

Wstęp do informatyki

Wykład 14

Trudność problemów
algorytmicznych/decyzyjnych

Instytut Informatyki UWr

Temat wykładu

- Rozstrzygalność, problemy nierozstrzygalne
- Klasy P i NP:
 - pojęcia
 - NP-zupełność
 - problem milenijny $P \stackrel{?}{=} NP$

Problemy decyzyjne

Problemy decyzyjne

Problem decyzyjny:

Zbiór słów L nad ustalonym alfabetem X , czyli
 $L \subseteq X^*$

S dla problemu decyzyjnego L :

Wejście:

słowo x nad ustalonym alfabetem X

Wynik:

tak – gdy $x \in L$

nie – gdy $x \notin L$

Problemy decyzyjne – przykład 0

Problem decyzyjny L

Zbiór palindromów nad alfabetem $\{0,1\}$.

Algorytm A dla problemu L

Wejście:

słowo x

Wynik:

tak – gdy x jest palindromem

nie – gdy x nie jest palindromem

Palindrom: słowo, które ma tę samą treść czytane od początku jak i od końca

Problemy decyzyjne – przykład 1

Problem decyzyjny L:

Zbiór grafów spójnych

Algorytm A dla problemu L

Wejście:

graf G

Wynik:

tak – gdy G jest spójny

nie – gdy G nie jest spójny

Problemy decyzyjne – reprezentacja

Problem decyzyjny L:

Zbiór grafów spójnych

Jak opisać graf jako słowo nad ustalonym alfabetem (przykład)?

- alfabet: $X = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9, :, (,)\}$
- wierzchołki grafu: liczby naturalne $1, 2, 3, \dots$
- opis grafu – liczba wierzchołków oraz ciąg krawędzi, np.

$10;(1;9)(2;7)(3;4)(2;6)$

Problemy decyzyjne – przykład 2

Problem decyzyjny L

Zbiór grafów spójnych

Algorytm A dla problemu L

Wejście:

słowo x nad alfabetem X

Wynik:

tak – gdy słowo x jest opisem grafu spójnego

nie – gdy słowo x NIE jest opisem grafu spójnego

Problemy decyzyjne – przykład 2

Zbiór grafów spójnych – działanie algorytmu A:

Wejście	Wynik
5;(1;2)(2;3)(3;4)(4;5)	tak
5;(1;3;2;4;5)	nie (niepoprawny opis)
5;(1;2)(1;3)(2;3)(4;5)	nie (niespójny)

(nie)rozstrzygalność

Problem rozstrzygalny

Definicja

Problem decyzyjny L nad alfabetem X nazywamy **rozstrzygalnym**, jeśli istnieje algorytm, który dla każdego słowa $x \in X^*$:

- kończy swoje działanie (**nie zapętla się!**)
- zwraca „**tak**” gdy $x \in L$
- zwraca „**nie**” gdy $x \notin L$

Problem L , który nie jest rozstrzygalny nazywamy **nierozstrzygalnym**.

Problem stopu - definicja

Ustalamy:

- język programowania ABC
- alfabet X używany do zapisu programów i danych w języku ABC

Oznaczenie:

- $p(d)$ – uruchomienie programu p na danych d

Problem

$$\text{STOP} = \{ (p, d) : p(d) \text{ zatrzymuje się (czyli nie zapętla się)} \}$$

Problem stopu – rozstrzygalność

Twierdzenie 1

Problem STOP jest nierozstrzygalny.

Dowód

Zakładamy nie wprost: istnieje program p_{STOP} , który rozwiązuje STOP, czyli spełnia specyfikację:

Wejście: p – program, d – dane

Wyjście:

tak – jeśli $p(d)$ zatrzymuje się

nie – jeśli $p(d)$ zapętla się

Problem stopu – rozstrzygalność

Dowód (c.d.)

Piszemy nowy program diag, który dla podanego na wejściu programu p działa następująco:

diag(p)

jeśli $p_{\text{STOP}}(p,p) = \text{„tak”}$: zapętlenie

jeśli $p_{\text{STOP}}(p,p) = \text{„nie”}$: zatrzymaj się

Problem stopu – rozstrzygalność

Dowód (c.d.)

Przeanalizujemy działanie $\text{diag}(\text{diag})$:

$\text{diag}(\text{diag})$

jeśli $p_{\text{STOP}}(\text{diag}, \text{diag}) = \text{„tak”}$: zapętlenie

jeśli $p_{\text{STOP}}(\text{diag}, \text{diag}) = \text{„nie”}$: zatrzymaj się

czyli:

- jeśli $\text{diag}(\text{diag})$ zatrzymuje się, to $\text{diag}(\text{diag})$ zapętla się
- jeśli $\text{diag}(\text{diag})$ zapętla się, to $\text{diag}(\text{diag})$ zatrzymuje się

SPRZECZNOŚĆ

Problem TESTER

Problem

TESTER = { p : p zatrzymuje się na każdym danych (czyli nigdy nie zapętla się) }

Inaczej mówiąc

Algorytm dla problemu TESTER (jeśli istnieje) umożliwia zweryfikowanie, czy Wasze programy mogą się (kiedykolwiek) zapętlić!

Problem TESTER

Problem

TESTER = { p : p zatrzymuje się na każdym danych (czyli nigdy nie zapętla się) }

Twierdzenie 2

Problem TESTER jest nierozstrzygalny.

Dowód

Założmy nie wprost, że **TESTER jest rozstrzygalny**. Pokażemy, że wynika z tego rozstrzygalność problemu STOP.

Problem TESTER

Dowód tw. 2 (cd)

Chcemy pokazać:

Istnieje p_{TESTER} rozstrzygający TESTER



STOP jest rozstrzygalny

Przypomnienie:

$\text{STOP} = \{ (p, d) : p(d) \text{ zatrzymuje się (czyli nie zapętla się)} \}$

Problem TESTER

Dowód tw. 2 (cd)

Nasz program dla STOP (wejście p , d):

1. Przekształć p w p' , który ignoruje dane wejściowe i zawsze, niezależnie od danych działa tak jak $p(d)$
2. Zwróć taką samą wartość jak $p_{\text{TESTER}}(p')$

Zatem:

- jeśli TESTER jest rozstrzygalny, to STOP jest rozstrzygalny;
- ale z tw. 1 wiemy, że STOP nie jest rozstrzygalny.

Nierozstrzygalność - wniosek

Wniosek 1

Nie istnieje program, który testuje (na zapętlenie) wszystkie programy.

Wniosek 2

Zawód testera oprogramowania (chyba) ma przyszłość...

Problemy łatwe i trudne?

Problem milenijny $P=?NP$

Nagrody za rozwiązanie

Drobna:

\$1 000 000

Duża:

sława i uznanie środowiska naukowego,
lukratywne oferty pracy, status naukowego
celebryty

Konsekwencja jeśli odpowiedź $P=NP$ (jedna z wielu):

podważenie wiarygodności systemów
kryptograficznych, na których bazuje np.
internetowy dostęp do usług bankowych

Klasa P („polynomial”)

Definicja

Problem algorytmiczny (decyzyjny) L należy do klasy **P** gdy istnieje wielomian w i algorytm A (rozstrzygający L), który dla każdego wejścia x :

- daje odpowiedź „tak” gdy $x \in L$
- daje odpowiedź „nie” gdy $x \notin L$
- kończy działanie po co najwyżej $w(n)$, krokach gdzie n to długość słowa x .

Dlaczego wielomian?

Pytanie

Która funkcja rośnie szybciej przy $n \rightarrow \infty$:

$$f(n) = n^{100} \quad [\text{wielomian}]$$

$$g(n) = 1,01^n \quad [\text{funkcja wykładnicza}]$$

Odp.: $g(n)$

Intuicja:

klasa P zawiera problemy, które można (dość) łatwo (czyli szybko) rozwiązać przy pomocy komputera.

„Wielomianowy czas”

Algorytm A działa w „wielomianowym czasie”, gdy istnieje wielomian w taki, że A dla każdych danych wejściowych x kończy działanie po co najwyżej $w(n)$ krokach, gdzie n to długość x .

Klasa P - przykłady

1. Spójność grafu
 2. Palindromy
 3. Czy w grafie jest ścieżka Eulera?
 4. Czy zadane drzewo binarne jest BST?
 5. Czy podany ciąg jest uporządkowany?
- ... jak i większość problemów rozważanych na tym przedmiocie
- ... ale na pewno problemy STOP i TESTER nie są w P (jak i inne problemy nierozstrzygalne)

Klasa NP („nondeterministic polynomial”)

Definicja (nieformalna)

Problem algorytmiczny L należy do **NP** gdy istnieje **wielomian** w i algorytm A , taki że:

- dla każdego $x \in L$ **istnieje** y takie, że A daje odpowiedź „tak” dla (x, y)
- dla każdego $x \notin L$ i **każdego** y , A daje odpowiedź „nie” dla (x, y)
- A zawsze kończy działanie po co najwyżej **$w(n)$** , krokach gdzie n to długość słowa x .

Intuicja: istnieje **krótki dowód** y , że $x \in L$, który może być szybko sprawdzony (przez A).

Klasa NP - przykłady

Niespójność grafu G opisanego słowem x – algorytm z „dowodem”:

- dowód y, że G nie jest spójny: podział grafu na dwie niepuste składowe spójności S_1 , S_2
- algorytm A: sprawdza, czy istnieją wierzchołki $s_1 \in S_1$ i $s_2 \in S_2$ połączone krawędzią.

Uwagi

- Z poprzedniego wykładu znacie algorytm rozwiązujący w wielomianowym czasie problem spójności bez „dowodu” y;
- Zatem problem (nie)spójności jest również w P (dowód w danych może być pusty)

Klasa NP - przykłady

Def. Ścieżka Hamiltona:

Ścieżka Hamiltona w grafie to ścieżka, która odwiedza KAŻDY wierzchołek grafu dokładnie jeden raz.

Ścieżka Hamiltona – algorytm z „dowodem”:

- dowód y : ciąg wierzchołków v_1, v_2, \dots, v_n . (graf ma n wierzchołków)
- algorytm A : dla każdego $i=1, 2, \dots, n-1$ sprawdza, czy (v_i, v_{i+1}) jest krawędzią w danym grafie wejściowym.

P vs NP

Twierdzenie

$$P \subseteq NP$$

Dowód

Jeśli problem L jest w P to istnieje algorytm A , który:

- daje wynik „tak” gdy $x \in L$
- daje wynik „nie” gdy $x \notin L$
- działa w wielomianowym czasie $w(n)$, gdzie n to długość wejścia x .

Algorytm A można traktować jako algorytm A' , który daje poprawną odpowiedź niezależnie od tego jaki „dowód” y zostanie dostarczony.

P vs NP

Pytanie (za milion dolarów)

Czy $P \neq NP$?

Czyli (nieprecyzyjnie):

- Czy istnieją problemy, dla których weryfikacja dowodu na odpowiedź „tak” jest istotnie łatwiejsza niż znalezienie takiego dowodu?

NP-zupełność

Problemy NP-zupełne:

- „najtrudniejsze” problemy w klasie NP

NP-zupełny problem L spełnia (m.in.) własności:

- $L \in \text{NP}$.
- jeśli $L \in \text{P}$, to $\text{P} = \text{NP}$.

Przykłady problemów NP-zupełnych:

- ścieżka Hamiltona
- spełnialność formuł rachunku zdań
- 3-kolorowalność grafu
- problem plecakowy (wersja decyzyjna)

NP-zupełność

Problem	$\in P$	$\in NP$	NP-zupełny?
Spójność	tak	tak	?
Ścieżka Hamiltona	?	tak	tak
3-kolorowalność	?	tak	tak
2-kolorowalność	tak	tak	?
Izomorfizm grafów	?	tak	?
Spełnialność – rachunek zdań	?	tak	tak
Spełnialność – rachunek kwantyfikatorów	?	?	
STOP	nie	nie	nie

NP-zupełność

Def. k-kolorowalność

Graf $G(V,E)$ jest k-kolorowalny wtw istnieje kolorowanie c wierzchołków G takie, że dla każdej krawędzi $(u,v) \in E$ kolor u jest inny niż kolor v .

Spełnialność rachunku zdań

We: formuła F rachunku zdań (bez kwantyfikatorów)

Wy:

tak – gdy istnieje wartościowanie zmiennych spełniające F

nie – w przeciwnym przypadku

...gdyby $P=NP$...

Problem	$\in P$	$\in NP$	NP-zupełny?
Spójność	tak	tak	tak
Ścieżka Hamiltona	tak	tak	tak
3-kolorowalność	tak	tak	tak
2-kolorowalność	tak	tak	tak
Izomorfizm grafów	tak	tak	tak
Spełnialność – rachunek zdań	tak	tak	tak
Spełnialność – rachunek kwantyfikatorów	?	?	

...gdyby $P \neq NP$...

Problem	$\in P$	$\in NP$	NP-zupełny?
Spójność	tak	tak	nie
Ścieżka Hamiltona	nie	tak	tak
3-kolorowalność	nie	tak	tak
2-kolorowalność	tak	tak	nie
Izomorfizm grafów	?	tak	?
Spełnialność – rachunek zdań	nie	tak	tak
Spełnialność – rachunek kwantyfikatorów	nie	?	

NP-zupełność a $P=?NP$

W co wierzymy?

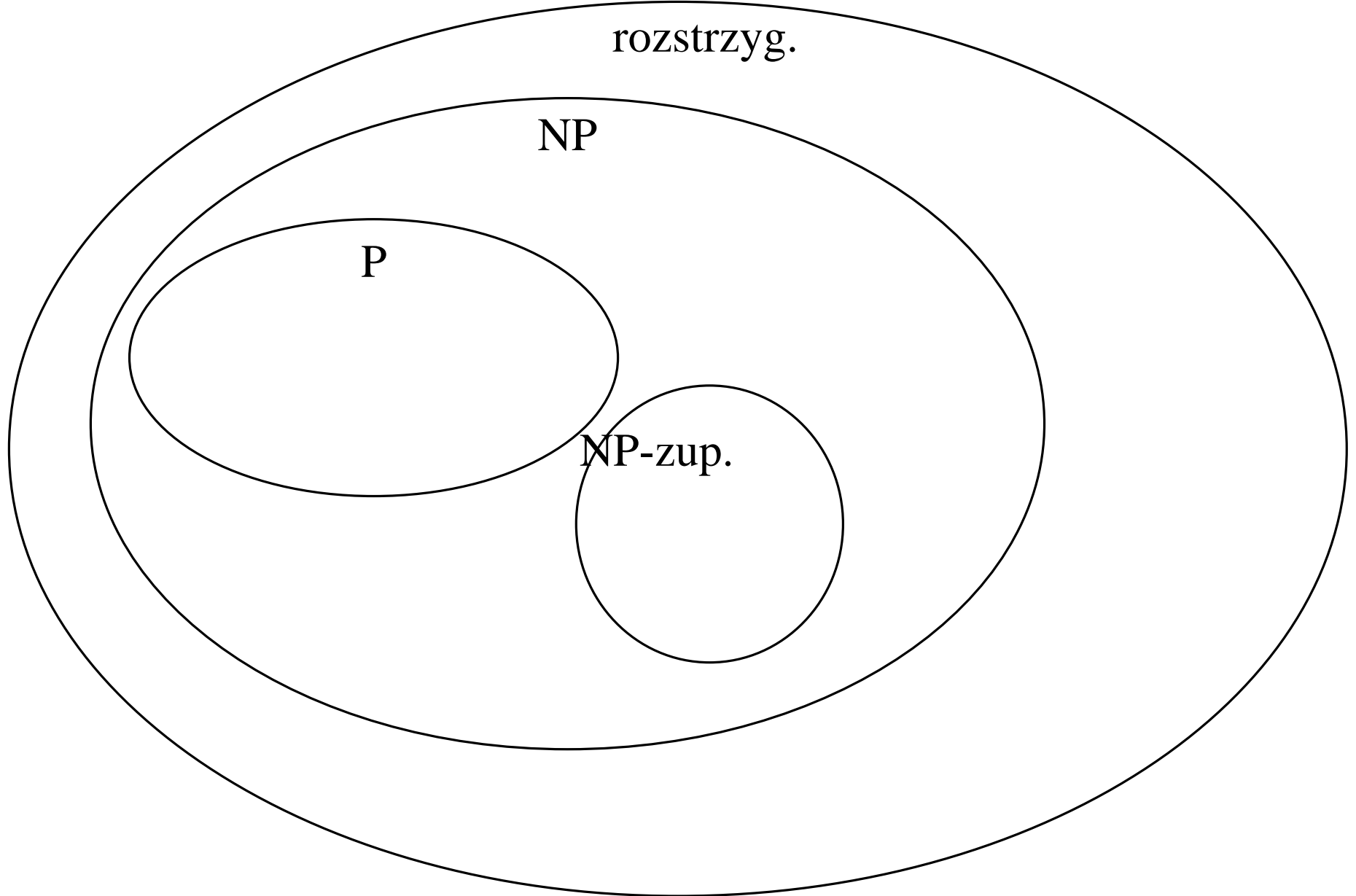
- $P=NP$ – nieliczni...
- $P\neq NP$ – większość ...

Prosty dowód na $P=NP$...?

- Algorytm wielomianowy (nie potrzebujący „dowodów” y) dla dowolnego NP-zupełnego problemu L .

Każdego roku publikowanych jest kilka(naście?) takich „dowodów” ... dotychczas wszystkie niepoprawne.

P, NP, NP-zupełność, rozstrzygalność



P, NP, NP-zupełność, rozstrzygalność

