PAWEŁ STACEWICZ

Politechnika Warszawska
Wydział Administracji i Nauk Społecznych

Czy informatykom musi wystarczyć nieskończoność potencjalna?

Tytuł niniejszego artykułu sugeruje, że w informatyce istnieje praktyka posługiwania się obiektami nieskończonymi. Sugestia taka może zaskakiwać, ponieważ systemy informatyczne kojarzą się raczej z czymś skończonym, na przykład ze skończonym zbiorem instrukcji programu komputerowego czy skończoną liczbą elementów w układzie scalonym. Efekt zaskoczenia ma łagodzić ostatnie słowo tytułu, które wyjaśnia, że współczesnych informatyków interesuje raczej nieskończoność „słaba", zwana potencjalną (w opozycji do aktualnej czy urzeczywistnionej).

Być może jednak silniejsze odwołania do struktur i procesów nieskończonych są w informatyce niezbędne… Tego tytuł nie wyklucza, co sygnalizuje wieńczący go znak zapytania. Potraktujmy ten znak jako zaproszenie do dyskusji inicjowanej poniższym tekstem.

1. Nieskończoność w informatyce: dwa aspekty

1.1. Stosunek informatyków do nieskończoności można określić jako ambiwalentny. Z jednej strony, gdy koncentrują się na *teoretycznych modelach obliczeń*, używają nieskończoności bardzo chętnie. Wystarczy tu wspomnieć nieograniczoną co do długości taśmę maszyn Turinga (która stanowi element modelu obliczeń cyfrowych), czy nieskończoną precyzję przetwarzania danych ciągłych (którą zakłada się w koncepcji obliczeń analogowych). Warto przywołać także nieco bardziej egzotyczne, lecz w pracach informatyków jak najbardziej obecne, teorie maszyn, które mogłyby wykonywać nieskończoną liczbę operacji w skończonym czasie. Jedną z nich jest przyspieszająca maszyna Turinga (por. Shagrir 2004).

Z drugiej strony, w odniesieniu do *faktycznych implementacji obliczeń*, nieskończoność postrzega się jako źródło rozlicznych kłopotów. Kłopotów, które wyznaczają nieprzekraczalne *granice* informatyki[1]. Na przykład, istota problemów nieobliczalnych cyfrowo polega na tym, że każda próba ich rozwiązania prowadzi do hipotetycznie nieskończonych sekwencji operacji, o których nie da się w skończonym czasie stwierdzić, czy faktycznie będą wykonywane w nieskończoność. W ich przypadku zatem kłopotliwe odniesienie do nieskończoności pojawia się na dwóch poziomach: operacyjnym i metodologicznym. Istota problemów NP-trudnych z kolei wyraża się w tym, że trzeba je rozwiązywać za pomocą ogromnej, tj. praktycznie nieskończonej, liczby elementarnych kroków[2]. Warto podkreślić, że przypadłość ta dotyczy każdego możliwego rozwiązania. Zauważmy jeszcze, że w dziedzinie zadań wymagających zastosowania programów uczących się infinitystyczny kłopot może pojawić się na poziomie struktury programu. Aby bowiem schemat uczenia się zapewniał maksymalną elastyczność programu (a więc zdolność do możliwie szerokich zmian w interakcji ze środowiskiem), trzeba go modyfikować na nieskończenie wielu poziomach, związanych z kolejnymi, coraz głębszymi, kryteriami zmian jego struktury (zob. Stacewicz 2016).

1.2. Już powyższe uwagi wskazują, że rozważając informatyczne zagadnienie nieskończoności trzeba rozróżnić jego dwa aspekty: *teoretyczny* (inaczej: formalny) oraz *fizyczny* (inaczej: realny).

Ten pierwszy wiąże się ze *stricte* formalnymi badaniami nad różnymi modelami obliczeń (jak cyfrowy czy analogowy); ten drugi dotyczy faktycznych realizacji obliczeń za pomocą fizycznych nośników i systemów. Przy podejściu pierwszym nieskończoność postrzega się jako własność *teoretyczną* (np. teoretycznie niekończący się czas trwania procedury); przy drugim – jako coś fizycznie *realizowalnego*, a więc obecnego w realnym świecie. Choć obydwa aspekty przenikają się – np. modele teoretyczne opisują różne sposoby fizycznej implementacji obliczeń – to z metodologicznego punktu widzenia warto je rozdzielić.

Dopowiedzmy krótko, że poczynione rozróżnienie zgadza się dobrze z „podwójną" naturą informatyki, która z pewnych względów uchodzi za dyscyplinę *formalną* (pokrewną matematyce), z innych natomiast – za naukę *empiryczno-techniczną*. Jej rys formalny ujawnia się najpełniej w obszarze badań nad algo-

---

[1] Są one nieprzekraczalne w tym sensie, że aby je przekroczyć, to znaczy algorytmicznie rozwiązać dany problem, trzeba założyć jakieś odniesienie do nieskończoności.

[2] Przypomnę, że każdy problem NP-trudny cechuje co najmniej wykładnicza złożoność czasowa. Złożoność taka określa średnią liczbę operacji elementarnych, jakie maszyna rozwiązująca problem musi wykonać dla danych o rozmiarze n. Przykładowo: dla złożoności rzędu 2^n i rozmiarze danych 50, otrzymujemy 2^{50} operacji do wykonania – co jest wielkością astronomiczną (praktycznie nieskończoną).

rytmami[3] oraz obliczalnością problemów; empiryczny zaś – w dziedzinie konstrukcji fizycznych automatów do przetwarzania danych (urządzeń w głównej mierze elektronicznych)[4] (por. Knuth 1974; Denning 2005).

2. Dwa sposoby rozumienia nieskończoności potencjalnej w informatyce

Dwojakie podejście informatyków do obiektów nieskończonych – z jednej strony formalne, z drugiej zaś fizykalne – znajduje swoje rozwinięcie w dwojakim sposobie rozumienia *nieskończoności potencjalnej*. Jej pojęcie znamy przede wszystkim z matematyki, gdzie nieskończoność potencjalną przeciwstawia się aktualnej[5]. Prócz matematyki jednak istnieje inny, bliższy potocznym intuicjom, sposób „dojścia" do interesującego nas typu nieskończoności. Zgodnie z nim potencjalne to tyle, co *hipotetyczne*, a mówiąc inaczej, hipotetycznie mogące zaistnieć. Z takiego punktu widzenia nieskończoność potencjalna byłaby czymś hipotetycznym, założonym, rozważanym tylko i wyłącznie w sposób teoretyczny.

Poniżej rozważę obydwa punkty widzenia, wskazując dodatkowo ich wzajemne powiązania.

2.1. Znane z filozofii matematyki rozróżnienie między nieskończonością aktualną i potencjalną przedstawia się następująco (zob. też Murawski 2014a). Obiekty *nieskończone aktualnie* rozumie się jako nieskończenie liczne całości, które jako takie właśnie można poddawać pewnym „całościowym" operacjom, np. porównywać ze sobą czy przekształcać w innego typu obiekty. Jako najprostsze i najbardziej powszechne przykłady wspomnianych całości podaje się zbiory liczb naturalnych (N) i rzeczywistych (R)[6].

---

[3] Niektórzy metodologowie uważają algorytmikę za jądro informatyki. Por. tytuł książki Davida Harela (2000): *Algorytmika. Rzecz o istocie informatyki*, a także artykuł Donalda Knutha (1974) o związkach informatyki z matematyką.

[4] Innego typu przejawem empiryczności informatyki są tzw. *obliczenia naturalne* (ang. *natural computing*). Nazwą tą określa się wszelkiego rodzaju techniki obliczeniowe, które w istotnym stopniu zależą od zachodzących w przyrodzie procesów – wykorzystywanych wprost (czerpanych z przyrody) lub nie wprost (poprzez realizację w systemach sztucznych obserwowanych w przyrodzie reguł przetwarzania danych). Do technik naturalnych pierwszego typu należą obliczenia kwantowe, do kategorii drugiej zalicza się chociażby obliczenia koneksyjne (realizowane przez sztuczne sieci neuronowe) i ewolucyjne (por. Kari, Rozenberg 2008; Rozenberg, Back, Kok 2012).

[5] Wprowadzenie tego rozróżnienia zawdzięczamy Arystotelesowi (por. Murawski 2014a).

[6] W odniesieniu do zbiorów N i R można podać następujące typy wspomnianych wyżej całościowych operacji: a) *porównywanie* liczebności – w sensie cantorowskim zbiory N i R nie są równoliczne, czyli przysługują im nieskończoności różnych „rzędów"; b) *zwężanie* – zbiór R można za pomocą odpowiednio dobranej funkcji przekształcić w równoliczny z nim przedział, np.

Inny rodzaj matematycznych obiektów, które w przeciwieństwie do poprzednich nazywa się *nieskończonymi potencjalnie* (nie zaś: aktualnie czy całościowo), ma charakter procesualny. Są to w istocie pewne *niekończące się procesy* (sekwencje), których przebieg, a także kolejne wyniki cząstkowe, określa pewna precyzyjna reguła. To w tej właśnie regule tkwi nieskończona potencja generowania kolejnych, różnych od siebie, wielkości pewnego typu. Dobrym przykładem obiektu potencjalnie nieskończonego jest ciąg liczbowy – rozważany jednak nie jako całość możliwych do uzyskania elementów, lecz jako zawsze „niegotowa", nigdy niezakończona sekwencja kolejnych wartości. Rozważmy dla ilustracji ciąg liczb naturalnych $\{a_n\}$, który określa reguła postaci „$a_0 = 1$ i $a_n = 2n + 1$ dla kolejnych n naturalnych" (albo inaczej i opisowo: do każdej kolejnej wartości począwszy od 1 dodawaj 2). W przypadku tak określonego ciągu mamy do czynienia z nieskończonym potencjalnie (tj. wciąż generowanym) „zbiorem" liczb nieparzystych[7].

Ponieważ pojęcie nieskończoności potencjalnej zakłada istnienie pewnej reguły determinującej przebieg „coś-liczącego" procesu, pozostaje ono bliskie informatyce. W kontekście informatycznym wspomnianą regułę trzeba utożsamić z programem definiującym sekwencję kroków wiodących od danych początkowych do wyniku. Odniesienie do nieskończoności potencjalnej staje się szczególnie wyraźne wówczas, gdy program służy do rozwiązywania problemów z dowolną zadaną dokładnością[8]. Im wyższą dokładność określi użytkownik, tym więcej kroków musi wykonać program i tym lepszy wynik generuje. Mówiąc zaś inaczej: potencjalnie nieskończonemu ciągowi coraz wyższych dokładności odpowiada niekończący się szereg coraz lepszych wyników.

Wkroczywszy na teren informatyki, warto przywołać jeszcze jeden, niezwykle ważny dla dalszych wywodów, przykład różnicy między nieskończonością potencjalną i aktualną. Chodzi o liczby *obliczalne* (ang. *computable*) i *nieobliczalne* (ang. *uncomputable*). Ich rozróżnienie zawdzięczamy Alanowi Turingowi, który w roku 1936 opracował koncepcję abstrakcyjnej maszyny programowalnej i za jej pomocą dokonał nowego, nieznanego dotychczas, podziału liczb rzeczywistych (*de facto*: niewymiernych) na dwie dopełniające się części (zob. Turing 1936). Liczby *obliczalne* zdefiniował jako te spośród rzeczywistych, dla których istnieje obliczająca je maszyna Turinga – obliczająca je cyfra po cyfrze,

---

(0,1); c) *wyodrębnianie* – ze zbioru R można wyodrębnić jako jego podzbiór właściwy zbiór N, czyli inny obiekt nieskończony o innej mocy.

[7] Warto dopowiedzieć, że w przypadku typowych dla analizy matematycznej wyrażeń typu $\lim_{n\to\infty} f(n)$ mamy do czynienia z nieskończonością potencjalną. Zapis „$n \to \infty$", czyli „n dąży do nieskończoności", znaczy tyle, że zgodnie z pewną regułą n staje się coraz większe (w przypadku n naturalnych chodzi o regułę „$n \to n + 1$").

[8] Przykładem takiego problemu jest wyznaczanie całki Riemanna (w podanym przedziale) za pomocą jednej z metod numerycznych (zob. np. Fortuna, Macukow, Wąsowski 2015).

z dowolną zadaną dokładnością. Liczby *nieobliczalne* określił zaś jako takie, dla których wspomniana maszyna-program nie istnieje. Udowodnił przy tym, że liczb nieobliczalnych musi być nieskończenie wiele[9].

Przywołane charakterystyki każą stwierdzić, że liczbom obliczalnym, o ile mają one nieskończone przedstawienie[10], przysługuje nieskończoność *potencjalna*, określona ściśle przez generujący te liczby program (generujący je z dowolną zadaną dokładnością)[11]. Liczby nieobliczalne natomiast cechuje nieskończoność *aktualna*. Ponieważ nie istnieje wyznaczająca je reguła, trzeba je traktować jako zdefiniowane w określony sposób nieskończone całości – całości rozumiane jako ogół tworzących je cyfr, których jednak nie sposób sukcesywnie wyznaczać za pomocą skończonego programu.

Rozróżnienie powyższe jest dla informatyki niezwykle ważne, ponieważ programy komputerowe można kodować jako liczby (chociażby ciągi zer i jedynek) i dzięki temu pewne rozważania dotyczące programów prowadzić w „przestrzeni liczb". Wobec takiej możliwości pojawia się bardzo interesujące pytanie o to, czy odkrytym przez Turinga liczbom nieobliczalnym (nieskończonym aktualnie) odpowiadają jakieś programy. Temat rozwinę w punkcie 3.3.

2.2. Kolejny z anonsowanych wcześniej sposobów rozumienia nieskończoności potencjalnej nie jest osadzony tak głęboko w filozofii matematyki, wywodzi się jednak z refleksji nad stosunkiem teorii (również matematycznych) do rzeczywistości. Rozważając ów stosunek, dochodzi się do przekonania (domysłu), że obiektom teoretycznym (w tym: liczbom) przysługuje dwojaki sposób istnienia. W samej teorii istnieją one faktycznie – jako obiekty określone za pomocą definicji lub poprawnych dowodów istnienia; poza teorią natomiast istnieją *hipotetycznie* lub *potencjalnie* – jako coś, czego byt muszą dopiero potwierdzić wiarygodne obserwacje i eksperymenty. Odpowiednie przykłady podam w kolejnych akapitach.

Hipotetyczny, a więc ograniczony do pewnej teorii, sposób istnienia przysługuje między innymi obiektom *nieskończonym* (które zresztą stanowią konieczny składnik wielu teorii matematycznych). I w tym właśnie, różnym od

---

[9] Dokładniej zaś: zbiór liczb nieobliczalnych ma moc *continuum*, czyli jest równoliczny ze zbiorem liczb rzeczywistych. Więcej informacji o liczbach nieobliczalnych zawierają następujące źródła: 1) Chaitin 1998 oraz 2) wpis w blogu Cafe Aleph pt. *O liczbach nieobliczalnych na chłopski rozum* autorstwa P. Stacewicza (<http://marciszewski.eu/?p=4490>, dostęp: 20.04.2017).

[10] Wśród liczb obliczalnych nie wszystkie są nieskończone w sensie posiadania nieskończonej reprezentacji dziesiętnej. Na przykład, wszystkie liczby naturalne są obliczalne i w powyższym sensie skończone; natomiast wymierne liczby okresowe są obliczalne i nieskończone (podobnie jak wszystkie obliczalne liczby wymierne, jak e czy π).

[11] Na przykład: dla niewymiernej liczby obliczalnej π (pi) ów program może być tożsamy z programem liczącym sumy cząstkowe następującego szeregu: $4 \cdot \sum_{n=0}^{\infty} \frac{(-1)^n}{2n+1}$.

poprzedniego, sensie można mówić o nieskończoności potencjalnej. Jest ona *potencjalna*, ponieważ postuluje się ją w teorii, nie przesądzając przy tym, czy istnieje w świecie.

I tak: w fizyce teoretycznej wolno rozważać *potencjalną* podzielność materii *ad infinitum*, powołując się nawet na możliwość definiowania liczb dowolnie bliskich zeru; nie wiadomo jednak, czy własność ta faktycznie materii przysługuje. W informatyce z kolei: proponuje się teoretyczne modele maszyn *analogowych* – maszyn, które potrafią uwzględniać nieskończenie małe różnice między wielkościami ciągłymi; nie wiadomo jednak, czy automaty takie są realizowalne w praktyce (por. Mycka 2010).

2.3. Podsumowując powyższe analizy i wyjaśnienia, mogę stwierdzić, co następuje. Z uwagi na dualny, tj. formalno-fizykalny, status badań informatycznych (zob. 1.2), należy rozróżnić dwa informatyczne sposoby rozumienia nieskończoności potencjalnej.

W sensie *matematycznym* (formalnym) nieskończoność jest *potencjalna*, jeśli nie istnieje jako całość – istnieje zawsze *fragmentarycznie*, na mocy dobrze określonej reguły, która pozwala w sposób niewyczerpywalny generować kolejne, różne od siebie wielkości pewnego typu. Znaczenie to pojawia się zatem w ramach matematycznej opozycji potencjalne – aktualne.

W sensie *fizycznym* natomiast nieskończoność jest *potencjalna*, jeśli istnieje *hipotetycznie*, na poziomie teorii, która niekoniecznie musi mieć przełożenie na rzeczywistość i osadzone w niej praktyczne rozwiązania. W tym znaczeniu obiekty potencjalnie nieskończone są hipotetycznymi konstrukcjami ludzkiego umysłu[12].

3. Nieskończoność z perspektywy maszyn Turinga

Uniwersalna maszyna Turinga – opisana pierwotnie w pracy Turinga (1936) – uchodzi za najbardziej sugestywny model programowalnych komputerów cyfrowych. Sugestywność ta bierze się po pierwsze z faktu, że jest to raczej prosty obiekt matematyczny, a po drugie stąd, że przypomina on realną maszynę – z taśmą, głowicą, rejestrem stanów i mechanizmem sterującym. Śmiało można powiedzieć, że obecne w turingowskim modelu połączenie abstrakcji z fizycz-

[12] Obydwa zestawione ze sobą sposoby rozumienia nie pokrywają się. Dowodzi tego chociażby fakt, że nieskończoność potencjalną w sensie fizycznym wolno przypisywać zarówno takim obiektom matematycznym, które są nieskończone potencjalnie, jak i aktualnie. Obydwa zatem typy obiektów nieskończonych można uważać za obiekty teoretyczne (być może tylko więc czysto potencjalnie realizowalne w świecie fizycznym). W szczególności: niekończące się procesy obliczeniowe (n esk potencjalna w sensie matematycznym) można uważać za równie fizycznie „podejrzane", jak nieskończenie liczne całości (n esk aktualna w sensie matematycznym).

nością określiło wstępnie przyszły profil informatyki (jako nauki po części formalnej, a po części empirycznej).

3.1. Na pierwszy rzut oka jedyny nieskończony element maszyny Turinga stanowi *taśma* – będąca teoretycznym odpowiednikiem pamięci realnych komputerów. Wnikliwsza analiza odsłania jednak dwa inne, powiązane z nieskończonością, aspekty omawianego modelu. Po pierwsze, jest to zapewniana przez wspomnianą taśmę dowolna, a więc nieskończenie duża, *dokładność obliczeń*. Własność ta stanowi o sile modelu i opartych na nim konstrukcji: zależnie od żądanej precyzji obliczenia mogą trwać krócej lub dłużej, zawsze jednak, w skończonej liczbie kroków, pozwalają uzyskać odpowiednio dokładny wynik. Aspekt drugi odpowiada z kolei za poważne ograniczenia modelu. Chodzi o możliwość powstawania *nieskończonych sekwencji* obliczeń, o których nie sposób stwierdzić w skończonym czasie, czy kiedykolwiek się skończą (ewentualnie: czy będą powtarzane w nieskończoność z zachowaniem pewnego regularnego cyklu[13]). Własność ta powoduje, że istnieją problemy *nieobliczalne* zasadniczo – do których należą m.in. problem stopu maszyn Turinga czy zagadnienie rozwiązywalności równań diofantycznych (zob. Harel 2000).

Warto podkreślić, że obydwie wskazane własności – pozytywna i negatywna – odnoszą się odpowiednio do nieskończoności *potencjalnej* i *aktualnej* (w sensie matematycznym). Nieskończoną dokładność obliczeń – którą zapewnia nieograniczenie długa taśma maszyny – trzeba interpretować *potencjalnie*, ponieważ chodzi o dokładność „na życzenie". Im większej dokładności zażyczy sobie użytkownik, tym więcej obliczeń maszyna wykona i tym więcej zasobów (komórek taśmy) zużyje. Podkreślmy: maszyna nie jest w stanie przeprowadzić nieskończenie wielu obliczeń, lecz może ich wykonywać dowolnie dużo. Innymi słowy, program maszyny zapewnia potencjalnie dowolną (w domyśle: coraz większą) precyzję rezultatu[14].

Nieskończone i nieregularne sekwencje obliczeń są z kolei przykładem nieskończoności *aktualnej*. Z matematycznych rozumowań wiadomo, że mogą one zaistnieć i że występują przy okazji rozwiązywania określonych problemów. Mimo to za pomocą żadnego skończonego programu (dla maszyny Turinga) nie sposób przewidzieć ich przebiegu. Są dane jako całość. Mogą się dokonywać, ale zawczasu, bez wykonania nieskończonego obliczenia, nie ma dostępu do wiedzy o ich kolejnych cząstkowych wynikach. Są zatem nieskończone aktualnie. Owa aktualność powoduje, że bez użycia fizycznej maszyny wykonującej

---

[13] Regularną cykliczność można by potraktować jako pewien wariant zakończenia pracy.

[14] W warstwie matematycznej, w największym możliwym uproszczeniu, moglibyśmy powiedzieć, że maszyny Turinga potrafią generować tylko liczby obliczalne, a te są definiowane jako liczby, których wartości można wyznaczać z dowolną zadaną dokładnością; przysługuje im zatem nieskończoność potencjalna (zob. 2.1).

od razu (aktualnie) nieskończenie wiele kroków, pewnych problemów nie można fizycznie rozwiązać (zob. rozdz. 4).

3.2. Z powyższych uwag wynika, że możliwości maszyn Turinga, a wraz z nimi wszelkich komputerów cyfrowych, są ograniczone ze względu na *niemożność wykroczenia* poza ramy nieskończoności potencjalnej. Maszyny takie mogą zrobić tylko to, co da się osiągnąć sukcesywnie, dzięki potencjalnie coraz większym pamięciom, szybkościom, ciągom wykonywanych operacji etc…

Jeśli idąc dalej, uznamy, że *uniwersalna maszyna Turinga* stanowi jedyny efektywnie realizowalny model obliczeń[15], to musimy dać odpowiedź pozytywną na tytułowe pytanie pracy. TAK, informatykom musi wystarczyć *nieskończoność potencjalna* – przynajmniej wówczas, gdy rozważają realne możliwości coraz szybszych i bogatszych w zasoby maszyn (czemu odpowiada w modelu potencjalnie nieskończona taśma). Zgodnie z takim przekonaniem dla kolejnych generacji maszyn będzie osiągalne tylko to, co umożliwiają: potencjalnie (a nie aktualnie) nieskończona taśma maszyny Turinga oraz potencjalnie (a nie aktualnie) nieskończone ciągi obliczeń.

3.3. Przeciwko zarysowanej w poprzednim punkcie tezie można zgłosić następujący zarzut. Wprawdzie pojęcie *nieskończoności potencjalnej* pozwala uchwycić istotę tego, co dla maszyn Turinga jest *wykonalne*; nie pozwala jednak rozpoznać tego, co niewykonalne (a także przyczyn owej niemożności). Można powiedzieć nawet ostrzej: przy realnych próbach określania ograniczeń maszyn cyfrowych wydaje się ono bezużyteczne.

Dopiero pojęcie nieskończoności aktualnej daje możliwość skutecznego wglądu we wspomniane ograniczenia – wglądu, który prowadzi do zrozumienia, dlaczego bez fizycznej realizacji jakichś postaci *nieskończoności aktualnej* pewnych problemów nigdy nie uda się rozwiązać. Wyjaśnię to na dwa sposoby.

Sposób pierwszy nawiązuje do faktycznie stosowanej przez informatyków strategii przezwyciężania *problemów nieobliczalnych* (cyfrowo). Jak wiadomo, problemy takie nie mogą zostać rozwiązane za pomocą jednego uniwersalnego algorytmu, który miałby zastosowanie do ich wszelkich możliwych przypadków szczególnych (czyli: dla wszelkich możliwych danych wejściowych). Mimo to, tj. mimo pojętej ogólnie nierozwiązywalności, każdy nieobliczalny problem P mieści w sobie pewne podproblemy P_i, dla których istnieją wystarczająco efektywne algorytmy „lokalne” A_i[16]. Okazuje się jednak, że żaden *zupełny podział*

---

[15] Jest to w istocie pewne sformułowanie słynnej tezy Churcha-Turinga, która w odniesieniu do modelu uniwersalnej maszyny Turinga (i wszelkich równoważnych jej konstrukcji formalnych) głosi, że „funkcja jest efektywnie obliczalna wtedy i tylko wtedy, gdy jest obliczalna za pomocą uniwersalnej maszyny Turinga” (por. Harel 2000; Mycka, Olszewski 2015).

[16] Weźmy za przykład problem rozstrzygania, czy dowolne równanie diofantyczne ma choć jedno rozwiązanie w liczbach całkowitych. Jest to problem nieobliczalny. Mimo to, wśród wszel-

problemu P na warianty P_i nie może składać się ze skończonej liczby wariantów P_i. Gdyby tak było, wówczas istniałby zbiorczy algorytm A (będący nawet sumą algorytmów A_i), który rozwiązywałby problem P we wszystkich przypadkach szczególnych. To zaś jest wykluczone ze względu na udowodnioną matematycznie nieobliczalność P. Problemów P_i, a wraz z nimi algorytmów A_i, musi być zatem nieskończenie wiele – *aktualnie nieskończenie wiele*. I na tym właśnie polega przyczyna nieobliczalności oryginalnego zagadnienia.

Drugi sposób objaśniania roli nieskończoności aktualnej w rozpoznawaniu ograniczeń technik cyfrowych odwołuje się do pojęcia *liczby nieobliczalnej*. Otóż wiadomo, że maszyny Turinga potrafią rozwiązywać tylko takie problemy, których rozwiązania wyrażają się (przy pewnej metodzie kodowania) przez liczby obliczalne. Wielkościom tym, jak pamiętamy z punktu 2.1, przysługuje nieskończoność potencjalna. Od czasów Turinga wiemy jednak, że prócz nich istnieją *liczby nieobliczalne*, które muszą odpowiadać hipotetycznym kodom algorytmów rozwiązujących problemy nieobliczalne. Liczbom tym przysługuje jednak *nieskończoność aktualna*. I to ona właśnie powoduje, że dla pewnych problemów hipotetyczne turingowskie algorytmy nie istnieją w postaci skończonej.

Podsumowując: pojęcie nieskończoności aktualnej – czy to odniesione do liczb, czy do podziału problemów na podproblemy – jest informatykom niezbędne do tego, by odkrywać problemy nieobliczalne i uzasadniać ich *nierozwiązywalność*[17].

4. Wzmacnianie maszyn Turinga za pomocą nieskończoności

Jak kilkukrotnie już zostało stwierdzone, ograniczenia obliczeń cyfrowych są pochodną faktu, że opisujący je formalnie model uniwersalnej maszyny Turinga dopuszcza posługiwanie się tylko i wyłącznie nieskończonością potencjalną. W związku z tym nasuwa się taka oto myśl, że model ów można wzmocnić, wprowadzając doń elementy *nieskończone aktualnie*.

---

kich możliwych równań diofantycznych znajdują się takie ich typy, których rozwiązywalność daje się sprawdzać przy użyciu wyspecjalizowanych algorytmów lokalnych.

[17] Dodajmy w przypisie, że pojęcie nieskończoności aktualnej umożliwia jeszcze jedną, istotną dla informatyki teoretycznej, procedurę. Pozwala mianowicie stwierdzać, że pewne modele obliczeń są równoważne uniwersalnej maszynie Turinga. Aby to stwierdzić, trzeba potraktować taśmę maszyny jako nieskończoną aktualnie całość i udowodnić, poprzez odpowiednie całościowe przekształcenie, że stanowi ona odpowiednik pewnego nieskończonego obiektu w innym modelu. W ten sposób, na przykład, dowodzi się równoważności modelu maszyny wielotaśmowej i jednotaśmowej (por. Harel 2000).

Okazuje się, że informatyczna teoria dostarcza dobrych (choć teoretycznych) rozwiązań w tym zakresie. Są nimi pewne rozszerzenia turingowskiego modelu obliczeń, które dopuszczają bądź przetwarzanie sygnałów ciągłych (a nie tylko dyskretnych), bądź wykonywanie nieskończonej liczby operacji w skończonym czasie.

4.1. Rozszerzenia pierwszego typu określa się w informatyce mianem *analogowych*[18]. Polegają one na takim poszerzeniu definicji obliczeń, by można je było przeprowadzać na danych ciągłych, reprezentowanych matematycznie przez liczby rzeczywiste ze wskazanego przedziału – np. z przedziału [0,1] będącego rozszerzeniem dyskretnego zbioru {0,1}. Ujmując rzecz fizycznie, danym takim odpowiadają pewne ciągłe wielkości mierzalne, takie jak natężenia, napięcia i potencjały elektryczne. W ślad za poszerzeniem dziedziny możliwych danych idzie wskazanie pewnych operacji podstawowych (jak różniczkowanie czy całkowanie), które wystarczają do wyznaczania wartości wszelkich funkcji złożonych z pewnej klasy (np. z klasy rekurencyjnych funkcji rzeczywistych, zob. Mycka 2010). Pierwszy teoretyczny model technik analogowych-ciągłych zawdzięczamy C. Shannonowi (1941); skrótowy przegląd modeli bardziej współczesnych zawiera praca Mycki i Piekarz (2004).

Jeśli chodzi o obecną w teorii obliczeń analogowych *nieskończoność aktualną*, to przejawia się ona dwojako. Po pierwsze, założenie ciągłości danych powoduje, że trzeba dopuścić *nieskończenie małe różnice* między przetwarzanymi i odczytywanymi sygnałami. Bez takiej możliwości analogowość byłaby sprowadzalna do cyfrowości. Po drugie, wspomniane wyżej założenie zmusza do przyjęcia, iż maszyna analogowa jest w stanie rozróżniać, odbierać i przetwarzać wielkości nieobliczalne (wyjaśniałem wcześniej, że są one nieskończone aktualnie). Wynika to z faktu, że liczby nieobliczalne stanowią konieczny składnik każdego ciągłego przedziału liczb rzeczywistych (reprezentujących przetwarzane sygnały).

Dzięki „dopuszczonej do działania" nieskończoności aktualnej obliczenia analogowe wykazują *większą moc* niż cyfrowe. To znaczy, że pozwalają rozwiązywać problemy nieobliczalne dla maszyn Turinga, jak chociażby zagadnienie stopu. Mówi o tym zarówno intuicja (skoro potrafimy generować liczby nieobliczalne, to potrafimy znajdować rozwiązania problemów nieobliczalnych), jak i odpowiednie twierdzenia matematyczne (zob. Mycka, Piekarz 2004 oraz Pour-El 1974). Pozostaje jednak pytanie, czy rozwijające intuicję twierdzenia

---

[18] Choć trzeba przyznać, że jest to tylko jedno z kilku możliwych rozumień analogowości. W tym przypadku chodzi o utożsamienie obliczeń analogowych z ciągłymi (tj. dopuszczającymi operacje na ciągłych danych). Analogowość w innym znaczeniu polega na realizowaniu obliczeń matematycznych za pomocą ich fizycznych analogonów, czyli tzw. naturalnych procesów obliczeniowych (por. Kari, Rozenberg 2008; Stacewicz 2017).

opisują faktyczny potencjał możliwych do fizycznego skonstruowania układów obliczeniowych?

4.2. Innego rodzaju sposób wzmacniania obliczeń turingowskich można określić mianem *infinityzmu operacyjnego*. Chodzi o możliwość wykonywania nieskończenie wielu operacji w skończonym czasie, na co nie pozwala klasyczny model Turinga. Najprostszym bodaj przykładem modelu infinitystycznego w sensie operacyjnym jest przyspieszająca maszyna Turinga, która każdą kolejną operację wykonuje dwa razy szybciej niż poprzednią – dzięki temu w czasie równym dwukrotności czasu trwania pierwszej operacji jest w stanie „zmieścić" nieskończoną liczbę obliczeń[19].

Inna grupa modeli odwołuje się do *teorii fizycznych* (a nie tylko abstrakcyjnych koncepcji obliczania), które przewidują, że w pewnych układach, a ponadto z punktu widzenia szczególnego obserwatora, czas może płynąć nieskończenie szybko. Mówiąc dokładniej: gdy w układzie obserwatora zachodzi skończona liczba zdarzeń, w układzie obserwowanym zdarzeń tego typu zachodzi nieskończenie wiele[20]. Wskazuje się nawet konkretne obiekty kosmiczne (np. czarne dziury Kerra), w pobliżu których opisany efekt mógłby występować. Dla teorii obliczeń istotny jest fakt, że zapewniające ów efekt zjawiska fizyczne można by wykorzystywać jako naturalne, faktycznie gdzieś-istniejące, procesy obliczeniowe. Co najważniejsze jednak, owe fizycznie dostępne nieskończoności aktualne zapewniałyby (niejako z definicji) rozwiązywanie problemów nieobliczalnych – problemów, które od maszyn Turinga wymagają nieosiągalnej dla nich nieskończonej liczby operacji.

4.3. Podsumowując sygnalizowane wyżej metody wzmacniania obliczeń turingowskich, trzeba stwierdzić, że jakkolwiek polegają one na przezwyciężaniu nieskończoności potencjalnej w sensie *matematycznym*, to pozostają one – jako modele teoretyczne – na poziomie nieskończoności potencjalnej w sensie *fizycznym*. Mówiąc inaczej: proponowane metody wykorzystywania nieskończoności aktualnej są czysto teoretyczne, a więc *hipotetyczne*. Stałyby się przydatne wówczas, gdyby postulowana przez nie nieskończoność aktualna była bytem rzeczywistym, a nie potencjalnym.

W szczególności: silniejsze od cyfrowych techniki analogowe byłyby realizowalne wtedy, gdyby faktycznie istniały ciągłe wielkości fizyczne; zaś techniki infinitystyczne w sensie operacyjnym byłyby implementowalne wówczas, gdyby występowały w przyrodzie procesy nieskończone, których wynik da się poznać w skończonym czasie. Jak wiadomo jednak, *realny byt* wspomnianych wielkości i procesów stoi pod dużym znakiem zapytania.

---

[19] W sprawie innych modeli tego typu zob. Ord 2006.

[20] Wyjaśnia to teoria względności, stosowana w ramach modeli czasoprzestrzeni Malamenta-Hogartha (zob. Etesi, Nemeti 2002).

5. Nieskończoność w informatyce: konieczna czy zbędna?

Zgodnie z deklaracją daną na wstępie – że niniejszy tekst jest zaproszeniem do dyskusji – chciałbym go zakończyć, przywołując fikcyjne głosy trzech informatyków, reprezentujących trzy sposoby podejścia do zagadnienia nieskończoności w informatyce. Ich wypowiedzi należy potraktować jako wstępne *odpowiedzi* na pytanie tytułowe artykułu[21].

Głos pierwszy należy do *informatyka-praktyka*, który widzi sprawę następująco: „Informatyce *nie jest potrzebna* żadna, nawet najsłabsza, forma nieskończoności – ani potencjalna w sensie matematycznym, ani potencjalna w sensie fizycznym. Maszyny i programy *zawsze będą obiektami skończonymi*, które trzeba badać i konstruować na sposób inżynierski, bez angażowania matematyczno-filozoficznej kategorii nieskończoności".

Inny pogląd prezentuje drugi dyskutant, którego moglibyśmy nazwać *teoretykiem-realistą*: „Informatyka *potrzebuje* matematycznej teorii nieskończoności (potencjalnej i aktualnej), ponieważ teoria ta daje wgląd w *ograniczenia* informatycznych technik. W szczególności: teoria liczb nieobliczalnych wyjaśnia, na czym polegają ograniczenia technik cyfrowych, modelowanych za pomocą uniwersalnej maszyny Turinga".

Jeszcze bardziej pronieskończonościowe nastawienie reprezentuje *teoretyk-wizjoner*, który wyraża je następująco: „Oprócz matematycznych teorii nieskończoności informatyce *są potrzebne* – niejako na zapas – pewne teorie przyrodnicze, postulujące istnienie nieskończoności w świecie fizycznym. Być może bowiem w już niedalekiej przyszłości uda się *obliczeniowo wykorzystać* nieskończone wielkości i procesy fizyczne" (może warto podać, skąd jest ten cytat?).

Wypada zadeklarować na koniec, że jako autor przedstawionego tekstu staję po stronie teoretyka-realisty, który dodatkowo, z wielką ciekawością i uwagą, wsłuchuje się w głosy nieobawiających się nieskończoności wizjonerów.

Literatura cytowana

Cafe Aleph <http://blog.marciszewski.eu/>, akademicki blog dyskusyjny W. Marciszewskiego i P. Stacewicza.

Chaitin G., 1998, *The Limits of Mathematics*, Singapore, Spinger.

Denning P.J., 2005, *Is Computer Science Science?*, "Communications of the ACM" t. 48, nr 4, s. 27–31.

---

[21] Zachęcam także do internetowej dyskusji w blogu Cafe Aleph w ramach wpisu pt. *Nieskończoność potencjalna w informatyce* (<http://marciszewski.eu/?p=9239>, dostęp: 24.04.2017).

Deutsch D., 1985, *Quantum theory, the Church-Turing principle and the universal quantum computer*, "Proceedings of the Royal Society of London A" 400, s. 97–117, London.

Etesi G., Nemeti I., 2002, *Non-Turing computations via Malament-Hogarth space-times*, "International Journal of Theoretic Physics" nr 41, s. 341–370.

Fortuna Z., Macukow B., Wąsowski J., 2015, *Metody numeryczne*, Warszawa, Wyd. WNT.

Harel D., 2000, *Rzecz o istocie informatyki. Algorytmika*, Warszawa, Wyd. Naukowo-Techniczne.

Hogarth M., 1994, *Non-Turing computers and non-Turing computability*, "Philosophy of Science Association" vol. 1, s. 126–138.

Kari L., Rozenberg G., 2008, *The many facets of natural computing*, „Communications of the ACM" nr 10(51), s. 72–83.

Knuth D.E., 1974, *Computer Science and its Relation to Mathematics*, "American Mathematical Monthly" nr 4(81), s. 323–343.

Marciszewski W., Stacewicz P., 2011, *Umysł – Komputer – Świat. O zagadce umysłu z informatycznego punktu widzenia*, Warszawa, Akademicka Oficyna Wydawnicza EXIT.

Michalewicz Z., 1992, *Genetic Alghorithms + Data Structures = Evolution Programs*, Berlin, Springer Verlag.

Murawski R., 2014a, *Nieskończoność w matematyce. Zmagania z potrzebnym, acz kłopotliwym pojęciem*, „Zagadnienia Filozoficzne w Nauce" nr 55, s. 5–42.

Murawski R. (red.), 2014b, *Filozofia informatyki. Antologia*, Poznań, Wyd. Naukowe UAM.

Mycka J., 2010, *Obliczenia dyskretne i ciągłe jako realizacje antropomorficznej i fizycznej koncepcji efektywnej obliczalności*, [w:] *Światy matematyki. Tworzenie czy odkrywanie*, red. I. Bondecka-Krzykowska, J. Pogonowski, s. 247–260, Poznań, Wyd. Naukowe UAM.

Mycka J., Olszewski A., 2015, *Czy teza Churcha ma jeszcze jakieś znaczenie dla informatyki?*, [w:] *Informatyka a filozofia. Od informatyki i jej zastosowań do światopoglądu informatycznego*, red. P. Stacewicz, Warszawa, Oficyna Wydawnicza PW.

Mycka J.M., Piekarz M., 2004, *Przegląd zagadnień obliczalności analogowej*, [w:] *Algorytmy, metody i programy naukowe*, red. S. Grzegórski, M. Miłosz, P. Muryjas, Lublin, Polskie Towarzystwo Informatyczne, s. 125–132.

Ord T., 2006, *The many forms of hypercomputation*, "Applied Mathematics and Computation" nr 178, s. 143–153.

Pour-El M.B., 1974, *Abstract Computability and its Relations to the General Purpose Analog Computer*, "Transactions of the American Mathematical Society" nr 199, s. 1–28.

Rozenberg G., Back T., Kok J.N., (red.), 2012, *Handbook of Natural Computing*, Berlin–Heidelberg, Springer-Verlag.

Shannon C., 1941, *Mathematical Theory of the Differential Analyzer*, "Journal of Mathematical Physics MIT" nr 20, s. 337–354.

Shagrir O., 2004, *Super-tasks, Accelerating Turing Machines and Uncomputability*, "Theoretical Computer Science" nr 317, s. 105–114.

Stacewicz P., 2016, *O algorytmach i algorytmicznej dostępności wiedzy*, „Studia Metodologiczne” nr 36, s. 315–331.

Stacewicz P., 2017, „O różnych sposobach rozumienia analogowości w informatyce”, artykuł złożony do publikacji w czasopiśmie „Semina Scientarum”.

Turing A.M., 1936, *On Computable Numbers, with an Application to the Entscheidungsproblem*, “Proceedings of the London Mathematical Society” nr 42, s. 230–265.

Wirth N., 1989, *Algorytmy + struktury danych = programy*, Warszawa, WNT.