Obliczalność
i
nieobliczalność

W odniesieniu do turingowskiego modelu obliczeń

Informacja, dane i liczby

Informacja > Dane > Liczby

- Informacja to coś więcej niż **dane**, bo dane stanowią pewną tylko, to znaczy komputerowo dostępną, formę informacji.

- Dane to coś więcej niż **liczby**, bo jakkolwiek dane koduje się liczbowo, to są one dodatkowo związane pewnymi informatycznymi strukturami.

Ale mimo wszystko: na poziomie najprostszego matematycznego opisu komputerowym danym odpowiadają jakieś liczby, a operacjom na danych jakieś **obliczenia**.

Co to znaczy „obliczać” (to compute)?

Co to znaczy „obliczać" (to compute)?

- wykonywać operacje na **liczbach**… ?
  (komputer = maszyna licząca)
- wyznaczać wartości pewnych **funkcji**…?
  (program = funkcja)
- przetwarzać **dane** (kody) za pomocą określonych zestawów reguł…?
  (dane > liczby, program > funkcja)
- realizować operacje zgodne z pewnym **modelem obliczeń**…?
  (komputer $\equiv_{\text{(np.)}}$ uniwersalna maszyna Turinga)

A zatem: co to znaczy „obliczać”?
- w sensie informatycznym -

- **OBLICZAĆ = PRZETWARZAĆ DANE**
  - *przy założeniu jednak, że w matematycznej teorii danym odpowiadają pewne **liczby**, a programom – **funkcje**.*

Model obliczeń cyfrowych

Najbardziej powszechny (być może: jedyny możliwy) model obliczeń wyznacza pojęcie **maszyny Turinga**.

Skończone realizacje programów dla **komputerów cyfrowych** są realnymi odpowiednikami obliczeń opisywanych w tym modelu.

Jest to model obliczeń **cyfrowych** (dyskretnych) i deterministycznych.

 $\approx$ 

Co to znaczy „obliczać cyfrowo?

- **OBLICZAĆ = PRZETWARZAĆ DANE**
  - *przy założeniu jednak, że w matematycznej teorii danym odpowiadają pewne **liczby**, a programom – **funkcje**.*

♦ **W cyfrowym modelu obliczeń, zdefiniowanym za pomocą pojęcia uniwersalnej maszyny Turinga:**

- *danym odpowiadają **liczby obliczalne** (w sensie Turinga),*
- *obliczeniom zaś **funkcje rekurencyjne**, czyli obliczalne w sensie Turinga.*

Jak jest „zbudowana” maszyna Turinga ?

- **Maszyna Turinga** składa się z*:*

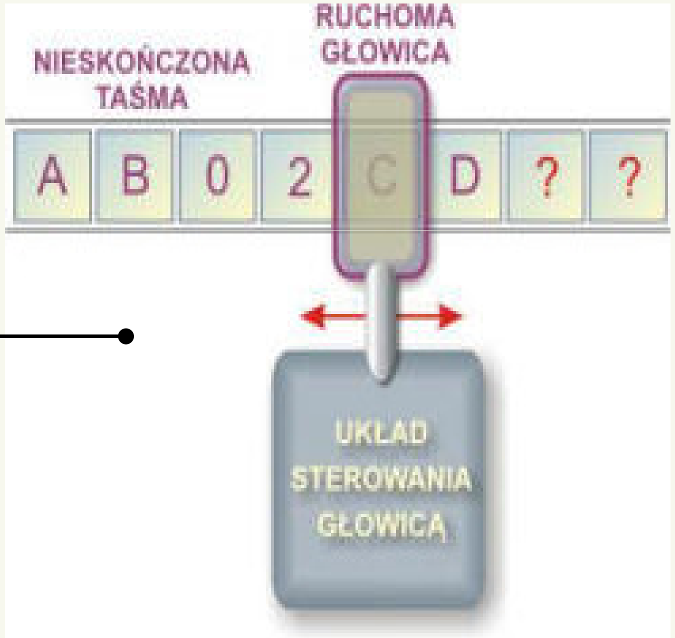
(1) nieskończonej, podzielonej na odrębne komórki, **taśmy**;

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

(3) **rejestru** stanów;

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

Maszyna „działa” na podstawie programu zawartego w tablicy (4).



Jak wygląda program maszyny Turinga?

	a	b
0	(1,a,P)	(0,b,L)
1	(0,a,P)	(1,b,L)
#	(#,b,L)	(#,c,P)

stany

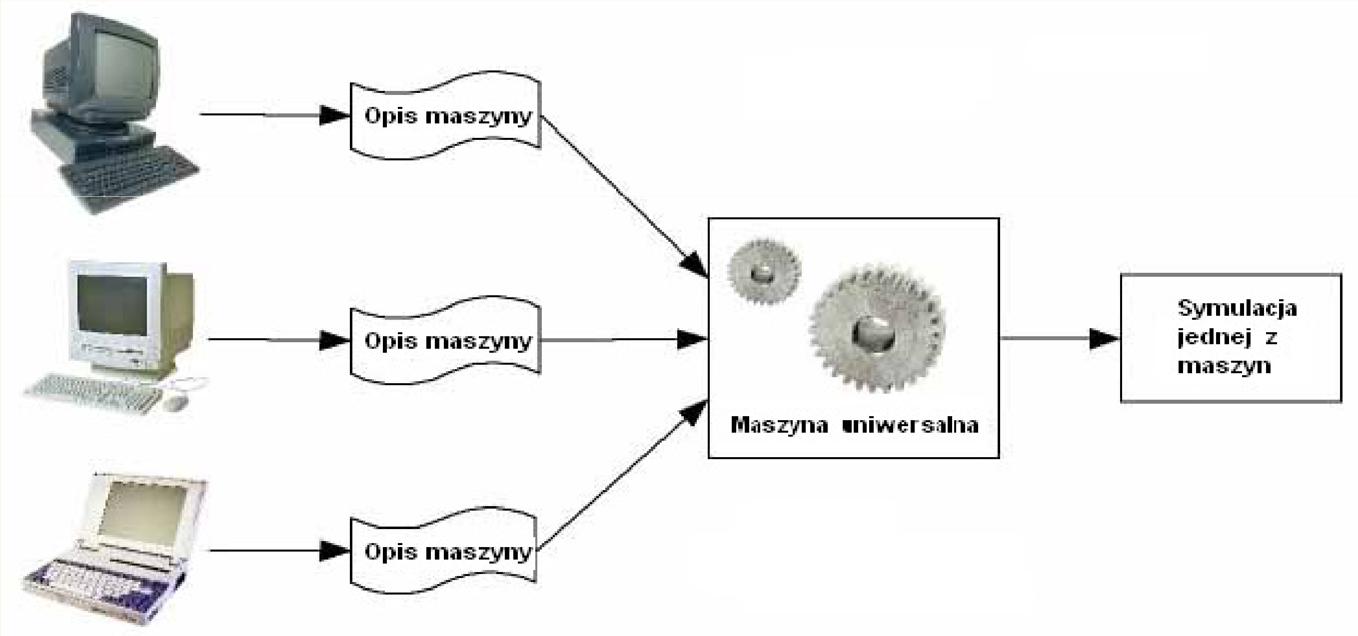
symbole

rozkazy

c – stan końcowy

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**.

Na czym polegają ograniczenia modelu turingowskiego?

Na czym polegają ograniczenia modelu turingowskiego?

Polegają na tym, że **nie wszystkie problemy** mogą zostać rozwiązane za pomocą operacji określonych w tym modelu

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 **nieobliczalne** (w modelu Turinga) dzielą się na:

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

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

Problemy o złożoności wykładniczej

Problem spełnialności (logika)

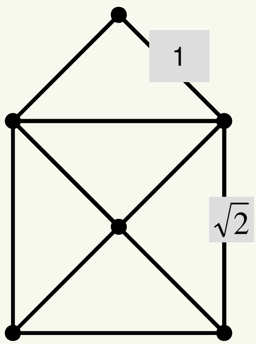
- *Czy istnieje takie wartościowanie zmiennych zdaniowych, przy którym formuła zawierająca n zmiennych jest prawdziwa ?*

$$(\neg b \wedge (a \rightarrow b)) \rightarrow (\neg a))$$

1 0 1 0

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.

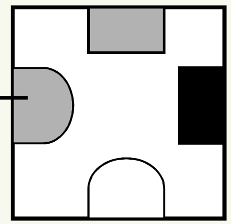
Przykład 1 (szyfr)

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

$$\underbrace{\boxed{3}\,\boxed{2}\,\boxed{4}\,\boxed{5}\,\boxed{1}}_{n\ \text{znaków}}$$

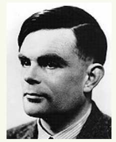
Przykład 2 (układanka)

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



jedna z n kart

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 ?

abbbcaabbbababc

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?

Obliczenia 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.

Typy liczb a modele obliczeń

Każdy model obliczeń określa dozwolony typ danych, któremu to typowi odpowiada z kolei pewien rodzaj liczb.

Krótko: każdy model odwołuje się do pewnego typu liczb.

- obliczenia **cyfrowe** → liczby **naturalne** (wielkości dyskretne)
- obliczenia **analogowe** → liczby **rzeczywiste** (wielkości ciągłe)
- obliczenia **kwantowe** →? liczby **zespolone** (q-bity, stany kwantowe)

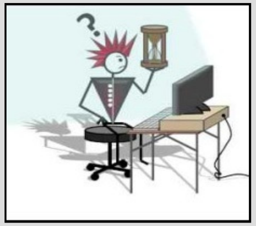
Zaskakujący wynik Turinga

- Istnieją **liczby**, których **nie można obliczyć**.

  (*algorytmicznie, za pomocą* **MT**)

- Istnieją **problemy** (ściśle zdefiniowane), których **nie można rozwiązać**.

  (*algorytmicznie, za pomocą* **MT**)



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.

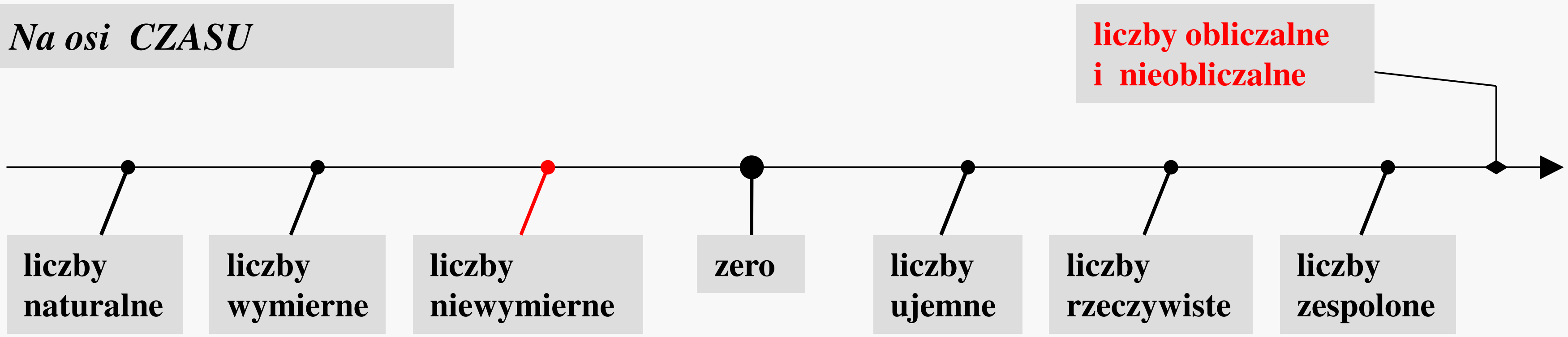
$$\pi = 4 - \frac{4}{3} + \frac{4}{5} - \frac{4}{7} + \ldots = 4 \cdot \sum_{n=0}^{\infty} \frac{(-1)^n}{2n+1}$$

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

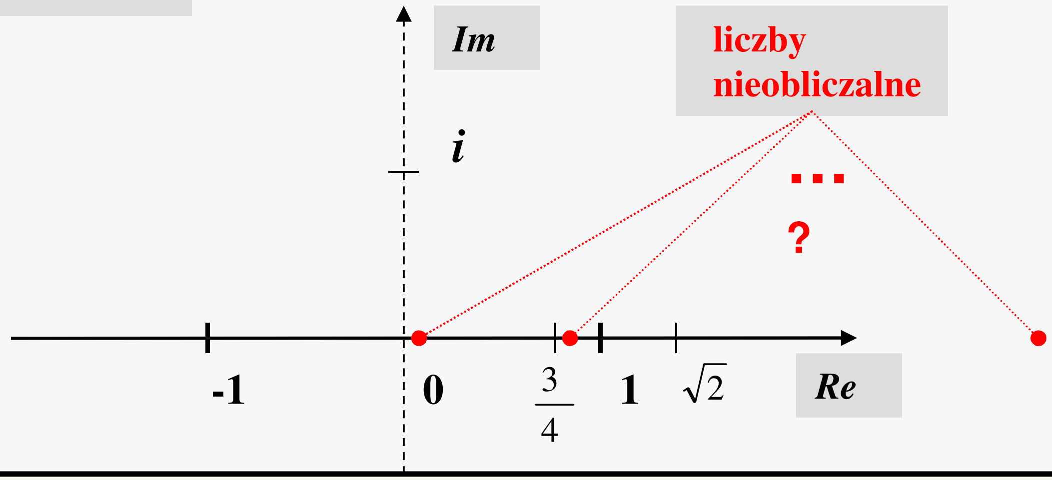
najszerzej znane obliczalne liczby niewymierne

Różne klasy liczb

Na osi CZASU



Na osiach LICZBOWYCH



Liczby nie(obliczalne) – *kilka określeń*

Liczby obliczalne…
(wg. Turinga)

- Istnieje klasyczny algorytm ich obliczania.
- Istnieje maszyna Turinga obliczająca je z dowolną zadaną dokładnością.
- Istnieje algorytm obliczania ich kolejnych cyfr.

Liczby nieobliczalne…
(wg. Turinga)

- Nie istnieje klasyczny algorytm ich obliczania.
- Żadna maszyna Turinga nie potrafi obliczyć ich z dowolną zadaną dokładnością.
- Nie istnieje algorytm obliczania ich kolejnych cyfr.

Definiowanie liczb nieobliczalnych

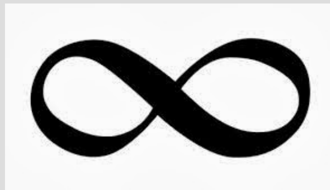
Niektóre liczby nieobliczalne można **zdefiniować**, nie podając jednak efektywnego schematu wyznaczania ich kolejnych cyfr.

Poglądowy przykład:

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

b) Definiujemy liczbę zapisaną binarnie

$\mathbf{L} = 0{,}b_1b_2b_3b_4\ldots$ przy czym



$$\begin{cases} b_i = 1, \text{ gdy } MT_i \text{ kończy pracę} \\ b_i = 0, \text{ gdy } MT_i \text{ działa w nieskończoność} \end{cases}$$