Języki, automaty i obliczenia — ćwiczenia 14

$\mathrm{P} \stackrel{?}{=} \mathrm{NP}$

Wykład: klasy złożoności obliczeniowych:

- Maszyna deterministyczna działa w czasie/pamięci $f(n)$, gdy na każdym wejściu długości n terminuje się w czasie $\leqslant f(n)$. Oznaczamy to odpowiednio: $\mathrm{DTime}(f(n))$, $\mathrm{DSpace}(f(n))$.
- Maszyna niedeterministyczna działa w czasie/pamięci $f(n)$, jeśli na każdym wejściu długości n *każdy jej branch* kończy się w czasie $\leqslant f(n)$. Oznaczamy to odpowiednio: $\mathrm{NTime}(f(n))$, $\mathrm{NSpace}(f(n))$.
- Przykłady klas złożoności:
  - $\mathrm{P} = \mathrm{DTime}(\mathrm{poly}(n))$; tj. języki, które maszyna deterministyczna umie rozpoznać w czasie wielomianowym;
  - $\mathrm{NP} = \mathrm{NTime}(\mathrm{poly}(n))$; tj. języki, które maszyna niedeterministyczna umie rozpoznać w czasie wielomianowym.
- Inna definicja klasy NP: klasa języków, które umie rozpoznać maszyna deterministyczna w czasie wielomianowym, jeśli dostanie jako wejście na oddzielnej taśmie świadka, że słowo należy do języka.
  - Przykład: Hamiltonian Cycle (cykl Hamiltona w grafie). Umiemy łatwo w czasie wielomianowym *zweryfikować*, że podany świadek (tu: permutacja wszystkich wierzchołków) jest poprawnym rozwiązaniem.
- **Nie wiadomo**, czy $\mathrm{P} = \mathrm{NP}$; to jest jeden z problemów milenijnych (za $1,000,000).
- Dany problem decyzyjny D jest:
  - *NP-trudny*, jeśli każdy problem z klasy NP można zredukować **w czasie wielomianowym** do D;
  - *NP-zupełny*, jeśli należy do NP **oraz** jest NP-trudny.

  (Na przykład problem stopu jest NP-trudny; ale nie NP-zupełny, bo nie należy do NP.)
- Raczej wierzymy, że $P \neq NP$ i wtedy problemów *NP-trudnych* nie zrobimy wielomianowo.
- Przykładowe problemy NP-zupełne z wykładu:

> SAT (spełnialność formuł logicznych)
> Input: formuła logiczna ze zmiennymi $x_1, x_2, \ldots, x_n$
> Output: czy istnieje takie wartościowanie zmiennych, że formuła będzie spełniona?

> Input: niedeterministyczna maszyna Turinga $\mathcal{M}$, liczba n (zapisana **unarnie**)
> Output: czy $\mathcal{M}$ terminuje na ε w n krokach?

> Input: zbiór kolorów $\mathcal{C}$, zbiór kafelków $K \subseteq \mathcal{C}^4$, liczba k (**unarnie**)
> Output: czy istnieje kafelkowanie kwadratu $k \times k$ kafelkami z K?

Jest ich sporo, sporo więcej!

- Aby udowodnić **NP-zupełność** problemu D:
  - Udowadniamy, że $D \in \text{NP}$.
  - Wybieramy ulubiony problem NP-zupełny E i wskazujemy redukcję z E do D **w czasie wielomianowym**; *innymi słowy, chcemy pokazać, że każdą instancję problemu E możemy rozwiązać problemem D.*

1. Wykaż, że dwie definicje klasy NP (przez wielomianową maszynę niedeterministyczną oraz przez wielomianową maszynę deterministyczną ze świadkiem) są równoważne.

2. Niech relacja $\leqslant_P$ oznacza redukcję w czasie wielomianowym. Wykaż, że jeśli dla problemów A, B, C mamy $A \leqslant_P B$ oraz $B \leqslant_P C$, to $A \leqslant_P C$.

Wywnioskuj, że jeśli problem A jest NP-trudny oraz $A \leqslant_P B$, to problem B też jest NP-trudny.

3. Wykaż, że następujący problem jest NP-zupełny:

> 3-CNF-SAT
> INPUT: formuła logiczna ze zmiennymi $x_1, x_2, \ldots, x_n$ o szczególnej postaci: koniunkcja wyrażeń, z których każde jest alternatywą trzech termów. Na przykład:
>
> $$(x_1 \vee \neg x_2 \vee \neg x_3) \wedge (\neg x_1 \vee x_4 \vee \neg x_2) \wedge (x_1 \vee x_2 \vee x_3) \wedge (x_1 \vee x_2 \vee x_4).$$
>
> OUTPUT: czy istnieje takie wartościowanie zmiennych, że formuła będzie spełniona?

4. Wykaż, że następujący problem jest NP-zupełny:

> 3-COLORABILITY
> INPUT: graf nieskierowany G
> OUTPUT: czy istnieje kolorowanie wierzchołków na 3 kolory (tj. funkcja $V(G) \to \{1, 2, 3\}$) takie, że każda krawędź łączy wierzchołki różnych kolorów?

5. Wykaż, że następujący problem jest NP-zupełny:

> k-COLORABILITY
> INPUT: graf nieskierowany G, liczba k, $1 \leqslant k \leqslant |V(G)|$
> OUTPUT: czy istnieje kolorowanie wierzchołków na k kolorów (tj. funkcja $V(G) \to \{1, 2, \ldots, k\}$) takie, że każda krawędź łączy wierzchołki różnych kolorów?

6. Wykaż, że następujący problem jest NP-zupełny:

> CLIQUE
> INPUT: graf nieskierowany G, liczba k, $1 \leqslant k \leqslant |V(G)|$
> OUTPUT: czy graf zawiera klikę k-wierzchołkową?

7. Wykaż, że następujący problem jest NP-zupełny:

> VERTEX COVER
> INPUT: graf nieskierowany G, liczba k, $1 \leqslant k \leqslant |V(G)|$
> OUTPUT: czy można w G wyróżnić k wierzchołków tak, by każda krawędź miała przynajmniej jeden wyróżniony koniec?

Jeśli słowo „przynajmniej” zastąpimy słowem „dokładnie”, to czy problem wciąż będzie NP-zupełny?