Problemy trudne i zupełne, przykłady problemów o różnej złożoności. Problem *Czy P=NP*?

Funkcja $f\colon \Sigma^* \to \Sigma^*$ jest obliczalna w czasie wielomianowym, jeśli istnieje maszyna Turinga M działająca w czasie wielomianowym zatrzymująca się dla każdego $\mathrm{w} \in \Sigma^*$ i zwracająca $\mathrm{f(w)}$.

Język $A \subseteq \Sigma^*$ jest wielomianowo redukowalny do języka $B \subseteq \Sigma^*$, jeśli istnieje funkcja $f\colon \Sigma^* \to \Sigma^*$ obliczalna w czasie wielomianowym taka, że $\forall_{w \in \Sigma^*} w \in \mathrm{A} \Leftrightarrow f(w) \in \mathrm{B}$.

Funkcja $f\colon \Sigma^* \to \Sigma^*$ jest obliczalna w pamięci logarytmicznej, jeśli istnieje maszyna Turinga M używająca logarytmicznej liczby komórek taśm roboczych zatrzymująca się dla każdego $\mathrm{w} \in \Sigma^*$ i zwracająca $\mathrm{f(w)}$.

Język $A \subseteq \Sigma^*$ jest redukowalny do języka $B \subseteq \Sigma^*$ w pamięci logarytmicznej, jeśli istnieje funkcja $f\colon \Sigma^* \to \Sigma^*$ obliczalna w pamięci logarytmicznej taka, że $\forall_{w \in \Sigma^*} w \in \mathrm{A} \Leftrightarrow f(w) \in \mathrm{B}$.

Uwaga. Redukcja w pamięci logarytmicznej $\underset{\not\Leftarrow}{\Rightarrow}$ redukcja w czasie wielomianowym.

Niech C będzie klasą złożoności obliczeniowej (czasowej lub pamięciowej). Język A jest C-trudny, jeżeli dla dowolnego języka $B \in C$ istnieje efektywna redukcja (tzn. w czasie wielomianowym lub pamięci logarytmicznej) z B do A.

Niech C będzie klasą złożoności obliczeniowej (czasowej lub pamięciowej). Język A jest C-zupełny, jeśli:

(a) A jest C-trudny oraz
(b) $\mathrm{A} \in \mathrm{C}$.

- P (PTIME) – polynomial time

Klasę *P* tworzą wszystkie problemy decyzyjne, które w co najwyżej wielomianowym czasie rozwiązuje deterministyczna maszyna Turinga.

- NP (NPTIME) – Nondeterministic polynomial time

Klasa *NP* problemów decyzyjnych zawiera wszystkie problemy decyzyjne, które w co najwyżej wielomianowym czasie rozwiązuje niedeterministyczna maszyna Turinga.

Inna równoważna definicja mówi, że problem *NP* to taki, dla którego rozwiązanie można zweryfikować w czasie wielomianowym.

$$P \subseteq NP \subseteq PSPACE \subseteq EXPTIME \subseteq NEXPTIME$$

Przykłady problemów

1. $PATH = \{(\mathrm{G}, \mathrm{v}_1, \mathrm{v}_2)\text{: w grafie skierowanym G istnieje ścieżka z } \mathrm{v}_1 \text{do } \mathrm{v}_2\}$
   Problem NL-zupełny.

Formuła logiczna – wyrażenie złożone ze zmiennych logicznych połączonych ¬, ∧, ∨.
Formuła Φ jest spełniana wtw. gdy istnieje wartościowanie zmiennych takie, że Φ jest prawdziwa.

2. $SAT = \{\text{Formuła logiczna } \Phi\text{: } \Phi - \text{spełnialna}\}$
   Problem NP-zupełny.

Literał – zmienna lub jej zaprzeczenie, klauzula – alternatywa literałów. Formuła Φ jest w koniunktywnej postaci normalnej (CNF) jeżeli jest koniunkcją klauzul. Formuła Φ jest w postaci 3CNF jeżeli każda klauzula zawiera dokładnie 3 literały.

3. $3SAT = \{\text{Formuła logiczna } \Phi\text{: } \Phi - \text{spełnialna formuła w postaci 3CNF}\}$
   Problem NP-zupełny.

X – wielozbiór liczb naturalnych, np. X = {1, 2, 3, 3, 7, 9, 9, 11}.

4. $SUBSET\_SUM = \{(\mathrm{X}, \mathrm{k})\text{: } \exists_{\mathrm{Y} \subseteq \mathrm{X}} \text{ takie, że elementy Y sumują się do k}\}$
   Problem NP-zupełny.

G – graf nieskierowany. Podzbiór X wierzchołków grafu G nazywamy jego pokryciem wierzchołkowym jeśli każda krawędź G sąsiaduje z co najmniej jednym wierzchołkiem ze zbioru X.

5. $VERTEX\_COVER = \{(\mathrm{G}, \mathrm{k})\text{: w G istnieje pokrycie wierzchołkowe rozmiaru k}\}$
   Problem NP-zupełny.

6. $HAMILTONIAN\_CYCLE = \{\text{G: w G istnieje cykl Hamiltona}\}$
   Problem NP-zupełny.

Wiemy, że $P \subseteq NP$ (każda deterministyczna maszyna Turinga to szczególny przypadek maszyny niedeterministycznej).

Pytanie, czy $P = NP$?

Jeśli np. postawimy przed komputerem zadanie faktoryzacji danej liczby, to niezwykle istotny jest czas, w jakim zadanie zostanie wykonane. Zbyt długi czas oznacza, że np. łamanie szyfru jest nieopłacalne gdyż użytkownik i tak go w międzyczasie zmieni. Czas potrzebny do wykonania zadania to P, a czas potrzebny do weryfikacji wyniku to NP. Jeśli zatem *P=NP*, oznacza to, że każdy problem, którego rozwiązanie może być szybko zweryfikowane, może zostać też szybko rozwiązany, istnieją efektywne (wielomianowe) algorytmy dla wszystkich problemów NP. Przez takie odkrycie cała współczesna kryptografia mogłaby upaść.