O ISTOTNYCH OGRANICZENIACH METODY ALGORYTMICZNEJ

ALGORYTMY (*typy i własności*)

Algorytm ≈ program komputerowy

Różne typy algorytmów

- ✓ *cyfrowe vs analogowe*
- ✓ *determ. vs niedeterm.*
- ✓ *szeregowe vs równoległe*
- ✓ *szeregowe vs rekurencyjne*
- ✓ *klasyczne vs populacyjne*
- ✓ *działania vs uczenia się (…)*

Różne informatyczne własności algorytmów

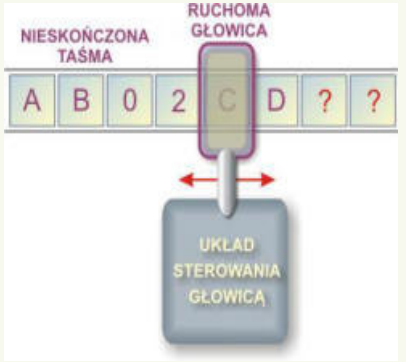
- ✓ ***złożoność czasowa***
- ✓ *złożoność pamięciowa*
- ✓ *złożoność struktury*
- ✓ *stabilność (numeryczna)*
- ✓ ***własność stopu***

Dwa pojęcia algorytmu (w informatyce)

W sensie wąskim

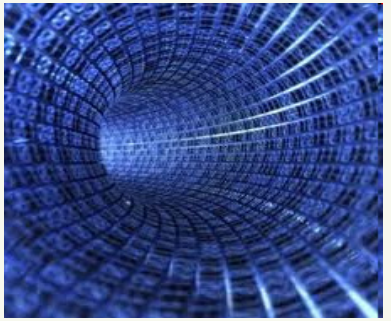
Algorytmem nazywa się każdy ogólny schemat procedury możliwej do wykonania przez **uniwersalną maszynę Turinga** (UMT).

(Ze względu na obliczeniową równoważność UMT i komputerów cyfrowych jest to pojęcie algorytmu dla maszyn cyfrowych.)



W sensie szerszym

Algorytmem nazywa się ogólny schemat procedury możliwej do wykonania przez **pewną maszynę** – niekoniecznie cyfrową i deterministyczną (np. analogową, kwantową, ewolucyjną – lista nie jest zamknięta).



Jak jest „zbudowana” maszyna Turinga ?

- **Maszyna Turinga** składa się z*:*

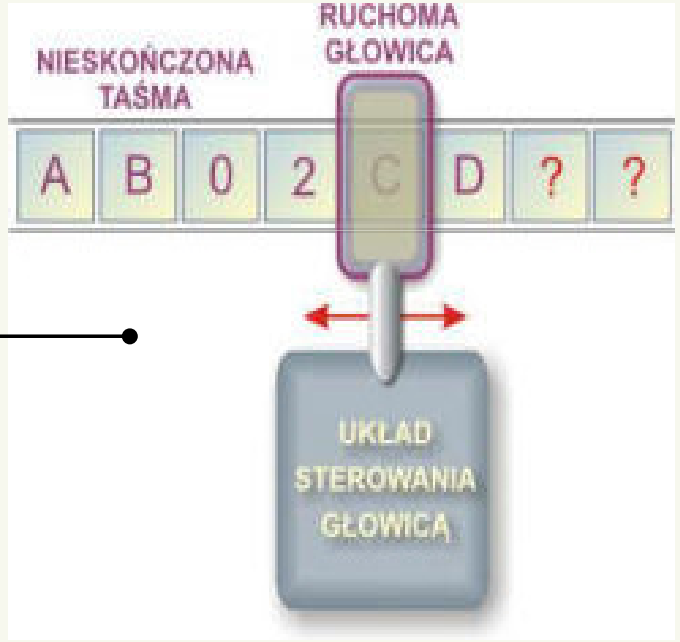
(1) nieskończonej, podzielonej na odrębne komórki, **taśmy**;

(2) **głowicy** do odczytu-zapisu danych;

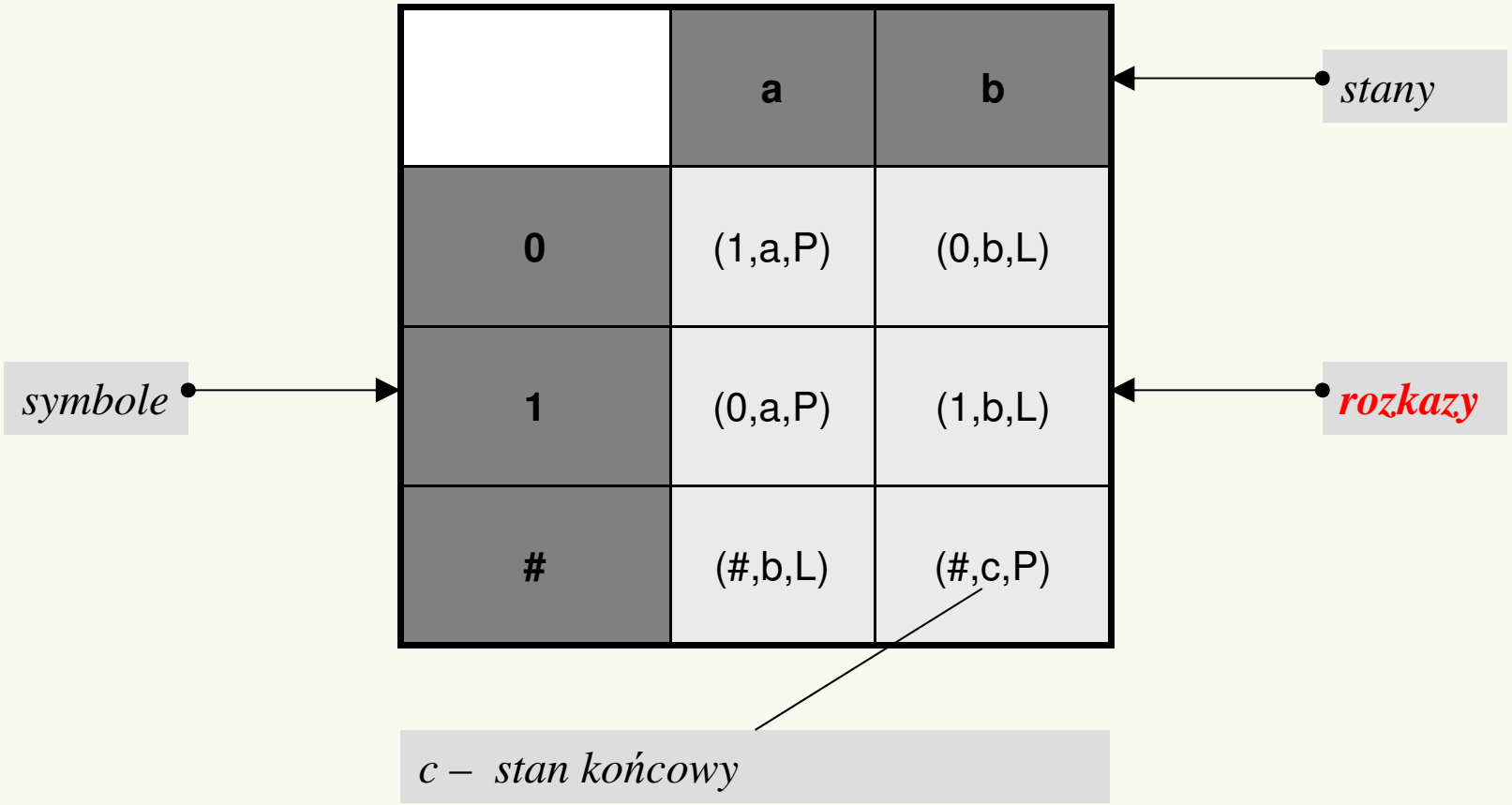
(3) **rejestru** stanów;

(4) **tablicy** przejść między stanami.

Automat ten działa na podstawie programu zawartego w tablicy (4).

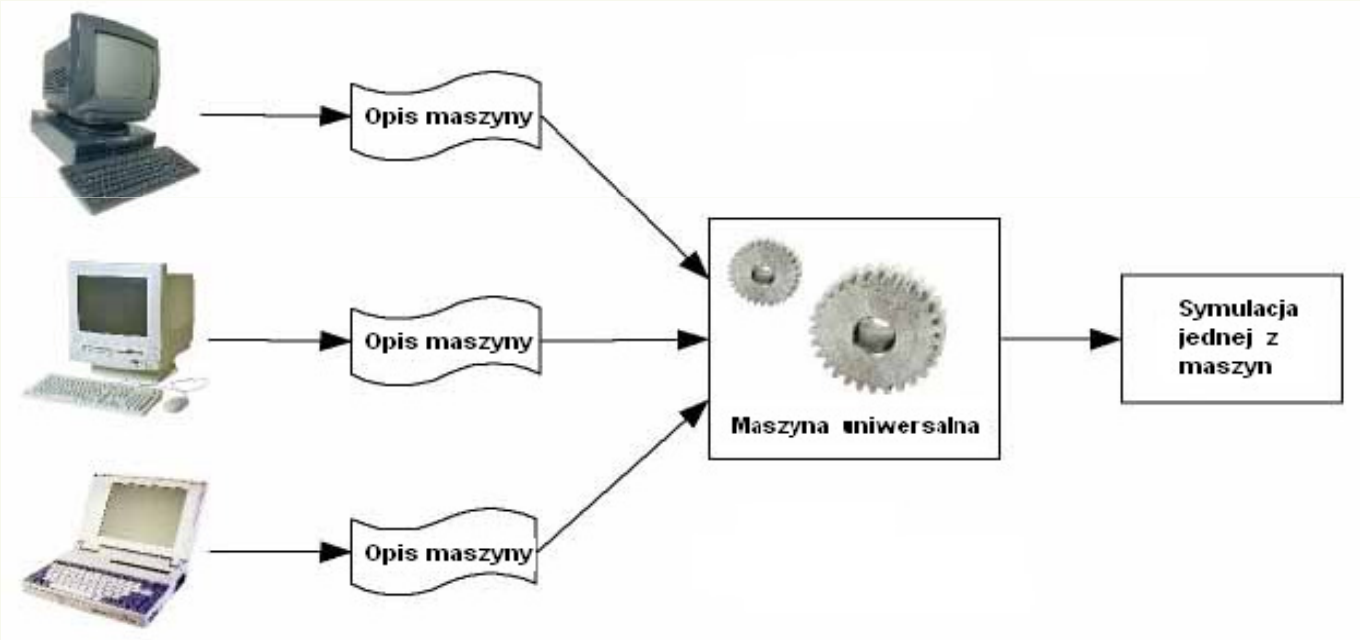


Jak wygląda program maszyny Turinga ?



Czym jest uniwersalna maszyna Turinga?

- **UMT** jest specjalną maszyną Turinga, której program ma za zadanie **symulować** działanie dowolnej, konkretnej MT.



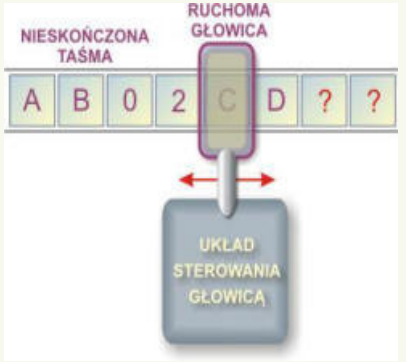
- Wykazano, że UMT może wykonać dowolnie złożony program dla dowolnie zaawansowanej technicznie **maszyny cyfrowej**.

Maszyny Turinga i algorytmy

A. w sensie wąskim

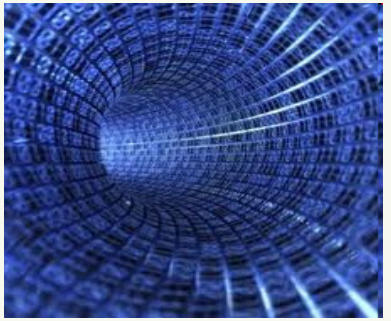
Algorytmem nazywa się każdy ogólny schemat procedury możliwej do wykonania przez **uniwersalną maszynę Turinga** (UMT).

(Ze względu na obliczeniową równoważność UMT i komputerów cyfrowych jest to pojęcie algorytmu dla maszyn cyfrowych.)



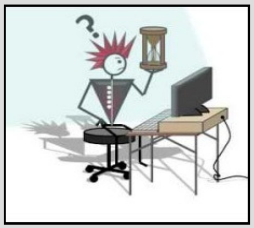
A. w sensie szerszym

Algorytmem nazywa się ogólny schemat procedury możliwej do wykonania przez **pewną maszynę** – niekoniecznie cyfrową i deterministyczną (np. analogową, kwantową, ewolucyjną – lista nie jest zamknięta).



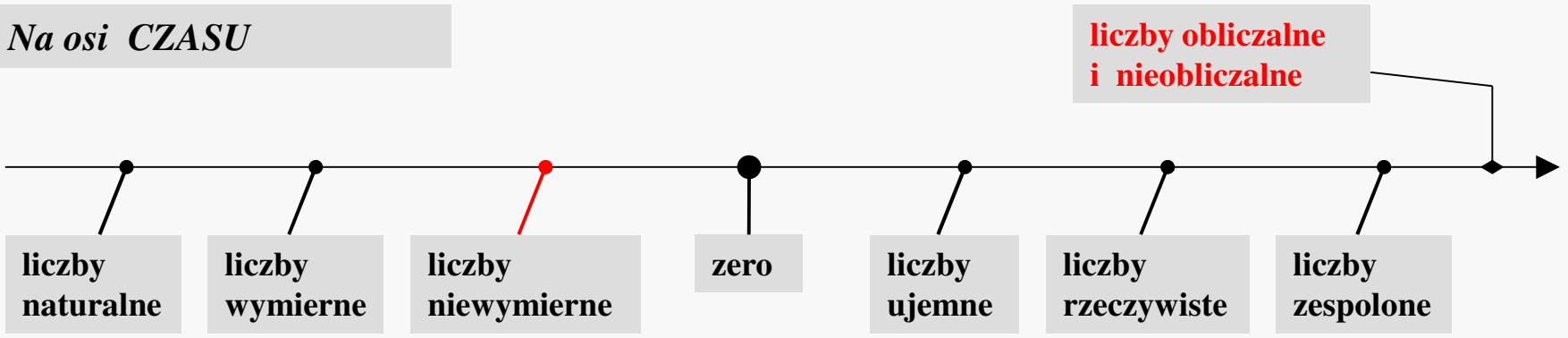
Zaskakujący wynik Turinga

- Istnieją **liczby**, których **nie można obliczyć**.
  (*algorytmicznie, za pomocą* **MT**)
- Istnieją **problemy** (ściśle zdefiniowane), których **nie można rozwiązać**.
  (*algorytmicznie, za pomocą* **MT**)

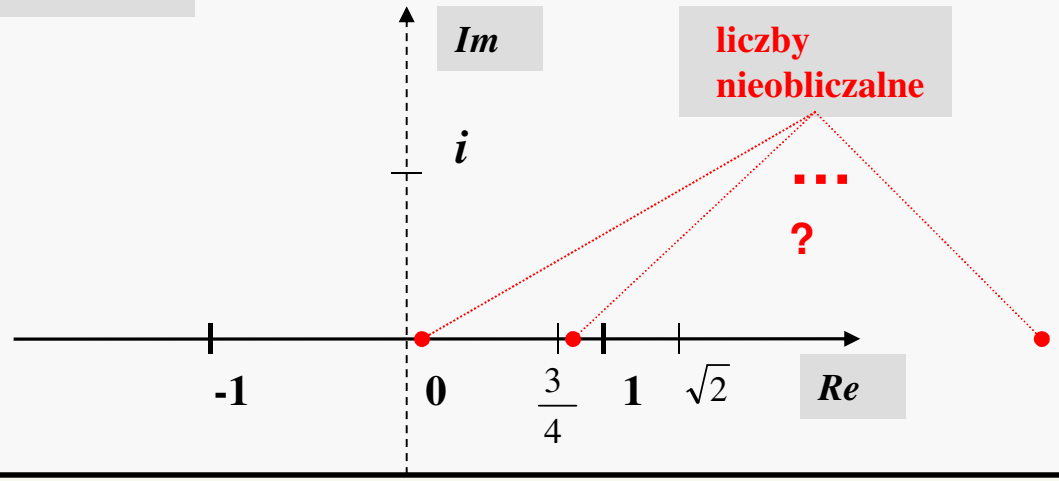


Różne klasy liczb

Na osi CZASU



Na osiach LICZBOWYCH



Liczby obliczalne i nieobliczalne

Liczba obliczalna jest to taka liczba rzeczywista, dla której istnieje **maszyna Turinga** (inaczej: program dla maszyny cyfrowej) pozwalająca obliczyć ją z dowolną zadaną dokładnością.

Liczba nieobliczalna nie ma powyższej własności: jest niewyznaczalna za pomocą maszyn Turinga.

$$\pi = 4 - \frac{4}{3} + \frac{4}{5} - \frac{4}{7} + \ldots = 4 \cdot \sum_{n=0}^{\infty} \frac{(-1)^n}{2n+1}$$

$$e = 1 + \frac{1}{1!} + \frac{1}{2!} + \frac{1}{3!} + \ldots = \sum_{n=0}^{\infty} \frac{1}{n!}$$

najszerzej znane obliczalne liczby niewymierne

Definiowanie liczb nieobliczalnych

Niektóre liczby nieobliczalne można **zdefiniować**, nie podając jednak efektywnego schematu wyznaczania ich kolejnych cyfr.

Poglądowy przykład:

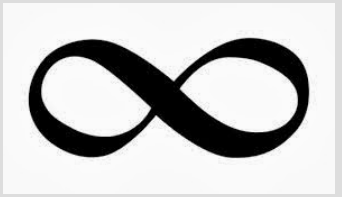
a) Tworzymy uporządkowaną listę maszyn Turinga MT_i (z danymi wejściowymi na taśmach).

b) Definiujemy liczbę zapisaną binarnie

L = $0,b_1b_2b_3b_4\ldots$ przy czym

$b_i = 1$, gdy MT_i kończy pracę

$b_i = 0$, gdy MT_i działa w nieskończoność



Na czym polegają ograniczenia metody algorytmicznej ?

Na czym polegają ograniczenia metody algorytmicznej ?

Polegają na tym, że **nie wszystkie problemy** mogą zostać rozwiązane za pomocą określonego typu algorytmów (np. cyfrowych).

O tym, że nie mogą, decydują dwie krytyczne własności algorytmów:

a) **złożoność czasowa** (zbyt wysoka)

b) **własność stopu** (gdy nie zachodzi)

Innymi słowy: **istnieją problemy nieobliczalne!**

Problemy nieobliczalne

Problemy algorytmicznie nierozwiązywalne nazywa się **nieobliczalnymi**.

Problemy te dzielą się na:

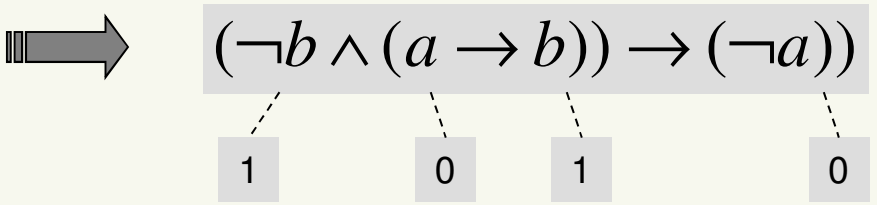
a) nierozwiązywalne **praktycznie** – gdy dla danego problemu nie istnieje algorytm o dostatecznie niskiej złożoności czasowej,

b) nierozwiązywalne **zasadniczo** – gdy nie istnieje algorytm rozwiązujący wszystkie przypadki szczególne danego problemu.

Problemy o złożoności wykładniczej

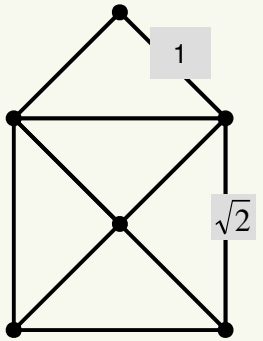
Problem spełnialności (logika)

- *Czy istnieje takie wartościowanie zmiennych zdaniowych, przy którym formuła zawierająca n zmiennych jest prawdziwa ?*



Problem komiwojażera (grafy)

- *W danym grafie z określonymi wagami krawędzi znajdź ścieżkę zawierającą wszystkie wierzchołki, która ma najmniejszą sumaryczną wagę (ścieżkę najkrótszą).*



Problemy o złożoności silniowej

Złożoność czasową n! mają:

- algorytmy sprawdzające wszystkie permutacje danych wejściowych o rozmiarze n.

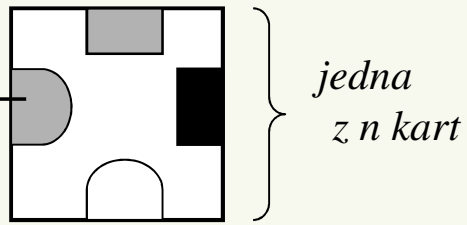
n! to liczba permutacji zbioru n-elementowego.

Przykład 1 (szyfr)

- *Odgadnij szyfr n-znakowy złożony z n różnych znaków nie powtarzających się.*

Przykład 2 (układanka)

- *Czy dla danych n kart istnieje złożony z nich kwadrat, w którym wszystkie karty stykają się odpowiednimi bokami ?*



Jaki problem nieobliczalny (zasadniczo) uznaje się za kanoniczny?

Turingowski problem stopu

Problem stopu (maszyny Turinga)

✓ *Dla dowolnej maszyny* MT_i *i jej dowolnych danych wejściowych* D_j

odpowiedz jednoznacznie,

czy MT_i *zatrzyma się dla danych* D_j
tj. zakończy przetwarzanie danych D_j*?*

Mniej technicznie:

Czy istnieje taki uniwersalny algorytm, który analizując zapis każdego innego algorytmu oraz dowolnych jego danych,
rozstrzygnie jednoznacznie
czy analizowany algorytm zakończy przetwarzanie swoich danych, czy też będzie je przetwarzał w nieskończoność?

Inne pr. nieobliczalne zasadniczo

Czy dane równanie diofantyczne, z dowolną liczbą niewiadomych i całkowitymi współczynnikami, ma choć jedno rozwiązanie w zbiorze liczb całkowitych ?

$$x^2 + 2y^3 - 4y^2 + z^4 = 0$$

Czy dane dwa języki sztuczne, z określonymi regułami budowania słów, pozwalają zbudować, zgodnie ze swoimi regułami, to samo (dane z góry i dowolne) słowo ?

abbbcaabbbbababc

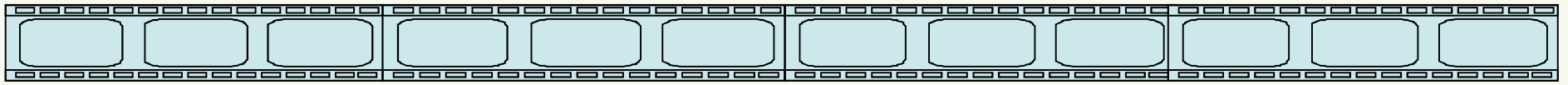
Problemy nieobliczalne - pytania

Istnienie problemów nieobliczalnych (PNB) prowadzi do następujących **pytań**:

a) Czy dla potrzeb realnych zastosowań nie wystarczą nam rozwiązania **problemów podobnych**, lecz obliczalnych?

b) Czy z każdego z problemu PNB nie daje się wydzielić takich **podproblemów**, dla których istnieją efektywne algorytmy „lokalne”?

c) Czy na gruncie **innych modeli obliczeń** niż cyfrowy (turingowski) problemy PNB nie stają się rozwiązywalne?

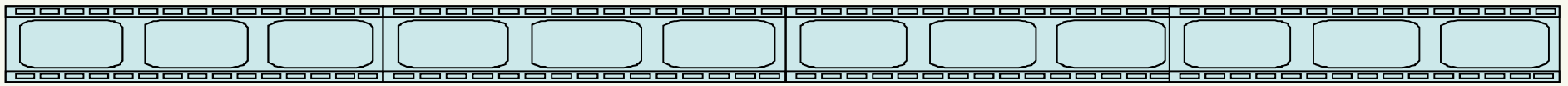


Algorytmy nieturingowskie

Algorytmy wykraczające poza model Turinga (UMT) sytuują się w sferze tzw. **hiperobliczeń**, do których należą:

- obliczenia **analogowe** (ciągłe)
- obliczenia kwantowe
- obliczenia infinitystyczne

Mimo istnienia teoretycznych modeli różnych typów hiperobliczeń, wciąż rozważa się pytania: a) o ich relację do modelu Turinga, b) o ich fizyczną realizowalność.



Algorytmy w nauce?

Cytat z Knutha:

Być może największym odkryciem będącym rezultatem wprowadzenia komputerów okaże się to, że algorytmom, jako przedmiotom badania, przysługuje niezwykłe bogactwo interesujących własności oraz to, że **algorytmiczny punkt widzenia** jest użytecznym sposobem organizacji wiedzy w ogólności.

Który z filozofów myślał o nauce algorytmicznie ?

Który z filozofów myślał o nauce algorytmicznie ?

G.W. Leibniz (1646-1716):

lingua characteristica
+
calculus ratiocinator

:

*„Należy znaleźć znaki lub **symbole** dla wyrażenia w sposób jasny i **ścisły** wszystkich myśli, jak w arytmetyce wyrażone są liczby lub w geometrii linie, aby można było z nimi czynić to samo, co czyni się w arytmetyce i geometrii, gdy ma się je jako przedmiot rozumowania.* (…)
*Z tego powodu wszystkie dociekania, które oparte są na **rozumowaniu**, dokonywane będą przez przemieszczanie tych znaków, przez pewien rodzaj **rachunku**”.*

Metoda algorytmiczna we współczesnej nauce

Metodologicznym odpowiednikiem (informatycznego) pojęcia algorytmu jest **metoda algorytmiczna** w nauce, która polega na:

a) **symboliczno-regułowym** zapisie wiedzy, w postaci dogodnej do automatycznych inferencji (współcześnie: wspomaganych komputerowo),

b) rozwiązywaniu właściwych danej nauce problemów poprzez konsekwentne stosowanie **reguł symbolicznych**,

c) zapisywaniu szczególnie efektywnych schematów w postaci możliwych do dalszego wykorzystywania **algorytmów**.

Polemiki i rozmówki w "Cafe Aleph"

Marciszewski i Stacewicz zapraszają do rozmów o światopoglądzie informatycznym

← Świato-ogląd i świato-pogląd

Jak można powiązać ze sobą różne znaczenia terminu „informacja"? →

Siła algorytmów?

Opublikowano 4 kwietnia 2015, autor: Paweł Stacewicz

Obecny wpis utworzyłem z intencją wywołania dyskusji wśród studentów PW (głównie informatyków i elektroników) na temat siły i ograniczeń algorytmów. Do rozmowy zapraszam oczywiście inne osoby, również stałych bywalców blogu. Zachęcam wstępnie do przeczytania krótkiego tekstu o algorytmach, który oświetli temat z szerszej perspektywy niż tylko programistyczna.

Oto link do tekstu: Algorytmy. Uwagi z pogranicza informatyki i metodologii nauk.

Zacznijmy od oczywistego faktu, że algorytmami zajmują się przede wszystkim informatycy i to oni zapisują je najbardziej precyzyjnie (co wymusza niejako specyfika języków programowania oraz przyszła implementacja na określonego typu maszynach).