Struktura danych

- Sposób uporządkowania informacji w komputerze.

Algorytm

- Skończony, uporządkowany ciąg jasno zdefiniowanych czynności, koniecznych do wykonania pewnego zadania.
- Al-Khwarizmi – perski matematyk i astronom w 825 A.D. napisał traktat „O obliczeniach w hinduskich liczbach”; w 12 wieku przetłumaczono go na łacinę „Algoritmi de numero Indorum”; nazwisko autora omyłkowo wzięto za „metodę obliczeń”
- Algorytm – proces obliczeniowy zdefiniowany w ramach maszyny Turinga [Gurevich]

Maszyna Turinga

- Prosta koncepcyjnie. Zużycie zasobów podczas obliczenia dobrze modeluje rzeczywiste komputery.
- Potrafi przeprowadzić dowolne obliczenie możliwe na innych znanych nam maszynach (teza Churcha-Turinga 1936)
- Składa się z głowicy i nieskończonej taśmy podzielonej na pola. Zależnie od kombinacji stanu maszyny i pola maszyna zapisuje nową wartość w polu, zmienia stan, a następnie może przesunąć się o jedno pole w prawo lub w lewo.

Maszyna Turinga

- M = (Q,Σ,Γ,#,δ,q_0,F)
  - Q to skończony zbiór stanów maszyny.
  - Σ to skończony zbiór symboli wejściowych, które służą do reprezentacji wejścia maszyny i są zapisane na taśmie przed rozpoczęciem obliczenia.
  - Γ to skończony zbiór symboli taśmowych - są to wszystkie symbole jakie mogą się znajdować na taśmie. $\Sigma \subseteq \Gamma$.
  - # to symbol pusty - specjalny symbol taśmowy nie należący do symboli wejściowych
  - q_0 to stan początkowy

Maszyna Turinga

- M = (Q,Σ,Γ,#,δ,q_0,F)
  - δ to funkcja przejścia. Jest to funkcja częściowa prowadząca z Q$\times\Gamma$ w Q$\times\Gamma\times\{\leftarrow,\rightarrow\}$. Tak więc δ(A,x)=(B,y,K) oznacza, że maszyna będąca w stanie A i czytając symbol x z taśmy przejdzie do stanu B, zapisze symbol y na taśmie i przesunie głowicę o jedną komórkę w lewo (K = $\leftarrow$) lub w prawo (K = $\rightarrow$).
  - F to zbiór stanów akceptujących (końcowych) będący podzbiorem Q. Obliczenie maszyny kończy się kiedy osiągnięty zostanie któryś ze stanów F.

Dodawanie liczb unarnych (n+1)

$\Sigma=\Gamma=\{0,1\}$ Q={A,B,C,D} q_0=A F={D}

Dane wejściowe (3+2): #11110111#

- Tabelka
  - A,0: A,1,→
  - A,1: A,1,→
  - A,#: B,#,←
  - B,1: C,#,←
  - C,1: D,#,→
- Diagram

Złożoność czasowa

- Czas obliczenia maszyny M na konkretnym słowie wejściowym w - T(M,w) to liczba kroków wykonanych zanim maszyna się zatrzymała. Jeżeli M nigdy się nie zatrzyma, to mówimy, że czas obliczenia jest nieskończony i zapisujemy $T(M,w) = \infty$.
- Funkcja f : N→N jest złożonością czasową dla maszyny M, jeżeli

$$\forall\, n \in \mathbb{N} : f(n) = max\{T(M, w) : w \in \Sigma^n\}$$

Złożoność pamięciowa

- Pamięć potrzebną do obliczenia maszyny M na słowie w - S(M,w) definiujemy jako liczbę komórek taśmy, które zostały odwiedzone przez głowicę zanim maszyna się zatrzymała. Jeżeli M nie zatrzymuje się na w, to wartość S(M,w) jest nieokreślona.
- Funkcja f : N→N jest złożonością pamięciową dla maszyny M z własnością stopu, jeżeli

$$\forall\, n \in \mathbb{N} : f(n) = max\{S(M, w) : w \in \Sigma^n\}$$

Własność stopu

- Mówimy, że M ma własność stopu, jeżeli dla dowolnego słowa w $\in \Sigma^*$ obliczenie M na w się kończy.

Problem stopu (halting problem)

- Nie istnieje algorytm potrafiący określić dla dowolnej pary program-wejście czy program się zatrzyma.

Problem Collatza

$x := x_0$
repeat
  if x jest parzyste then
    x := x/2
  else
    x := 3*x + 1
until x = 1

Paradoks Epimenidesa (kłamcy)

ok. 600 B.C.:
„Kreteńczycy zawsze kłamią.”
(autor pochodził z Krety)
Czy Epimenides mówi prawdę? Da się dowieść, że zdanie jest fałszywe, więc nie jest do końca paradoksalne.

Inne wersje:
„Teraz kłamię.”
„To zdanie jest fałszywe.”

Twierdzenie Gödla (1931)

- Program Davida Hilberta (ok.1920) miał za zadanie sformalizowanie wszystkich teorii matematycznych za pomocą skończonego zbioru aksjomatów i udowodnienie, że te aksjomaty są spójne. Kurt Gödel udowodnił, iż jest to niemożliwe.
  - System formalny spójny (niesprzeczny) - system, w którym nie da się udowodnić jednocześnie pewnego zdania i jego zaprzeczenia.
  - System formalny zupełny - system, w którym możliwe jest przeprowadzenie dowodu dowolnego, prawidłowo zapisanego zdania tego systemu, lub jego zaprzeczenia.

Twierdzenie Gödla (1931)

Dowolny system formalny zawierający w sobie aksjomaty arytmetyki liczb naturalnych, jest albo zupełny albo spójny i nigdy nie posiada obu tych cech jednocześnie. Innymi słowy: można dowodzić prawdziwości wszystkich zdań takiego systemu, jednak wówczas istnieje w systemie pewne prawdziwe zdanie P, którego zaprzeczenie ~P również jest prawdziwe. Tym samym system albo jest sprzeczny wewnętrznie, albo system nie musi być sprzeczny, lecz wówczas istnieją zdania, których prawdziwości nie da się wywieść z aksjomatów i twierdzeń rozważanego systemu formalnego.

Problem stopu - dowód

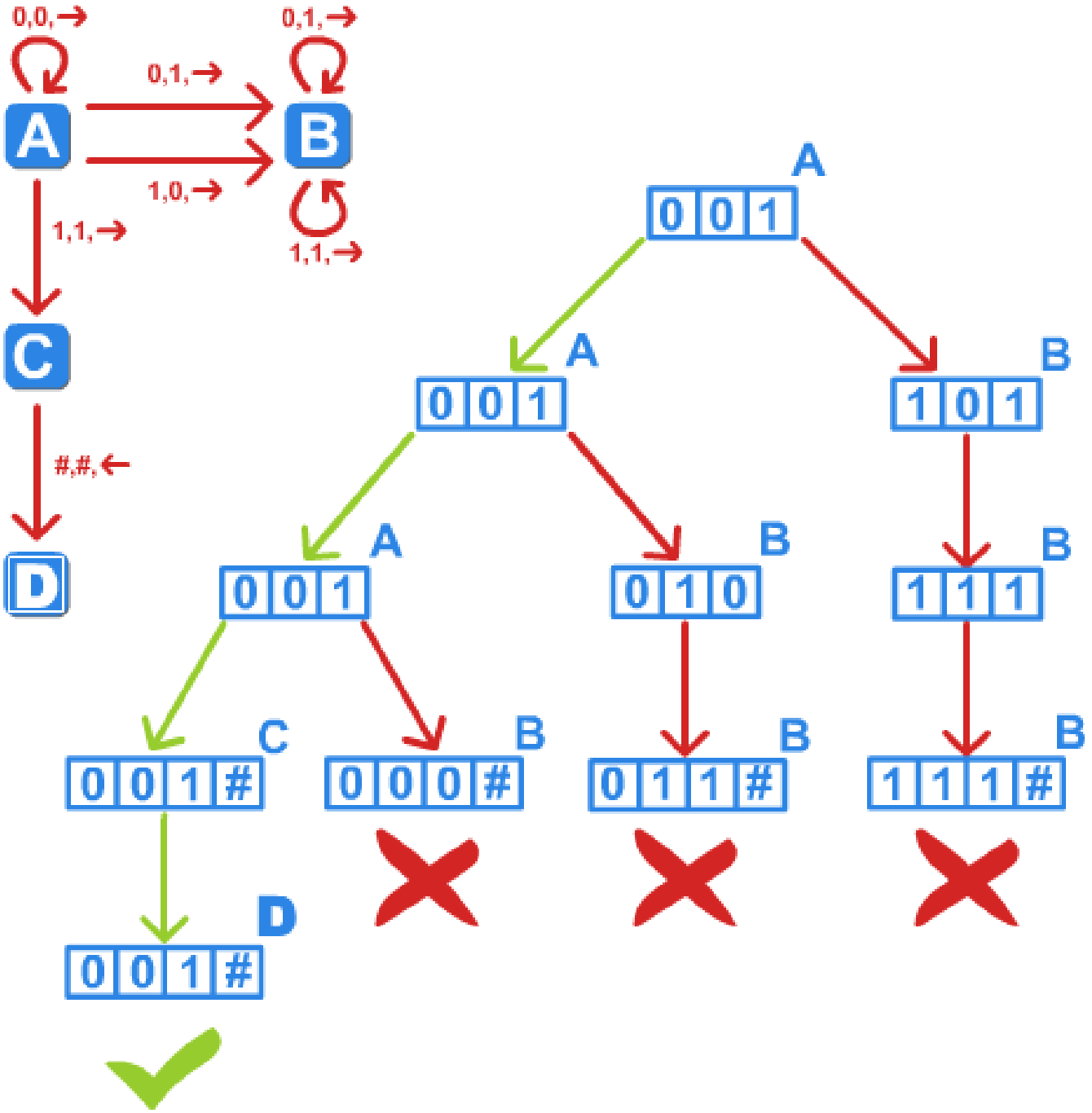
- Załóżmy, że istnieje program S, który dla dowolnego programu P i danych D:
  - zatrzymuje się i zwraca 1 jeżeli P zatrzymuje się na danych wejściowych D, oraz
  - zatrzymuje się i zwraca 0 w przeciwnym razie.
- program T dla dowolnego programu P zatrzymuje się wtedy i tylko wtedy, kiedy P zapętla się na swoim własnym kodzie podanym jako dane wejściowe
  - T(P):
  - if S(P,P)=1 then loop
  - else stop
- Czy T(T) się zatrzyma?

Wielotaśmowa maszyna Turinga

- Równoważna jednotaśmowej

Niedeterministyczna m. T.

- Mechanizm sterujący może przy konkretnym stanie i odczycie z taśmy postąpić na kilka różnych sposobów.
- Niedeterministyczna maszyna M akceptuje słowo w jeżeli istnieje taki ciąg przejść, który kończy się stanem akceptującym.
- Czas wykonania odpowiada wysokości drzewa możliwych wykonań.



źródło: http://wazniak.mimuw.edu.pl

Klasy problemów

- P – deterministycznie wielomianowy (*deterministic polynomial*) - problem decyzyjny, dla którego rozwiązanie można znaleźć w czasie wielomianowym.
- NP - niedeterministycznie wielomianowy (*nondeterministic polynomial*) - problem decyzyjny, dla którego rozwiązanie można zweryfikować w czasie wielomianowym. Inaczej mówiąc, problem jest w klasie NP, jeśli może być rozwiązany w wielomianowym czasie na niedeterministycznej maszynie Turinga.

Przykład

- Czy jakikolwiek podzbiór zadanego zbioru (np. {-2,6,-3,72,10,-11}) sumuje się do zera ?
- Sprawdzenie wszystkich podzbiorów ma złożoność wykładniczą (2^n).
- Przykładowe rozwiązanie (np. {-2,6,-3,10,-11}) możemy sprawdzic w czasie wielomianowym.

Relacje między klasami

- Oczywiste jest, że $P \subseteq NP$
- Czy P=NP? (pytanie warte milion dolarów dla Clay Mathematics Institute)

Problemy NP-trudne

- Jest to problem, taki że dowolny problem należący do NP może być do niego zredukowany w czasie wielomianowym.

Problemy NP-zupełne (NP-complete)

- Jest to problem, który należy do klasy NP oraz dowolny problem należący do NP może być do niego zredukowany w czasie wielomianowym (czyli problem NP-trudny należący do NP).
- Pierwszym problemem, którego NP-zupełność wykazano, był problem SAT, czyli problem spełnialności formuł zdaniowych. Udowodnił to w 1971 roku Stephen Cook.
- Inne problemy: komiwojażera, plecakowy.

Problemy NP-zupełne (NP-complete)

- Jeśli tylko potrafimy rozwiązać jakikolwiek problem NP-zupełny w czasie wielomianowym, to potrafimy rozwiązać w czasie wielomianowym wszystkie problemy NP. Wówczas P=NP.
- Aby udowodnić, że problem jest NP-zupełny wystarczy udowodnić, że należy do NP oraz zredukować do niego dowolny znany problem NP-zupełny.

Asymptotyczne tempo wzrostu

- Miara określającą zachowanie wartości funkcji wraz ze wzrostem jej argumentów. W teorii obliczeń, stosuje się ją w celu opisu złożoności obliczeniowej, czyli zależności ilości potrzebnych zasobów (np. czasu lub pamięci) od rozmiaru danych wejściowych algorytmu.
- Mówimy, że f jest co najwyżej rzędu g (zapis f(n)=O(g(n))), gdy istnieją takie stałe n_0>0, oraz c>0, że:

$$\forall_{n \geqslant n_0} \; f(n) \leqslant cg(n)$$

Asymptotyczne tempo wzrostu

- Mówimy, że f jest co najmniej rzędu g (zapis $f(n)=\Omega(g(n))$), gdy istnieją takie stałe $n_0>0$, oraz $c>0$, że:
- $$\forall_{n \geqslant n_0} \; f(n) \geqslant cg(n)$$
-
- Mówimy, że f jest dokładnie rzędu g (zapis $f(n)=\theta(g(n))$), gdy istnieją takie stałe $n_0>0$, oraz $c_1>0$, $c_2>0$, że:

$$\forall_{n \geqslant n_0} \; c_1 g(n) \leqslant f(n) \leqslant c_2 g(n)$$

Rzędy złożoności obliczeniowej

- 1 – stała
- $\log_2 n$ – logarytmiczna
- n – liniowa
- $n\log_2 n$ – liniowo-logarytmiczna (quasi-liniowa)
- n^2 – kwadratowa
- n^c – wielomianowa
- c^n - wykładnicza