---

[29] A. Turing, *On computable numbers with an application to the Entscheidungsproblem*, Proc. Lond. Math. Soc. Ser. 2, 42, 1936.

b. szczególną pozycję taśmy w maszynie (między innymi przez to, który kwadrat jest skanowany)

c. szczególne zapisanie całej taśmy

$$[s][s][s][s][s][s][\ ]$$
$$\updownarrow \quad \text{głowica odczytująco-zapisująca}$$
$$\big|\ \text{stan wewnętrzny } a_j\ \big|$$

Stan aktywny maszyny, czyli między t a $t+1$, zawiera trzy możliwe ruchy:

(i) zapisywanie na taśmie symbolu s_1 jako stanu kwadratu skanowanego w momencie t; później połączone z (ii) przesunięciem taśmy w ten sposób, że w momencie $t+1$ maszyna jest albo w kwadracie na prawo, albo w kwadracie na lewo od pozycji skanowanej w momencie t (opisuje się to L, C, R); (iii) wybór nowego stanu wewnętrznego a_j dla następnego cyklu czasowego $t+1$.

Działanie maszyny może więc być opisane przez trójki:

$$s_{j'}\begin{bmatrix} L \\ C \\ R \end{bmatrix} a_{j'} \quad \text{lub} \quad i'\begin{bmatrix} L \\ C \\ R \end{bmatrix} j'$$

Pary $s_i a_j$ lub ij nazywa się konfiguracjami maszyny. Konfiguracja maszyny determinuje jej późniejsze zachowanie. Jeżeli konfiguracja mówi „przesuń w lewo", a skanowany kwadrat jest już lewym kwadratem, to maszyna idzie w stan pasywny a_0. Istnieje $(l+1)k$ aktywnych konfiguracji. Szczególna maszyna Turinga jest definiowana przez tablicę pokazującą dla każdej aktywnej konfiguracji, które działanie ma być wykonane.[30]

5.3. Problem stopu

Problem stopu wiąże się z pytaniami: (i) Jak system mechaniczny będzie ewoluował? (ii) Co będzie na wyjściu dostarczał program? (iii) Które twierdzenia są wyprowadzalne z systemu formalnego? Generalnie, problem stopu związany jest ze sprawą przewidywania przyszłości w systemach mechanicznych.

DEFINICJA 9 (problem stopu). Rozważmy algorytm A oraz wejście x. Problem stopu wiąże się z pytaniem, czy $A(x)$ dostarczy specyficzne wyjście, czy nie. Równoważnie, można pytać, czy A zakończy się na x, czy nie.

---

[30] Zob. S. C. Kleene, *Turing's Analysis of Computability, and Major Application of It, in The Universal Turing Machine*, A Half-Century Survey, red. Herken R., Kammerer & Unverzagt, Hamburg 1988.

Przypadek „A zakończy się na x" denotujemy $A(x)\downarrow$. Przypadek „A nie zakończy się na x" denotujemy $A(x)\uparrow$.

TWIERDZENIE 7 (o rekurencyjnej nierozwiązywalności problemu stopu). Nie istnieje efektywnie obliczalny algorytm/funkcja częściowo rekurencyjna, który rozstrzyga problem stopu. Problem stopu jest nierozwiązywalny.

Dowód (przez zaprzeczenie) powyższego twierdzenia wykorzystuje metodę diagonalizacji Cantora: [31]

Rozważmy dowolny algorytm $A(x)$ z wejściem x (x jest ciągiem symboli). Załóżmy (nie wprost), że istnieje „algorytm stopu" (STOP), który jest w stanie rozstrzygnąć, czy A zakończy się na x, czy nie ($STOP(A(x))\downarrow$; zakończenie obliczeń jest własnością fikcyjnego algorytmu). Wykorzystując $STOP(A(x))$ łatwo teraz skonstruować inny algorytm, denotujmy go B, który zachowuje się następująco: B odczytuje program A jako wejście i czyni jego kopię. Odtąd program A można przedstawić jako B (w pewnej zakodowanej formie, to znaczy jako ciąg symboli). Kod $\#(A)$ jest użyty jako ciąg wejściowy dla A, to jest B formuje $A(\#(A))$ [na przyszłość oznaczone przez $A(A)$] i przekazuje do podprogramu STOP. Teraz B może się zachować dwojako:

przypadek a) jeżeli $STOP(A(A))$ rozstrzygnie, że $A(A)$ staje, to B nie staje.

przypadek b) jeżeli $STOP(A(A))$ rozstrzygnie, że $A(A)$ nie staje, to B staje.

Następnie trzeba pokazać, że jest coś nie w porządku z B, a co za tym idzie, że jest coś nie w porządku ze $STOP$ (wyprowadzenie B ze $STOP$ jest oczywiste i obliczalne). Podstawiamy zatem B za A, tj. na wejściu B pojawia się on sam, oraz przeprowadzamy rozumowanie jak wyżej:

przypadek a') zakładając, że $B(B)$ staje, to $STOP(B(B)$ działając na $B(B)$ nie staje.

przypadek b') zakładając, że $B(B)$ nie staje, to $STOP(B(B))$ steruje $B(B)$ w stronę W obu przypadkach osiągamy sprzeczność. Sprzeczności można uniknąć tylko przez założenie o nieistnieniu B, a odkąd B jest konstrukcją $STOP$, przez założenie o niemożliwości istnienia algorytmu $STOP$.

Dodać należy, że syntaktyczna struktura dowodu jest ekwiwalentna strukturze dowodu twierdzenia Gödla o niezupełności. Siła twierdzenia o rekurencyjnej nierozwiązywalności problemu stopu związana jest z tym, że inne problemy nierozstrzygalne można zredukować do problemu stopu. Wystarczy wtedy pokazać, że gdyby wskazany problem był rozstrzygalny, to musiałby istnieć algorytm stopu. Przykład może być następujący: nie istnieje algorytm

[31] Interpretacja dowodu za: K. Svozil, *Randomness & Undecidability in Physics*, Word Scientific, Singapore, New Jersey, London, Hong Kong 1993, s. 115.

oddzielający przypadki, gdy rozwiązania równań ruchu są okresowe od przypadków, gdy są nieokresowe, bo gdyby taki algorytm istniał, to musiałby istnieć algorytm stopu.

Komputery uniwersalne o ograniczonych źródłach są mechanicznymi systemami fizycznymi, których ewolucja jest nieprzewidywalna/nieobliczalna/nierozstrzygalna. Podstawą nieprzewidywalności jest (rekurencyjna) nierozwiązywalność problemu stopu. Często uważa się, że Uniwersum jest rodzajem skończonego automatu. Teza ta zawiera dwa żądania:

— prawa natury muszą być mechanistyczne, czyli obliczalne w sensie tezy Churcha-Turinga;

— zdolności obliczeniowe systemów fizycznych muszą być skończone.

6. ALGORYTMICZNA INFORMACJA

Idea teorii algorytmicznej informacji opiera się na spostrzeżeniu, że obiekt matematyczny można definiować przez długość najkrótszego programu, który na wyjściu dostarcza kod tego obiektu. Jeżeli więc przyjmiemy, że ewolucję systemów fizycznych daje się reprezentować obliczeniowo, to wymieniony sposób deskrypcji stosuje się też do niej. Wprowadzenie kategorii *shorter length* jest zasadne, ponieważ w następnej kolejności pojawi się pojęcie systemu chaotycznego, którego oryginalna deskrypcja nie może być „ściśnięta" do postaci krótszej. [32]

6.1. Kodowanie

Kodowanie jest niezbędne, ponieważ dane ze źródła muszą mieć algorytmicznie rozpoznawalną postać. Trzeba więc dokonać odwzorowania z alfabetu źródłowego w alfabet kodu. Sama technika kodowania musi zapewnić unikalność kodu. W tym celu wprowadza się tzw. prefix-kody (*self-delimiting codes, prefix-free, instantaneous codes*). Ich wykorzystanie znosi niebezpieczeństwo, że ta sama informacja, przy tej samej strategii kodowania będzie miała różne znaczenie. Prefix-kody spełniają też funkcję natychmiastowego dekodowania: symbole ze źródła mogą być dekodowane w trakcie transmisji a nie dopiero po jej zakończeniu.[33] Co się tyczy systemów fizycznych, to kodowanie zapewnia odpowiednią reprezentację dla danych eksperymentalnych. W efekcie strumień danych ze źródła (eksperyment) przy pomocy algorytmu (prawa natury) zmienia się w strumień danych, który koresponduje z zachowaniem układu fizycznego.

---

[32] Teoria algorytmicznej informacji wiąże się z nazwiskami Chaitin, Solomonoff, Kolgomorov; por. np. G. J. Chaitin, *Algorithmic Information Theory*, Cambridge University Press, Cambridge 1987.

[33] R. J. McEliece, *The Theory of Information and Coding,* w: *Encyclopedia of Mathematics and its Applications*, v. 3. London 1977.

6.2. Algorytmiczna informacja

$O(1)$ (czyta się „porządek 1”) denotuje funkcję, której wartość bezwzględna jest ograniczona przez nieokreśloną stałą dodatnią; $\varphi(x) = O(1)$ oznacza $|\varphi| < A$, gdzie A jest stałą niezależną argumentów x funkcji φ; $|s|$ denotuje długość obiektu s zakodowanego w notacji binarnej; U oznacza komputer uniwersalny; H' jest miarą „długości najkrótszego programu”, który na wyjściu dostarcza kod obiektu (apostrof ‘ oznacza nieokreślony program/kod); H' jest mierzona w bitach.

Jak wspomnieliśmy, obiekt matematyczny można charakteryzować przez długość najkrótszego programu, który na wyjściu dostarcza kod tego obiektu. Rozważmy teraz ciąg binarny $x(n)$ o długości n. Przy pierwszym spojrzeniu wydaje się, że zawartość informacyjna H' ciągu $x(n)$ nie może przekroczyć długości tego ciągu. Czyli, że $H'(x(n)) \le n + O(1)$; plus $O(1)$ z dodatkowego programu, na przykład z programu *„DRUKUJ $x(n)$”*. Technika kodowania umożliwia jednak wykorzystanie specyficznych symboli, takich jak niezapisany symbol _, które nazywa się *end-markerami* (do kończenia enumeracji). Dzięki *end-markerom* program może skanować wszystkie cyfry $x(n)$, określić jego długość w realnym czasie wykonania, drukować $x(n)$ oraz n, w końcu stawać. Wiadomość, że $x(n)$ ma długość n bitów stanowi dodatkową informację, która ma wartość $H'(n)$ bitów. W efekcie tego zabiegu programistycznego w $H'(n)$ można „ścisnąć” informację o $x(n)$, następnie dodać do $x(n)$, co powoduje, że jest jej więcej niż w n-bitach $x(n)$. Całościowa informacja równa się wtedy $n + H'(n)$. Rozumowanie można iterować. Teraz można podać definicję algorytmicznej informacji.

DEFINICJA 10 (algorytmiczna informacja albo algorytmiczna złożoność). Zakładamy kodowanie z prefix-free. Kanoniczny program związany z obiektem s, reprezentowanym jako ciąg, jest denotowany przez s^* i definiowany przez

$$s^* = \min_{U(p)=s} p,$$

gdzie s^* jest pierwszym elementem w uporządkowanym zbiorze wszystkich ciągów takim, że program dla U oblicza s. Ciąg s^* jest w ten sposób kodem najkrótszego programu, który implementowany w U na wyjściu dostarcza s. Jeżeli istnieją osobne programy binarne o równej długości, to wybierany jest ten, który pierwszy przeprowadza enumerację wykorzystując leksykograficzną porządkującą relację $0 <1$. Niech $|x|$ obiektu zakodowanego jako ciąg binarny występuje jako długość tego ciągu. Algorytmiczna (statyczna) złożoność $H(s)$ obiektu s reprezentowanego jako ciąg jest zdefiniowana przez długość najkrótszego programu p, który implementowany w U generuje wyjście s

$$H(s) = |s^*| = \min_{U(p)=s} |p|$$

Jeżeli żaden program w komputerze U nie dostarcza wyjścia s, to $H(s) = \infty$. Wspólna algorytmiczna informacja $H(s,t)$ reprezentowanych jako ciągi bitów jest długością najkrótszego programu binarnego do obliczania konkatenacji s i t jednocześnie. Relatywna albo warunkowa algorytmiczna informacja $H(s/t)$ obiektu s przy danym t jest długością najkrótszego programu do obliczania s z najkrótszego programu do obliczania t.

$$H(s/t) = \min_{U(p,t)=s} |p|$$

Z pojęciem relatywnej algorytmicznej informacji blisko związane jest pojęcie relatywnej obliczeniowej złożoności zbioru A. Mierzy się ją porównując zbiór A z innymi zbiorami przez wykorzystanie procedury zwanej redukowalnością $\leq_r$. Najogólniejszym rodzajem redukowalności jest redukowalność Turinga $\leq_T$, która formalizuje intuicję, że nasze rozumienie zdania „X jest przynajmniej tak złożony jak Y", sprowadza się do rozumienia zdania „X może być obliczony przy pomocy Y" (albo „X może być obliczony relatywnie do Y"). Redukowalność $\leq_r$ na zbiorach oznaczamy $X \equiv_r Y$. Klasy ekwiwalentne nazywa się r-stopniami. r-stopień X zawiera zbiory mające tą samą złożoność co X, z uwzględnieniem $\leq_r$.[34]

6.3. Prawdopodobieństwo stopu Ω (Chaitin)

Załóżmy, że program p (z *prefix-free*) dla komputera U reprezentowany przez ciąg bitów o długości $|p|$ dostarcza specyficzny obiekt s. Niech będzie to przykładowo 00101110100....10100010100. Można teraz zadać pytanie, jakie jest prawdopodobieństwo, że program p dostarczający obiekt s otrzymamy w wyniku procesu losowego, takiego jak rzut idealna monetą (orzeł/reszka $\leftrightarrow 0/1$ lub reszka/orzeł $1/0$; prawdopodobieństwo pojedynczego rzutu $= \frac{1}{2}$). Definicja algorytmicznego prawdopodobieństwa Ω ujmuje tą sprawę ściśle.

DEFINICJA 11 (algorytmiczne prawdopodobieństwo, prawdopodobieństwo stopu Ω). Jeżeli s jest obiektem zakodowanym jako ciąg binarny i $S = \{s_i\}$ jest zbiorem takich obiektów s_i, to algorytmiczne prawdopodobieństwo P definiuje się przez

1) $$P(s) = \sum_{U(p)=s} 2^{-|p|}$$

---

[34] Zob. A. Nies, *Computability and Randomness*, Oxford Logic Guides 51, Oxford University Press 2009, s. 8–16 oraz 238–258.

2) $P(s) = \sum_{s \in S} P(s_i) = \sum_{U(p) \in S} 2^{-|p|}$

3) $\Omega = \sum_{s} P(s) = \sum_{s} \sum_{U(p)=s} 2^{-|p|} = \sum_{U(p)\downarrow} 2^{-|p|}$

Ω jest prawdopodobieństwem stopu, to znaczy miarą prawdopodobieństwa, że dowolny program ewentualnie stanie. Dokładnie rzecz biorąc, Ω nie jest żadnym prawdopodobieństwem, tylko miarą częstotliwości z jaką program p generuje s. Chociaż Ω jest przypadkowe, w sensie Martin-Löf przypadkowości, to może być otrzymane z efektywnie obliczalnego algorytmu, w granicy nieskończonego czasu obliczeń. Zbiór wszystkich twierdzeń postaci $P(s) > 2^{-n}$ jest też rekurencyjnie przeliczalny, ponieważ można „empirycznie" znaleźć $H(s)$ przez przebiegnięcie wszystkich programów o rozmiarze mniejszym lub równym n i sprawdzenie, czy dostarczają s (trwać to może bardzo długo, ale nie wiecznie). Program do obliczania prawdopodobieństwa stopu Ω można zakodować jako wykładnicze równanie diofantyczne.[35]

6.4. Algorytmiczna przypadkowość

Rozważmy ciągi binarne zawierające 0 i 1. Symbol ω występuje jako liczba porządkowa nieskończoności. 2^{ω} denotuje zbiór wszystkich nieskończonych ciągów (binarnych). Nieskończone ciągi $x = x_1 x_2 x_3 \ldots$ w 2^{ω} mogą być reprezentowane przez binarne liczy rzeczywiste z przedziału $[0,1]$, jeżeli identyfikujemy x z $r = 0, x_1 x_2 x_3 \ldots$. Należy odnotować, że przypadkowość definiuje się dla ciągów o nieskończonej długości, czyli dla ciągów fizycznie nie do wygenerowania. Definicja jest zatem nieoperacyjna (łatwiej ją sformułować niż podać przykład).

DEFINICJA 12 (przypadkowość Chaitina)[36] Ciąg $x \in 2^{\omega}$ jest przypadkowy wtedy, gdy statyczna złożoność segmentu początkowego $x(n) = x_1, .., x_n$ długości n, dwu-bazowego rozwinięcia x, jest i pozostaje dowolnie większa niż n:

$$\lim_{n \to \infty} [H(x(n)) - n] = \infty$$

lub $\forall k \exists N_k \forall (n \geq N_k)[H(x(n)) \geq n = k]$.

Nieformalnie ujmując, informacyjna zawartość jednostki długości ciągu przypadkowego nie może być „ściśnięta" do dowolnej reprezentacji, która ma krótszą długość niż oryginalny ciąg. Liczba rzeczywista jest przypadkowa dokładnie wtedy, gdy jej segment początkowy nie podlega kompresji. Gene-

---

[35] G. J. Chaitin, *Algorithmic Information Theory,* Cambridge University Press, Cambridge 1987.

[36] Albo Martin-Löf/Solovay/Chaitin-przypadkowość; o ekwiwalencji definicji w: G. J. Chaitin, *Algorithmic Information Theory,* Cambridge University Press, Cambridge 1987.

ralnie, jeżeli algorytmiczna deskrypcja pewnych danych nie jest krótsza niż te dane, to dane te są przypadkowe.

6.5. Obliczeniowa złożoność

Pojęcie obliczeniowej (dynamicznej) złożoności związane jest z czasem obliczeń, czyli liczbą dyskretnych kroków w procesie kodowania obiektu. Obiektem mogą być na przykład rozwiązania pewnego problemu matematycznego, choćby rozwiązania równań ruchu, które wykorzystujemy do konstruowania zdań o przyszłości. Niech N będzie liczb a określającą rozmiar problemu, x binarną reprezentacją tego problemu/obiektu, $|x(N)|$ długością x.

DEFINICJA 13. Przyjmujemy problem z uporządkowanej listy N i jego rozwiązanie $x(N)$, jeżeli istnieje. Dynamiczna albo obliczeniowa złożoność $H_D(x(N))$ jest czasem (liczbą cykli), w którym najszybszy program p dla U oblicza $x(N)$.

$$H_D(x(N)) = \min_{U(p)=x(N)} \text{czas } p$$

Jeżeli żaden program dla U nie daje wyjścia $x(N)$, to $H_D(x(N)) = \infty$.

Ilustracją może tu być mechaniczna procedura sprawdzania uporządkowanej listy N-wpisów książki telefonicznej w celu odnalezienia wpisu/rozwiązania $x(N)$, w minimalnym czasie $H_D = O(\log N)$. Łatwo zauważyć, że $H_D(x(N)) \leq O(N!)$, chociaż stwierdzenie, że problem jest nieobliczalny w praktyce (*intractable*), czyli że nie może być rozwiązany w czasie wielomianowym, tj., że minimalny czas równy jest $H_D(x(N)) = O(N^k)$, przy $k < \infty$, pozostaje zawsze hipotezą. Nie istnieje bowiem uniwersalny algorytm, który w każdym przypadku potrafi oszacować czas obliczeń. Wynika to z twierdzenia o (rekurencyjnej) nierozwiązywalności problemu stopu. Alternatywnym do pojęcia obliczeniowej złożoności jest pojęcie logicznej głębi (*logical depth*). [37]

6.6. Klasa problemów P (rozwiązywalnych w czasie wielomianowym)

Jest to klasa problemów/języków rozstrzygalnych w „rozsądnym" czasie, przy „rozsądnym" zapasie danych. Z uwagi na to, że wielomiany rosną istotnie wolniej od dowolnej funkcji wykładniczej ($2^n, n!$) za rozsądny postanowiono brać czas wielomianowy, wielomianowe tempo wzrostu. Podział na „czas wielomianowy" i „czas wykładniczy" jest jednak matematycznym uproszczeniem, które w pewnych przypadkach może być mylące. Istnieją

[37] C. H. Bennet, *Dissipation, Information, Computational Complexity and the Definition of Organization*, w: *Emerging Synthesis in Science*, Academic Press, New York 1985.

efektywne w praktyce metody obliczania, które nie są wielomianowe, oraz wielomianowe metody obliczania, które nie są efektywne.

DEFINICJA 14. Klasa wielomianowych algorytmów P zawiera algorytmy, które mogą być rozwiązane w czasie wielomianowym. Problemy obliczeniowe, które mogą być rozwiązane przez wielomianowe algorytmy są nazywane problemami „obliczalnymi w praktyce" (*tractable*), albo „wykonalnymi" (*feasible*).

Problemem rozwiązywalnym w czasie wielomianowym jest wspomniane już wcześniej poszukiwanie wpisu w książce telefonicznej.

6.7. Klasa problemów NP (nierozwiązywalnych w czasie wielomianowym)

Istnieje interesująca klasa problemów/języków NP, których rozwiązanie (rozstrzygnięcie) wymaga wprowadzenia maszyn niedeterministycznych (np. niedeterministycznej *oracle*). Maszyny te są nieekwiwalentne standardowej maszynie Turinga. Klasa maszyn niedeterministycznych koresponduje z czasem wykładniczym 2^{n^k}.

DEFINICJA 15. (algorytmy NP). Klasa niedeterministycznych wielomianowych algorytmów zawiera algorytmy, które mogą być rozwiązane przez niedeterministyczną maszynę z wyrocznią (*oracle*) i zweryfikowane w czasie wielomianowym.

W odróżnieniu od deterministycznej niedeterministyczna maszyna nie ma jednoznacznie wyznaczonej kolejnej operacji do wykonania. Ma za to możliwość wyboru między kilkoma akcjami. Ujmując metaforycznie, maszyny niedeterministyczne uzyskują swoją moc dzięki niewielkim wymaganiom nałożonym na związek wejścia z wyjściem. Słowo wejściowe jest tu akceptowane już wtedy, gdy istnieje przynajmniej jeden ciąg wyborów niedeterministycznych, który doprowadza do stanu `tak`. Pozostałe wybory mogą dawać w rezultacie odrzucenie. Odrzucenie słowa wejściowego jest też trudniejsze, ponieważ wszystkie możliwe ciągi wyborów musza kończyć się odrzuceniem. Przykładem problemu rozwiązywalnego przez niedeterministyczną maszynę jest tak zwany problem komiwojażera (*Travelling Selesman Problem*). [38]

6.8. $P = ? NP$

Problemy z klasy NP mają zwięzłe dowody i problemy z klasy dopełnienie NP mają zwięzłe dowody. W pierwszym przypadku są to dowody pozytywne. W drugim przypadku są to dowody negatywne, czyli dyskwalifikacji.

---

[38] Zob. C. H. Papadimitriou, *Złożoność obliczeniowa*, Wydawnictwa Naukowo-Techniczne, Warszawa 2002.

Żaden problem nie posiada dowodu pozytywnego oraz negatywnego równocześnie. Jest oczywiste, że dowolny problem z klasy P znajduje się również w klasie $NP \cap$ dopełnienie NP. Istnieją jednak problemy z klasy NP dopełnienie NP, o których nie wiadomo, czy są w P.

Zgodnie z teorią algorytmicznej informacji kryterium wyboru między alternatywnymi teoriami jest minimalna długość ich reprezentacji. Widać tu związek z tak zwaną brzytwą Ockhama, gdzie kryterium była prostota. W narracji teorii algorytmicznej informacji prawa przyrody zarządzające ewolucją systemów fizycznych można reprezentować obliczeniowo (przez tezę Churcha-Turinga rekurencyjnie). Algorytmiczna informacja jest miarą długości takiej deskrypcji. Zatrudniając aparaturę pojęciową teorii algorytmicznej informacji, możemy stwierdzić, że prawa przyrody to „krótkie" kody dla „długich" danych eksperymentalnych.

7. CHAOS I NIEOBLICZALNOŚĆ

Naszym celem jest wykazanie, że klasa równań ruchu jest nierozstrzygalna, co równa się temu, że nie istnieje program wycinający okresowość w rozwiązaniach równań ruchu. Aby to zrealizować, potrzebna jest nam nieobliczalność, którą otrzymamy teraz z algorytmicznie zinterpretowanej teorii chaosu. [39] Trzeba tu przypomnieć, że problem istnienia algorytmu wycinającego okresowość sprowadzony został do problemu istnienia algorytmu stopu. Argumentacja przebiega w ten sposób, że gdyby istniał algorytm wycinający okresowość, to musiałby istnieć algorytm stopu.

7.1. Kodowanie teorii fizycznych

Istotną cechą teorii fizycznych jest istnienie obiektywnych zasad inferencji, które można reprezentować obliczeniowo, a przez tezę Churcha połączyć z funkcją rekurencyjną, która zastosowana do aksjomatów produkuje twierdzenia. W tym ujęciu terminy „komputer" i „zasady inferencji", „program" i „aksjomaty", „wyjście" i „twierdzenia" są synonimiczne. Schemat można rozbudować przez wkomponowanie pojęć takich jak „mechaniczny system fizyczny", „obserwacja", „prawo", „predykcje". Założeniem wyjściowym jest hipoteza, że *Uniwersum* jest rodzajem skończonego automatu, która zawiera dwa postulaty:

1. Prawa przyrody muszą być mechanistyczne, czyli reprezentowalne przez algorytm.
2. Zdolności obliczeniowe systemów fizycznych muszą być skończone.

---

[39] W innym miejscu interesowała nas nieobliczalność *analityczna* obecna w pracach Banacha/Mazura i Pour-El/Richardsa; patrz *Obliczalność a świat realny* (tekst zostanie opublikowany w: II tomie czasopisma FILOZOFIA A NAUKA)

Przez to otrzymujemy, że modelowi obliczeń opartemu na dyskretnych cyklach egzekucji odpowiada dyskretny model zmiany. Podstawową ideą jest jednak to, że teorie fizyczne można traktować jako źródło symboli, które mogą być „czytane" przez kogoś lub coś (abstrahujemy teraz od semantycznej zawartości teorii). W drugim przypadku przez komputer wyposażony w efektywną procedurę, która rozwiązuje dany problem (na przykład podaje rozwiązania równań ruchu). Jest oczywiste, że jednym z celów takiego przedsięwzięcia jest przewidywanie zachowania systemu fizycznego. Powinno się je wiązać ze strumieniem danych na wyjściu algorytmu. Ze strumieniem danych na wejściu algorytmu należy znów wiązać dane eksperymentalne. Sam algorytm koresponduje z prawem natury. Dlatego na miejscu wydaje się być stwierdzenie, że prawa przyrody to „krótkie kody dla długich danych eksperymentalnych". Co do techniki gwarantującej unikalność kodu, to kodowanie przeprowadza się z wykorzystaniem oryginalnej konstrukcji Gödla. Istnieje zatem unikalne odwzorowanie ze źródła symboli w zbiór ciągów symboli z alfabetu kodu. Rezultat kodowania systemu fizycznego, który stanowi kolekcję zdań o eksperymentach, jest zawsze prawdziwy lub fałszywy.

DEFINICJA 16. Niech $\bigcup_{j=1}^{M} p_{ij} = pj$ będzie zdarzeniem rozłożonym na zdarzenia elementarne p_{ij}. Dla zdarzeń elementarnych alfabet źródłowy zawiera tylko dwa symbole s_1 i s_2 korespondujące z PRAWDĄ (1) i FAŁSZEM (0). Kod $\#(p_{ij})$ dla p_{ij} jest w ten sposób zdefiniowany przez

$$\#(p_{ij}) = \frac{s_1}{s_2} \text{ gdy } \frac{p_{ij} = 1}{p_{ij} = 0}$$

Identyfikowanie prawa natury z funkcją rekurencyjną nie powoduje jednak, że można przeliczyć wszystkie prawa. Na przeszkodzie stoi twierdzenie o (rekurencyjnej) nierozwiązywalności problemu stopu dla maszyny Turinga, a równoważnie twierdzenie Gödla. W konsekwencji istnieją zdarzenia, które są niedowodliwe, czyli że nie są efektem jakiegokolwiek prawa.

7.2. Przypadkowość

Zbiory obliczane przez maszynę Turinga są ekwiwalentne zbiorom rozstrzygalnym przez algorytm. Matematyczne pojęcie Martin-Löf przypadkowości odpowiada intuicyjnemu pojęciu przypadkowości zbioru Z, które ma dwa aspekty:[40]

1. Z nie posiada (nie spełnia) wyjątkowych własności.
2. Z jest trudny do deskrypcji.

[40] P. Martin-Löf, *On the notion of randomness*, w: *Intuitionism and Proof Theory*, North-Holland Publishing Comp., Amsterdam, London 1970, s. 73–78.

ad.1. Niech zbiór Z powstaje w wyniku idealnego procesu losowego, który przebiega w czasie i dostarcza nieskończenie wiele bitów (0,1). Bity są niezależne. Zero i jedynka mają to samo prawdopodobieństwo ½, jak przy podrzucaniu idealna monetą. Prawdopodobieństwo, że ciąg x jest segmentem początkowym zbioru Z wynosi $2^{-|x|}$. Własności wyjątkowe reprezentowane są przez *null-klasy*, w odniesieniu do jednostajnej miary λ w przestrzeni Cantora. Potrzebne jest tu wyjaśnienie, że zbiory liczb naturalnych można widzieć jako atomowe obiekty i identyfikować z nieskończonymi ciągami nad {0,1}. Ciągi te są elementami przestrzeni Cantora $\{0,1\}^N$, zwykle denotowanej przez 2^N. Podzbiory 2^N nazywa się klasami, dla odróżnienia od zbiorów liczb.

DEFINICJA 17. Klasę $A \subseteq 2^N$ nazywa się null-klasą wtedy, gdy $\lambda A = 0$. Jeżeli $2^N - A$ jest null, to mówimy, że A jest conull.

Przykładem są własności wyjątkowe P i Q. Pierwsza stanowi, że wszystkie bity na parzystych pozycjach są zerami:

$$P(Y) \leftrightarrow \forall i Y(2i) = 0$$

Druga stanowi, że jest przynajmniej dwa razy więcej zer niż jedynek w granicy:

$$Q(Y) \leftrightarrow \liminf \#\{i < n : Y(i) = 0\} / n \geq 2/3$$

Odpowiadającymi klasami są null-klasy. Odtąd, w zgodzie z intuicją, nie powinny zawierać zbioru przypadkowego. Rodzajem null-klasy jest $\prod_2^0$ klasa.

DEFINCJA 18. Niech $A \subseteq N^k \times 2^N$ oraz $n \geq 1$.

(i) A jest $\sum_n^0$ wtedy, gdy

$$\langle e_1,...,e_k, X\rangle \in A \leftrightarrow \exists y_1 \forall y_2,..., Q y_n R(e_1,...,e_k, y_1,..., y_{n-1}, X \uparrow_{y_n})$$

gdzie R jest relacją obliczalną, oraz Q jest "$\exists$" wtedy, gdy n jest nieparzyste, oraz Q jest "$\forall$" wtedy, gdy n jest parzyste.

(ii) A jest $\prod_n^0$ wtedy, gdy dopełnieniem A jest $\sum_n^0$, tzn.

$$\langle e_1,...,e_k, X\rangle \in A \leftrightarrow \forall y_1 \exists y_2,..., Q y_n S(e_1,...,e_k, y_1,..., y_{n-1}, X \uparrow_{y_n})$$

gdzie S jest relacją obliczalną i Q jest "$\forall$" wtedy, gdy n jest nieparzyste, oraz Q jest "$\exists$" wtedy, gdy n jest parzyste.

Relacja jest arytmetyczna wtedy, gdy jest $\sum_n^0$ dla pewnego n.

($X \uparrow_{y_n}$ denotuje segment początkowy X)

Na przykład, $\prod_2^0$ klasa ma formę $\{X : \forall y_1 \exists y_2 S(y_1, X \upharpoonright y_2)\}$, a $\sum_3^0$ klasa ma formę $\{X : \exists y_1 \forall y_2 \exists y_3 R(y_1, y_2, X \upharpoonright_{y_3})\}$, gdzie S i R są relacjami obliczalnymi.[41]

ad. 2. Obiekt przypadkowy nie posiada wzoru. Jest niezorganizowany. Naszemu intuicyjnemu pojęciu przypadkowości odpowiada intuicyjne pojęcie „trudny do deskrypcji". Należy zanotować, że istnieją systemy deskrypcji zwane optymalnymi maszynami, które są w stanie rywalizować z każdą inną maszyną, więc opisać każdy możliwy ciąg symboli.[42] Zgodnie z tym ujęciem, pojęcie „trudny do deskrypcji" można sformalizować jako „niepodlegający kompresji", w odniesieniu do optymalnej maszyny. Nieformalnie rzecz ujmując, ciągi nie podlegające kompresji danych posiadają tę samą własność, której wymagamy od ciągów przypadkowych. Dla zbiorów pojęcie „trudny do deskrypcji" jest jednak trudniejsze, ponieważ każdy system deskrypcji opisuje tylko obliczalnie wiele zbiorów. Trzeba więc wprowadzić pojęcie deskrypcji domkniętej. Typ deskrypcji domkniętej reprezentują właśnie *null-klasy* $\prod_1^0$ i $\prod_2^0$. Konkluzja jest następująca: zbiór jest trudny do deskrypcji wtedy, gdy nie dopuszcza deskrypcji domkniętej, powiedzmy w sensie *null* $\prod_2^0$ *klasy*. Jeżeli zbiór dopuszcza deskrypcję np. w sensie *null* $\sum_1^0$ *klasy*, to nie jest trudny do deskrypcji. Warunki 1 i 2 wzięte razem charakteryzują pojęcie testu na przypadkowość, a w efekcie matematyczne pojęcie przypadkowości: Z jest przypadkowy jeżeli przechodzi wszystkie testy danego typu.

Pojęcie testu (formalne). Aby podać definicję Martin-Löfa-testu na przypadkowość potrzebne są dwa preliminaryjne fakty: [43]

FAKT 1 (zbiory otwarte). $R \subseteq 2^N$ jest rekurencyjnie przeliczalnym zbiorem otwartym wtedy, gdy $R = [W_e]$ dla pewnego e.

(W_e = dziedzinie (Φ_e); Φ_e denotuje funkcję częściowo-rekurencyjną o indeksie e).[44]

FAKT 2. $A \subseteq 2^N$ jest null wtedy, gdy istnieje ciąg $(G_m)_{m \in N}$ zbiorów otwartych taki, że $\lim_m \lambda G_m = 0$ oraz $A \subseteq \bigcap_m G_m$.[45]

---

[41] Zob. A. Nies, *Computability and Randomness*, Oxford Logic Guides 51, Oxford University Press 2009.

[42] Optymalna maszyna dostarcza ciąg a_n wtedy, gdy na wejściu dostaje jego deskrypcję.

[43] Właściwie test Martin-Löfa powinien nazywać się „testem na nieprzypadkowość". Test bowiem dobrze wycina jedynie obiekty nieprzypadkowe, uporządkowane.

[44] Dowód w: A. Nies, *Computability and Randomness*, op. cit., s. 53.

[45] Dowód w: A. Nies, op. cit., s. 70.

(klasa $B = \bigcap_m G_m$ jest borelowska oraz $\lambda B = 0$).[46]

Definicja Martin-Löfa-testu efektywizuje określenie *null klasy* z FAKTU 2.

DEFINICJA 19. (i) Martin-Löf-test jest rekurencyjnie przeliczalnym ciągiem $(G_m)_{m\in N}$ takim, że $\forall m \in N \lambda G_m \leq 2^{-m}$.[47]

(ii) Zbiór $Z \subseteq N$ nie przechodzi testu wtedy, gdy $Z \in \bigcap_m G_m$, w przeciwnym razie Z przechodzi test.

(iii) Z jest Martin-Löf-przypadkowy wtedy, gdy Z przechodzi każdy Martin-Löfa-test.

Nieformalnie, Martin-Löfa-test na przypadkowość wychwytuje strukturę, porządek, wzór. Jeżeli zbiór ich nie posiada, to jest przypadkowy. Albo inaczej, jeżeli zbiór nie należy do borelowskiej $\prod_2^0$ *null klasy*, to jest przypadkowy.[48] Analogicznie w przypadku ciągu/liczby rzeczywistej. W rezultacie ciąg nie podlega kompresji. Dodać trzeba, że testy same w sobie są obiektami, które można opisać, czyli tylko obliczalnie wiele *null klas* jest danych przez testy.

7.3. Przypadkowość implikuje nieobliczalność (ale nie *vice versa*)

Aby pokazać, że algorytmiczna przypadkowość pociąga nieobliczalność przedstawimy wpierw algorytmiczną parafrazę twierdzenia Gödla o niezupełności.

TWIERDZENIE 8. Niech L będzie teorią aksjomatyczną zawierającą arytmetykę (Peano), której arytmetyczne konsekwencje są prawdziwe.

(i) Istnieje stała c_L taka, że wewnątrz L żadne zdanie postaci

"$H(n) > C_L$" nie jest dowodliwe.

(ii) Niech $\#(L)$ będzie rekurencyjnie przeliczalnym indeksem L. Wtedy istnieje pewna stała c' niezależna od L taka, że wewnątrz L żadne zdanie postaci "$H(n) > H(\#(L)) + c'$" nie jest dowodliwe.

(iii) Istnieją szczególne teorie, których aksjomaty mają zawartość informacyjną $H(aksjomatów) = m + O(1)$, w których jest możliwość ustalenia wszystkich prawdziwych twierdzeń postaci "$H(x) = k$", z

$k < H(aksjomatów) + O(1)$, oraz postaci

"$H(x) \geq H(aksjomatów) + O(1)$".[49]

---

[46] Z rodziny wszystkich zbiorów zwartych w przestrzeni o pewnej strukturze (algebraicznej), poprzez branie przeliczalnych sum, różnic i przecięć, można utworzyć zbiory borelowskie.

[47] Zbieżność ciągu do granicy musi być efektywna.

[48] Zbiory przypadkowe nazywane są też normalnymi zbiorami Borela.

[49] G. J. Chaitin, *Algorithmic Information Theory*, Cambridge University Press, Cambridge 1987.

Istnieją zatem systemy formalne o skończonej liczbie aksjomatów, a tym samym skończonej zawartości informacyjnej, które mogą dostarczać obiekty (twierdzenia) o dowolnie wysokiej zawartości informacyjnej. Chaitin dowiódł następnie, że czas obliczeń dla twierdzeń postaci "$H(x) = k$", z

$k < m = H(aksjomatów) + O(1)$,

oraz "$H(x) \geq m = H(aksjomatów) + O(1)$" jest nieobliczalny.[50] Wiadomość ta jest zawarta w dwóch poniższych twierdzeniach.

TWIERDZENIE 9 (ograniczenie na czas obliczeń i złożoność obliczeniową). Albo program p staje w cyklu czasowym mniejszym niż

$\sum(H(p) + O(1))$, albo nigdy nie staje. Z tego względu, jeżeli definiujemy

$d(n) = \max_{|x^*| \leq n} H_D(x) = \max_{|x^*| \leq n} D(x)$, to

$$d(n) = \sum(n + O(1))$$

$\sum(n + O(1))$ jest minimalnym czasem $d(n)$ przy którym wszystkie programy o złożoności $\leq n$ (faktycznie) stają.

Dowód: Dowodzimy tylko, że nie istnieje górna granica dla programów o algorytmicznej informacji $\leq n$, czyli dla obiektów x z $H(x) \leq n$, $H_D(x)$, $D(x) \leq d(n) \leq \sum(n + O(1))$. Rozważamy dwa uniwersalne komputery U i U'. Dowolny program p z $H(p) \leq n$ symuluje $U'(p)$, a dodatkowo liczy cykl czasowy t dla U', do momentu aż p stanie. Teraz, jeżeli $d(n)$ jest określony, to istnieje przynajmniej jeden program p_L o algorytmicznej informacji $\leq n$, który zajmuje najwięcej czasu i taki, że $U(p_L) = d(n)$. Odtąd, niezależnie od n, symulacja U' na U oraz liczenie czasu wymaga $O(1)$ dodatkowych bitów programu, czyli $H(d(n)) \leq n + O(1)$. Zatem $d(n) \leq \sum(n + O(1))$.

TWIERDZENIE 10. $\sum$ nie jest efektywnie obliczalna.

Drugi dowód opiera się na tym, że gdyby czas obliczeń miał być obliczalny, to musiałby istnieć algorytm stopu. Obliczeniowa złożoność związana ze skończonymi lub nie obiektami jest więc nieobliczalna. Przypadkowość implikuje nieobliczalność.

7.4. Chaos

Ograniczamy się teraz do pojedynczego przypuszczenia, że chaos w świecie fizycznym koresponduje z przypadkowością w matematyce. W dalszym ciągu będziemy też korzystać z pojęcia przypadkowej liczby rzeczywistej. Mając w pamięci, że nie można jej efektywnie obliczyć.

---

[50] G. J. Chaitin, *Information, Randomness and Incompleteness*, Word Scientific, Singapore 1987.

DEFINICJA 20. Liczba rzeczywista r jest Martin-Löf-przypadkowa wtedy, gdy nie jest zawarta w dowolnym zbiorze nieskończonego rekurencyjnie przeliczalnego ciągu A_i zbiorów interwałów takim, że miara $\mu(A_i)$ jest zawsze mniejsza lub równa $2^{-i}[\mu(A_i) \leq 2^{-i}]$. Czyli, że r jest Martin-Löf-przypadkowa wtedy, gdy

$$\forall i[\mu(A_i) \leq 2^{-1}] \Rightarrow \neg\forall i[r \in A_i]$$

Z czysto algorytmicznego punktu widzenia chaos deterministyczny charakteryzowany jest przez:

(i) efektywnie obliczalną/rekurencyjną/deterministyczną ewolucję;

(ii) własność nieliniowego systemu ewolucji do wykładniczego w czasie rozchodzenia się początkowo bliskich trajektorii (dodatnie wykładniki Lapunowa);

(iii) przypadkowe wartości początkowe.

Przypadkowość jest tu definiowana najczęściej jako Martin-Löf-przypadkowość. Można argumentować też w ten sposób, że jeżeli liczba rzeczywista jest elementem continuum, to prawdopodobieństwo, że jest Martin-Löf-przypadkowa równa się jeden. „Prawie wszystkie" wartości początkowe są przypadkowymi liczbami rzeczywistymi. Idea chaosu deterministycznego opiera się na spostrzeżeniu, że przypadkowość, albo niekompletna informacja wartości początkowej, ujawnia się w trakcie ewolucji. Dlatego kryterium specyfikacji chaosu jest obecność odpowiedniej ewolucyjnej funkcji zdolnej odsłonić informację „prawdziwej", lecz nieznanej wartości początkowej x_0. Szczegóły sprawy są domeną fizyków, jednakże, albo tak zwana niepewność δx_0 wartości początkowej, albo korespondująca zmienność wartości początkowej, zmienia się w czasie. Jako miarę separacji dwóch różnych wartości początkowych przyjmuje się wykładnik Lapunowa λ.[51] Scenariusz jest więc zaskakujący, bo efektywnie obliczalna funkcja dostarcza Martin-Löf-przypadkową ewolucję systemu „odsłaniając" informacje zawartą w Martin-Löf-przypadkowej wartości początkowej. Przypadkowość jest tu jednak pierwotnie usytuowana w wartości początkowej.[52] Chociaż dla celu przez nas postanowionego nie jest ważne, gdzie ona pierwotnie rezyduje, tylko że w ogóle znalazła miejsce. Można zatem sformułować następującą równoważność:

chaos $\Leftrightarrow$ wrażliwość na warunki początkowe $\Leftrightarrow$ algorytmiczna przypadkowość

---

[51] Zob. H. G. Schuster, *Deterministic Chaos*, Phisik Verlag, Weinheim 1984.

[52] Zachowanie (w sensie sekwencji na wyjściu) może okazać się chaotyczne nawet wtedy, gdy wartości początkowe nie są przypadkowe, por. S. Wolfram, *New Kind of Science*, Wolfram Media Inc. 2002.

Jeśli więc fizyka słusznie wykorzystuje teorię algorytmicznej informacji do specyfikacji układów chaotycznych, to mamy nieobliczalność w świecie fizycznym.[53]

7.5. Nie istnieje algorytm wycinający okresowość w rozwiązaniach równań ruchu

Zidentyfikujmy teraz miejsce postępowania:

1. Rozważamy przestrzeń Q i R oraz maszynę T, która może gromadzić binarne liczby rzeczywiste i wykonywać na nich skończoną liczbę operacji. Maszyna reprezentuje standardowy model obliczeń.

2. Każda liczba rzeczywista jest granica ciągu liczb wymiernych. Mamy więc rzeczywiste obliczanie na ciągach Cauchy'ego.

3. Liczba rzeczywista jest (naiwnie) obliczalna wtedy, gdy istnieje ciąg $(a_n) \subseteq Q$, z $x = \lim_n a_n$. Wiadomo jednakże, że istnieje obliczalny ciąg liczb wymiernych, który jest nieefektywnie zbieżny do nieobliczalnej liczby rzeczywistej.

4. Poszukując nieobliczalności stwierdziliśmy, że przypadkowość w sensie Martin-Löf-przypadkowości pociąga nieobliczalność (ale nie vice versa). Poprawność przejścia gwarantują rezultaty Chaitina oraz twierdzenie o rekurencyjnej nierozwiązywalności problemu stopu. Sama Martin-Löf-przypadkowość wyznaczona została przez Martin-Löf-test na przypadkowość.

5. Zgodnie z teorią algorytmicznej informacji istnieją obiekty, których oryginalna deskrypcja nie może być „ściśnięta" do postaci krótszej. Są to obiekty przypadkowe, w szczególności przypadkowe liczby rzeczywiste. Można powiedzieć też tak: obserwacja segmentu początkowego liczby rzeczywistej nie pozwala przewidzieć „całej" liczby.

6. Okazuje się, że pojęcie „obiekt nie podlegający kompresji" daje się wykorzystać do charakteryzacji chaosu (deterministycznego).

Wobec zaprezentowanych faktów wykazanie, że nie istnieje program oddzielający okresowość od nieokresowości w rozwiązaniach równań ruchu, jest trywialne. Wiadomo, że gdyby taki algorytm istniał, to musiałby istnieć algorytm stopu. Łatwo zauważyć, że kluczowym ogniwem rozumowania jest obecność logicznej nieefektywności w maszynie matematycznej, a ogólnie, w obrębie instrumentarium, które służy do obliczania przyszłości. Co się tyczy implementacji tej nieobliczalności w naturze, to algorytmicznie zinterpretowana teoria chaosu daje podstawy do przekonania, że procesy przypadkowe i nieobliczalne mają jednak miejsce. W rezultacie można stwierdzić, że

---

[53] Przykładem układu chaotycznego charakteryzowanego algorytmicznie jest *koło ruletki plus krupier*; zob. R. W. Batterman, *Chaos: Algorithmic Complexity vs. Dynamical Instability*, w: *Law and Prediction in the Light of Chaos Research*, Weingartner P. Schurz G. (red.), Springer, Berlin 1996; jak również wpomniany wcześniej *układ dwóch i więcej ciał.*

klasa równań ruchu jest nierozstrzygalna. Podobnie jak zbiór zdań $(S_1,...,S_n)$ zbudowanych z ich pomocą. Oznacza to, że prawda o przyszłości transcenduje wszelkie jej możliwe diagnozy. Trzeba jednak podkreślić, że konkluzję tą daje się wyprowadzić niezależnie od tego, czy zachodzi implementacja nierozstrzygalności w świecie realnym. Wystarczającym powodem jest obecność nierozstrzygalnych modeli matematycznych, które wykorzystujemy do jego deskrypcji.

BIBLIOGRAFIA

Adamowicz Z., Zbierski P., *Logika matematyczna*, PWN, Warszawa 1991.

Batterman R. W., *Chaos: Algorithmic Complexity vs. Dynamical Instability*, w: Weingartner P, Schurz G.(eds.), *Law and Prediction in Light of Chaos Research*, Springer, Berlin–Heidelberg, New York 1996.

Benett. C. H., *Dissipation, Information, Computational Complexity and the Definition of Organization*, w: *Emerging Synthesis in Science*, red. Pines D., Academic Press, New York 1985.

Chaitin G. J., *Algorithmic Information Theory*, Cambridge University Press, Cambridge 1987.

Chaitin G. J., *Information, Randomness and Incompleteness*, World Scientific, Singapore 1987.

Davis M., Putnam H., Robinson J., *The decision problem for exponential diophantine equations*, Annals of Mathematics, 74 (1961).

Dummett M., *Realism and Anti-Realism*, w: *The Seas of Language*, Clarendon Press, Oxford 1993, s. 462–478; polskie tłum.: Szubka T. (1998), *Realizm i antyrealizm*, w: *Filozofia brytyjska u schyłku XX wieku*, red. Gutowski P., Szubka T., TN KUL, Lublin.

McEliece R. J., *The Theory of Information and Coding*, w: *Encyclopedia of Mathematics and Its Applications*, v. 3, London 1977.

Heyting A., *Der intuitionistische Grundlegung der Mathematik*, Erkenntnis (1931).

Jones J. P., *Diophantine Representation of the Set of Prime Numbers*, American Math. Monthly 83 (1976).

Kleene. S. C., *Turing's Analysis of Computability, and Mayor Applications of It*, *in the Universal Turing Machine,* A Half-Century Survey, ed. by Herken R., Kammerer & Unverzagt, Hamburg 1988.

Krajewski S., *Twierdzenie Gödla a filozofia*, Studia Filozoficzne nr 6/7 (1988).

Kreisel G., *Church's Thesis: a Kind of Reducibility of Axiom for Constructive Mathemtics*, w: *Intuitionism and Proof Theory*, North-Holland Publ. Comp., Amsterdam–London 1970.

de Laplace P. S., , *A Philosophical Essay on Probability*, Dover, New York 1951, franc. org., *Essai philosophique sur les probabilites* 1814.

Malinowski G., *Logiki wielowartościowe,* PWN, Warszawa 1990.

Martin-Löf P, *On the Notion Randomness*, w: *Intuitionism and Proof Theory*, North-Holland Publ. Comp., Amsterdam–London 1970.

Matijasiewicz J. V., *Diofantnost pierieczyslimych mnożestv*, Dokl. Akad. SSSR 191, 2 (1970).

Mazur S., *Computable Analysis*, Rozprawy Matematyczne XXIII, PWN., Warszawa 1963.

Murawski R., *The Present State of Mechanized Deduction, and Present Knowledge of its Limitations*, w: *Issues of Decidability and Tractability*, University of Bialystok 2002.

Nies A., *Computability and Randomness*, Oxford Logic Guides 51, Oxford University Press 2009.

Papadimitriou C. H., *Złożoność obliczeniowa*, Wydawnictwa Naukowo-Techniczne, Warszawa 2002.

Pour-El M. B., Richards J. I., *Computability in Analysis and Physics*, Springer, Berlin–Heidelberg 1989.

Schuster H. G., *Deterministic Chaos*, Physik Verlag, Weinheim 1984.
Svozil K., *Randomness & Undecidability in Physics*, World Scientific Publ., Singapore, New Jersey–London–Hong Kong 1993.
Tarski A., *Pojęcie prawdy w językach nauk dedukcyjnych*, Towarzystwo Naukowe Warszawskie, Warszawa 1933.
Tarski A., *Remarks of Alfred Tarski*, Revue Internationale de Philosophie, 27–28, (1986),
Tarski A., Mostowski A. W., Robinson R. M., *A General Method in Proofs of Undecidability*, w: *Undecidable Theories*, North-Holland Publ. Comp., Amsterdam 1953.
Trzęsicki K., *From the Idea of Decidability to the Number Ω*, w: *Issues of Decidability and Tractability*, University of Bialystok (2006).
Turing A., *On Computable Numbers with an Application to the Entscheidunsproblem*, Proc. Lond. Math. Soc., Ser. 2, 42 (1936).
Weingartner P., Schurz G., red., *Law and Prediction in Light of Chaos Research*, Springer, Berlin 1996.
Weyl H., *David Hilbert and his Mathematical Work*, Bulletin of the Amer. Math. Soc., (1944).
Woleński J., *Epistemologia*, Wydawnictwo Naukowe PWN, Warszawa 2005.
Wolfram S., *New Kind of Science*, Wolfram Media Inc. 2002.

FUTURE TRUTH AND THE CONCEPT OF COMPUTABILITY

ABSTRACT

The text is devoted to the problem of the application of the computability theory to empirical knowledge on future events. The fundamental problem examined in this paper is the following one: how to connect the intuitive concept of impredictability with the exact concept of computability? Central to this line of thought is the problem: can the real world be modeled on a computer? The author assumes: firstly, that the knowledge about the future is stored in statements, secondly, that a way of its acquiring depends on the usage of mathematical formulas which describe the evolution in time, that is, which are motion equations (statements about the future are statements about the positions of objects in the future). The task is trivial: to show that the statements about the future can be formulated in a way that demands the non-algorithmic mathematics application. More precisely: to show that for every motion equation and all possible initial (input) data, there is no programme which answers yes/no to the question whether the equation has the periodic solutions or no. The scheme of reasoning is as follows: we assume (*reductio ad absurdum*) that the Turing machine that solves every motion equation is equipped with the programme that cuts out periodic solutions. The testing of periodicity follows from the fact that its existence would show the computability of operation/function. It is obvious that the initial data must be computable/recursive. From that that the set real numbers is infinite/denumerable we can admit that some operation (function) will be uncomputable (results: Banach/Mazur, Turing, Pour-El/Richards, Chaitin, Batterman). That allows to apply the "Stop-Theorem" for the Turing machine, and, in the effect, to set forth that the class of motion equations is undecidable.

Keywords: theory of computability, knowledge about the future, philosophy.

Adres Autora: awilk@ifispan.waw.pl