Języki, automaty i obliczenia

Wykład 13: Wielomianowy czas i pamięć

Sławomir Lasota

Uniwersytet Warszawski

27 maja 2015

Plan

Klasy złożoności

Złożoność problemów z dokładnością do wielomianu:

$$
\begin{array}{rrrrrl}
 & & \ldots & & & \\
 & & \mathrm{EXPSPACE} & = \bigcup_c \mathrm{DSPACE}(2^{n^c}) & = \bigcup_c \mathrm{NSPACE}(2^{n^c}) & \\
 & & \mathrm{NEXPTIME} & = \bigcup_c \mathrm{NTIME}(2^{n^c}) & & \\
 & & \mathrm{EXPTIME} & = \bigcup_c \mathrm{DTIME}(2^{n^c}) & & \\
 & & \mathrm{PSPACE} & = \bigcup_c \mathrm{DSPACE}(n^c) & = \bigcup_c \mathrm{NSPACE}(n^c) & \\
NP & = & \mathrm{NPTIME} & = \bigcup_c \mathrm{NTIME}(n^c) & & \\
P & = & \mathrm{PTIME} & = \bigcup_c \mathrm{DTIME}(n^c) & & \\
NL & = & \mathrm{NLOGSPACE} & = \mathrm{NSPACE}(\log(n)) & & \\
L & = & \mathrm{LOGSPACE} & = \mathrm{DSPACE}(\log(n)) & &
\end{array}
$$

Determinizacja klas pamięciowych

Twierdzenie (Savitch 1970)

Jeśli $f(n) \geq n$ *to* $NSPACE(f(n)) \subseteq DSPACE(f(n)^2)$. *założenie!*

Twierdzenie (Savitch 1970)

Jeśli $f(n) \geq n$ *to* $NSPACE(f(n)) \subseteq DSPACE(f(n)^2)$. *założenie!*

Dowód:

Niech $\mathcal{M}$ – maszyna niedet. działająca w pamięci $f(n)$. Zaprojektujemy algorytm deterministyczny sprawdzający czy $c_0 \longrightarrow^*_{\mathcal{M}} c$, dla konfiguracji akceptującej c.

Twierdzenie (Savitch 1970)

Jeśli $f(n) \geq n$ *to* $NSPACE(f(n)) \subseteq DSPACE(f(n)^2)$. *założenie!*

Dowód:

Niech $\mathcal{M}$ – maszyna niedet. działająca w pamięci $f(n)$. Zaprojektujemy algorytm deterministyczny sprawdzający czy $c_0 \longrightarrow^*_{\mathcal{M}} c$, dla konfiguracji akceptującej c.

Liczba konfiguracji $\leq 2^m$, $m = \mathcal{O}(f(n))$, więc wystarczy sprawdzić czy $c_0 \longrightarrow^{\leq 2^m}_{\mathcal{M}} c$.

Twierdzenie (Savitch 1970)

Jeśli $f(n) \geq n$ *to* $NSPACE(f(n)) \subseteq DSPACE(f(n)^2)$. *założenie!*

Dowód:

Niech $\mathcal{M}$ – maszyna niedet. działająca w pamięci $f(n)$. Zaprojektujemy algorytm deterministyczny sprawdzający czy $c_0 \longrightarrow^*_{\mathcal{M}} c$, dla konfiguracji akceptującej c.

Liczba konfiguracji $\leq 2^m$, $m = \mathcal{O}(f(n))$, więc wystarczy sprawdzić czy $c_0 \longrightarrow^{\leq 2^m}_{\mathcal{M}} c$.

Algorytm

```
bool p(c, c', k) {
    if  (k == 0) { return ((c==c') || c ⟶_M c'); }
    else {
        for each c'' {
            if (p(c, c'', k-1) && p(c'', c', k-1)) return true;
        }
        return false;
    }
}
```

Determinizacja klas pamięciowych

Twierdzenie (Savitch 1970)

Jeśli $f(n) \geq n$ *to* $NSPACE(f(n)) \subseteq DSPACE(f(n)^2)$. *założenie!*

Dowód:

Niech $\mathcal{M}$ – maszyna niedet. działająca w pamięci $f(n)$. Zaprojektujemy algorytm deterministyczny sprawdzający czy $c_0 \longrightarrow^*_{\mathcal{M}} c$, dla konfiguracji akceptującej c.

Liczba konfiguracji $\leq 2^m$, $m = \mathcal{O}(f(n))$, więc wystarczy sprawdzić czy $c_0 \longrightarrow^{\leq 2^m}_{\mathcal{M}} c$.

Algorytm

```
bool p(c, c', k) {
    if  (k == 0) { return ((c==c') || c ⟶_M c'); }
    else {
        for each c'' {
            if (p(c, c'', k-1) && p(c'', c', k-1)) return true;
        }
        return false;
    }
}
```

Pamięć: $m \cdot f(n) = \mathcal{O}(f(n)^2)$.

Wniosek

NPSPACE $=$ *PSPACE.*

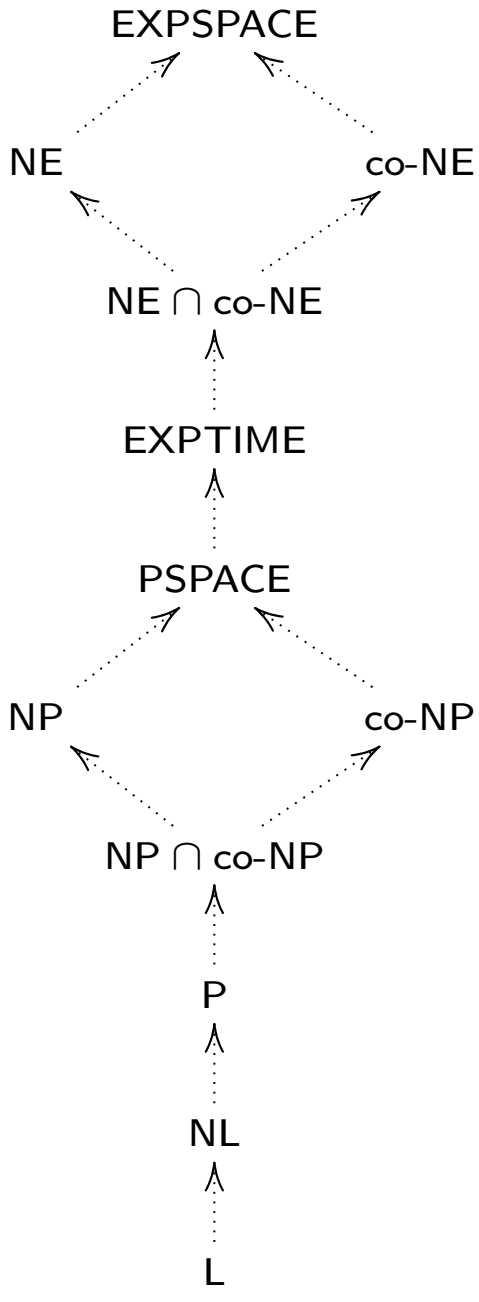
Ale nie wiemy, czy NSPACE(n) $=$ DSPACE(n).

Wniosek

NPSPACE $=$ *PSPACE.*

Ale nie wiemy, czy NSPACE(n) $=$ DSPACE(n).

co-$\mathcal{C}$ $=$ problemy, których dopełnienie należy do $\mathcal{C}$



Kolaps

Wniosek

NPSPACE = PSPACE.

Ale nie wiemy, czy NSPACE(n) = DSPACE(n).

co-$\mathcal{C}$ = problemy, których dopełnienie należy do $\mathcal{C}$

Wniosek

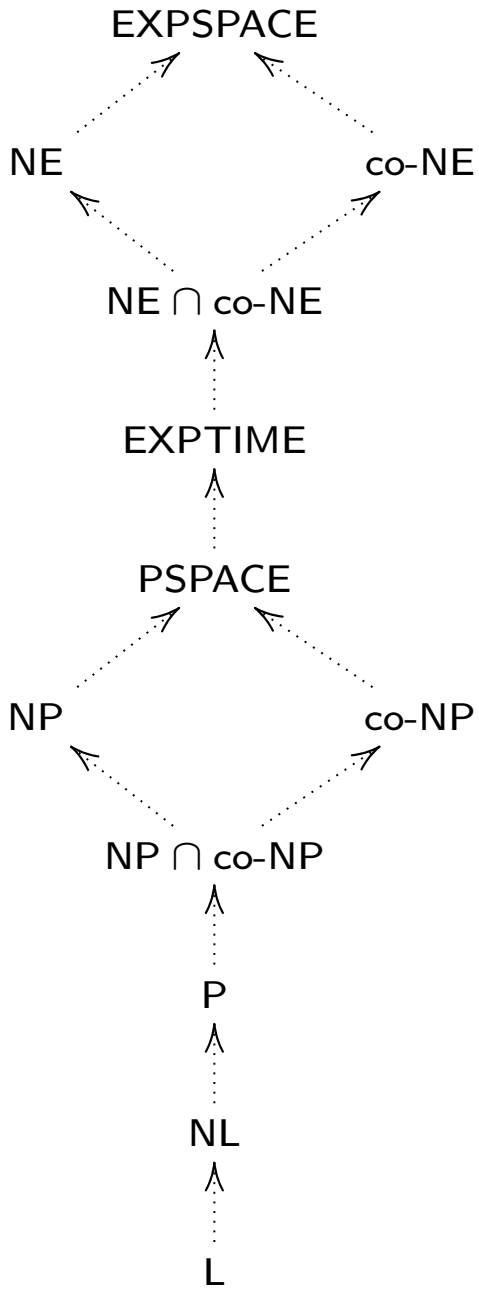
co-NPSPACE = NPSPACE.

Twierdzenie (Immerman–Szelepcsényi 1987)

co-NL = NL.

Wniosek

co-NSPACE(n) = NSPACE(n).



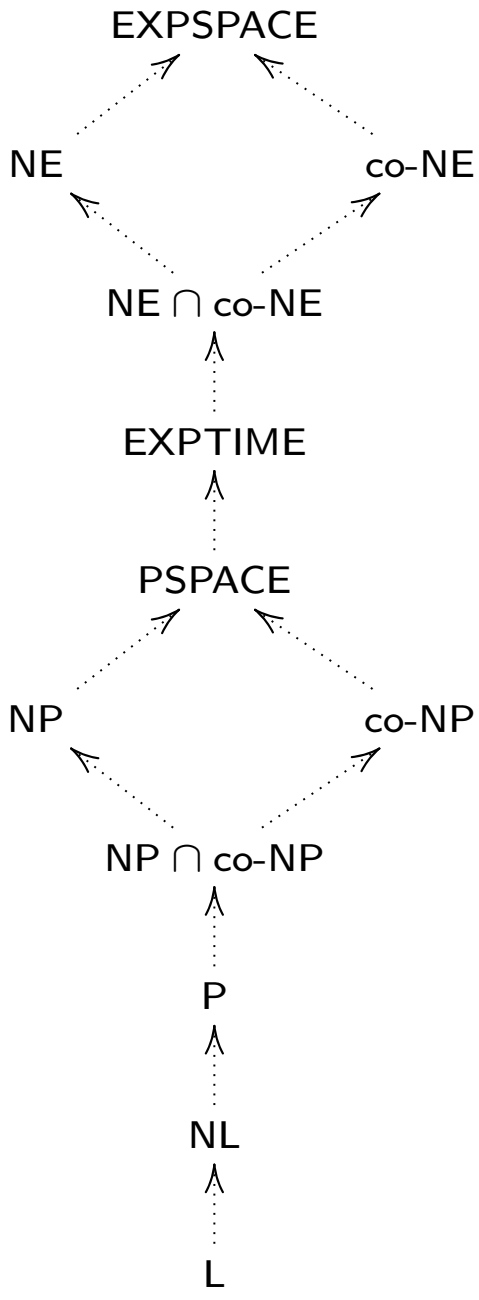
Pytania

Wiemy, że

$$\mathrm{P} \neq \mathrm{EXPTIME} \neq \text{2-EXPTIME} \neq \ldots$$

$$\mathrm{NP} \neq \mathrm{NEXPTIME} \neq \text{2-NEXPTIME} \neq \ldots$$

$$\mathrm{NL} \neq \mathrm{PSPACE} \neq \mathrm{EXPSPACE} \neq \ldots$$



Pytania

Wiemy, że

$$P \neq EXPTIME \neq 2\text{-}EXPTIME \neq \dots$$

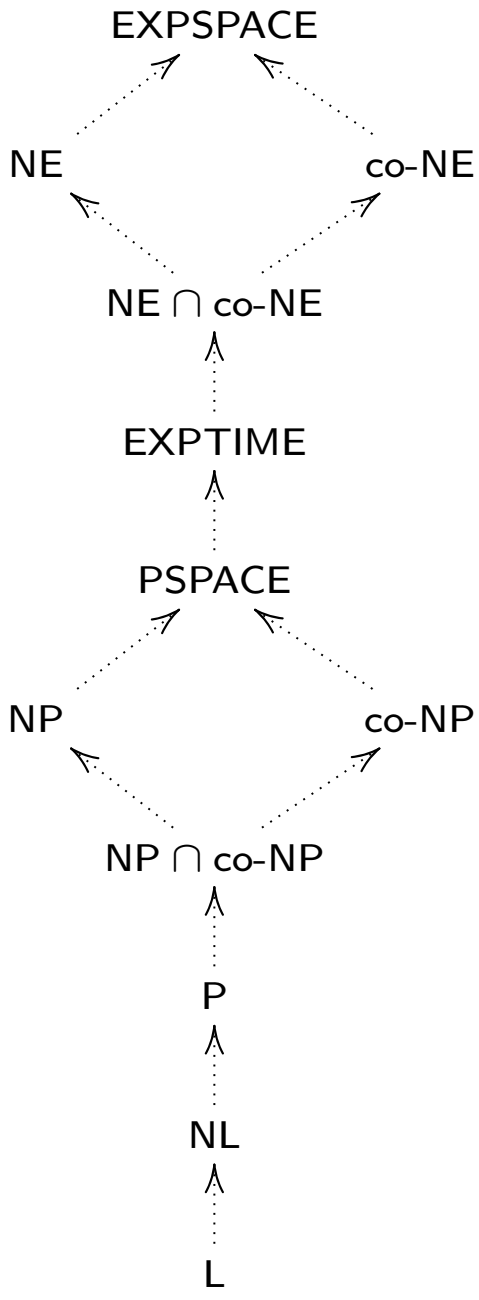
$$NP \neq NEXPTIME \neq 2\text{-}NEXPTIME \neq \dots$$

$$NL \neq PSPACE \neq EXPSPACE \neq \dots$$

Czy $P = NP$?

Pytanie

Dlaczego to pytanie jest takie ważne?



Pytania

Wiemy, że

$$\mathrm{P} \neq \mathrm{EXPTIME} \neq \text{2-EXPTIME} \neq \ldots$$

$$\mathrm{NP} \neq \mathrm{NEXPTIME} \neq \text{2-NEXPTIME} \neq \ldots$$

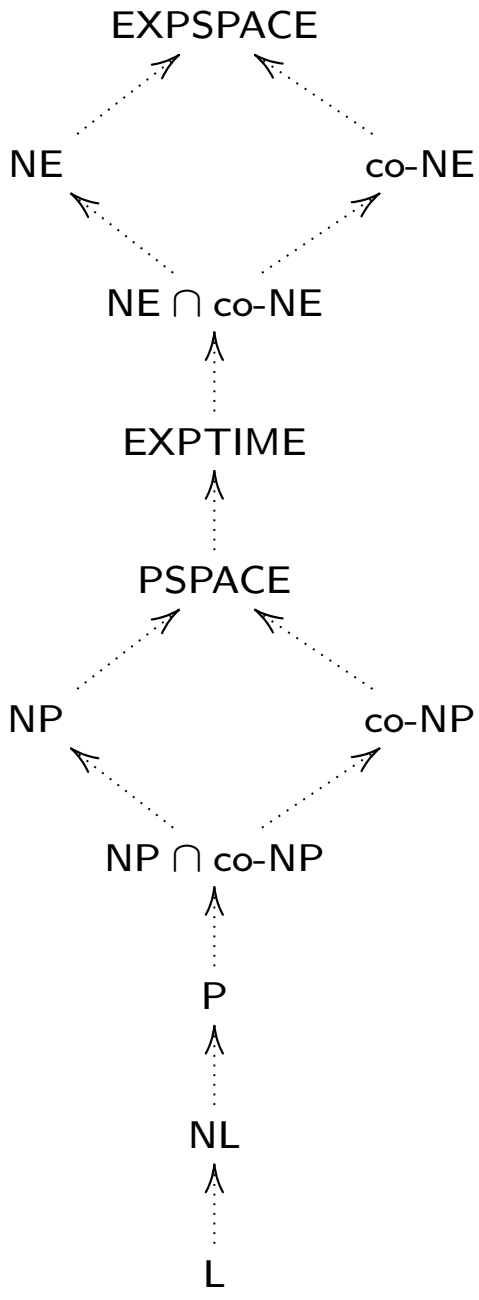
$$\mathrm{NL} \neq \mathrm{PSPACE} \neq \mathrm{EXPSPACE} \neq \ldots$$

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

Pytanie

Dlaczego to pytanie jest takie ważne?

Czy $\mathrm{P} = \mathrm{PSPACE}$?



Klasy złożoności a alternacja

$$
\begin{array}{rrrrrrrrr}
 & & & \ldots & & & & & \\
 & & & \mathrm{EXPSPACE} & = & \bigcup_c \mathrm{DSPACE}(2^{n^c}) & = & \bigcup_c \mathrm{NSPACE}(2^{n^c}) & = & \mathrm{AEXPTIME} \\
 & & & \mathrm{NEXPTIME} & = & \bigcup_c \mathrm{NTIME}(2^{n^c}) & & & & \\
 & & & \mathrm{EXPTIME} & = & \bigcup_c \mathrm{DTIME}(2^{n^c}) & & & = & \mathrm{APSPACE} \\
 & & & \mathrm{PSPACE} & = & \bigcup_c \mathrm{DSPACE}(n^c) & = & \bigcup_c \mathrm{NSPACE}(n^c) & = & \mathrm{APTIME} \\
NP & = & & \mathrm{NPTIME} & = & \bigcup_c \mathrm{NTIME}(n^c) & & & & \\
P & = & & \mathrm{PTIME} & = & \bigcup_c \mathrm{DTIME}(n^c) & & & = & \mathrm{ALOGSPACE} \\
NL & = & & \mathrm{NLOGSPACE} & = & \mathrm{NSPACE}(\log(n)) & & & & \\
L & = & & \mathrm{LOGSPACE} & = & \mathrm{DSPACE}(\log(n)) & & & &
\end{array}
$$

Plan

Redukcje wielomianowe

Problem $K \subseteq A^*$ *redukuje się* *wielomianowo* do problemu $L \subseteq B^*$ (ozn. $K \leq_p L$) jeśli istnieje funkcja obliczalna w czasie wielomianowym

$$f : A^* \to B^*$$

taka, że

$$w \in K \iff f(w) \in L, \quad \text{dla każdego } w \in A^*.$$

Zamkniętość na redukcje wielomianowe

Fakt

Niech $\mathcal{C}$ – klasa złożoności z listy poniżej. Jeśli $K \leq_p L$ i $L \in \mathcal{C}$ to $K \in \mathcal{C}$.

$$
\begin{aligned}
& & \ldots & & & \\
& & \mathrm{EXPSPACE} &= \bigcup_c \mathrm{DSPACE}(2^{n^c}) &&= \bigcup_c \mathrm{NSPACE}(2^{n^c}) \\
& & \mathrm{NEXPTIME} &= \bigcup_c \mathrm{NTIME}(2^{n^c}) && \\
& & \mathrm{EXPTIME} &= \bigcup_c \mathrm{DTIME}(2^{n^c}) && \\
& & \mathrm{PSPACE} &= \bigcup_c \mathrm{DSPACE}(n^c) &&= \bigcup_c \mathrm{NSPACE}(n^c) \\
NP &= & \mathrm{NPTIME} &= \bigcup_c \mathrm{NTIME}(n^c) && \\
P &= & \mathrm{PTIME} &= \bigcup_c \mathrm{DTIME}(n^c) &&
\end{aligned}
$$

(W przypadku klas NL, L trzeba stosować redukcje w pamięci logarytmicznej.)

Trudność i zupełność

Problem L jest $\mathcal{C}$-*trudny* jeśli każdy problem $K \in \mathcal{C}$ redukuje się wielomianowo do L.

Problem L jest $\mathcal{C}$-*zupełny* jeśli jest $\mathcal{C}$-trudny i należy do $\mathcal{C}$.

Fakt

Jeśli $K \leq_p L$ i K jest $\mathcal{C}$-trudny to L jest $\mathcal{C}$-trudny.

$$
\begin{aligned}
&\dots \\
\text{EXPSPACE} &= \bigcup_c \text{DSPACE}(2^{n^c}) &&= \bigcup_c \text{NSPACE}(2^{n^c}) \\
\text{NEXPTIME} &= \bigcup_c \text{NTIME}(2^{n^c}) \\
\text{EXPTIME} &= \bigcup_c \text{DTIME}(2^{n^c}) \\
\text{PSPACE} &= \bigcup_c \text{DSPACE}(n^c) &&= \bigcup_c \text{NSPACE}(n^c) \\
NP = \quad \text{NPTIME} &= \bigcup_c \text{NTIME}(n^c)
\end{aligned}
$$

(W przypadku klas P, NL trzeba stosować redukcje w pamięci logarytmicznej.)

Plan

NP-trudność i zupełność

Problem L jest *NP-trudny* jeśli każdy $K \in \text{NP}$ redukuje się wielomianowo do L.

Problem L jest *NP-zupełny* jeśli jest NP-trudny i należy do NP.

Fakt

Jeśli $K \leq_p L$ i K jest NP-trudny to L jest NP-trudny.

Problem spełnialności formuły zdaniowej (SAT)

Dane: formuła zdaniowa ϕ np. $x \wedge y \vee (x \wedge (\neg y \vee z))$.

Wynik: czy ϕ jest spełnialna?

Twierdzenie (Cook 1971, Levin 1973)

SAT jest NP-zupełny.

SAT jest NP-zupełny (dowód)

Niech $\mathcal{M}$ – maszyna niedet. działająca w czasie n^c. Pokażemy $L(M) \leq_p$ SAT.

funkcja obliczalna w czasie wiel.:	$w = a_1 \dots a_n$	$\longmapsto$	$\phi_{\mathcal{M},w}$
poprawność:	$w \in L(\mathcal{M})$	$\Longleftrightarrow$	$\phi_{\mathcal{M},w}$ spełnialna

SAT jest NP-zupełny (dowód)

Niech $\mathcal{M}$ – maszyna niedet. działająca w czasie n^c. Pokażemy $L(M) \leq_p$ SAT.

funkcja obliczalna w czasie wiel.:	$w = a_1 \dots a_n$	$\longmapsto$	$\phi_{\mathcal{M},w}$
poprawność:	$w \in L(\mathcal{M})$	$\Longleftrightarrow$	$\phi_{\mathcal{M},w}$ spełnialna

Zmienne:

- $t^{i,j,a}$ – po i krokach, na pozycji j taśmy jest symbol a
- $s^{i,q}$ – po i krokach, stan maszyny to q
- $g^{i,j}$ – po i krokach, głowica maszyny jest na pozycji j taśmy

SAT jest NP-zupełny (dowód)

Niech $\mathcal{M}$ – maszyna niedet. działająca w czasie n^c. Pokażemy $L(M) \leq_p$ SAT.

funkcja obliczalna w czasie wiel.:	$w = a_1 \dots a_n$	$\longmapsto$	$\phi_{\mathcal{M},w}$
poprawność:	$w \in L(\mathcal{M})$	$\iff$	$\phi_{\mathcal{M},w}$ spełnialna

Zmienne:

- $t^{i,j,a}$ – po i krokach, na pozycji j taśmy jest symbol a
- $s^{i,q}$ – po i krokach, stan maszyny to q
- $g^{i,j}$ – po i krokach, głowica maszyny jest na pozycji j taśmy

$$\bigwedge_{j \leq |w|} t^{0,j,a_j} \quad \wedge \quad \bigwedge_{j > |w|} t^{0,j,\mathbb{B}} \quad \wedge \quad s^{0,q_0} \quad \wedge \quad g^{0,1} \qquad \text{(konfiguracja początkowa)}$$

$$\bigwedge_{i,j} \Big(\bigvee_{a} t^{i,j,a} \quad \wedge \quad \bigwedge_{a \neq b} \neg (t^{i,j,a} \wedge t^{i,j,b}) \Big) \quad \wedge \quad \dots \qquad \text{(niesprzeczność)}$$

$$\bigvee_{i} s^{i,q_F} \qquad \text{(akceptacja)}$$

$$\bigwedge_{i,j} t^{i,j,a} \wedge s^{i,q} \wedge g^{i,j} \implies \qquad \text{przejścia z } (q,a):$$

$$t^{i+1,j,a'} \wedge s^{i+1,q'} \wedge g^{i+1,j-1} \quad \vee \qquad (q, a, q', a', \leftarrow)$$

$$t^{i+1,j,a''} \wedge s^{i+1,q''} \wedge g^{i+1,j} \qquad (q, a, q'', a'', \circlearrowright)$$

co-NP-trudność i zupełność

Problem L jest *co-NP-trudny* jeśli każdy $K \in$ co-NP redukuje się wielomianowo do L.

Problem L jest *co-NP-zupełny* jeśli jest co-NP-trudny i należy do co-NP.

Fakt

Jeśli $K \leq_p L$ i K jest co-NP-trudny to L jest co-NP-trudny.

Problem tautologii zdaniowej

Dane: formuła zdaniowa ϕ np. $x \wedge y \vee (x \wedge (\neg y \vee z))$.

Wynik: czy ϕ jest tautologią?

Wniosek

Problem tautologii zdaniowej jest co-NP-zupełny.

Problemy NP-zupełne (przykłady)

3-kolorowalność

Dane: graf nieskierowany G.

Wynik: czy da się pokolorować wierzchołki G trzema kolorami tak, żeby kolory sąsiadów były różne?

Problemy NP-zupełne (przykłady)

3-kolorowalność

Dane: graf nieskierowany G.

Wynik: czy da się pokolorować wierzchołki G trzema kolorami tak, żeby kolory sąsiadów były różne?

Problem plecakowy (szczególny przypadek)

Dane: zbiór liczb $\{n_1, \ldots, n_k\}$ i liczba m, reprezentowane binarnie.

Wynik: czy istnieje podzbiór $\{n_{i_1}, \ldots, n_{i_l}\}$ taki, że $n_{i_1} + \ldots + n_{i_l} = m$?

Problemy NP-zupełne (przykłady)

3-kolorowalność

Dane: graf nieskierowany G.

Wynik: czy da się pokolorować wierzchołki G trzema kolorami tak, żeby kolory sąsiadów były różne?

Problem plecakowy (szczególny przypadek)

Dane: zbiór liczb $\{n_1, \ldots, n_k\}$ i liczba m, reprezentowane binarnie.

Wynik: czy istnieje podzbiór $\{n_{i_1}, \ldots, n_{i_l}\}$ taki, że $n_{i_1} + \ldots + n_{i_l} = m$?

PCP z ograniczeniem

Dane: ciąg par słów $(w_1, v_1), \ldots, (w_n, v_n)$ i liczba k reprezentowana unarnie.

Wynik: czy istnieje niepusty ciąg $(i_1, \ldots, i_m)$, $m \leq k$, t. że $w_{i_1} \ldots w_{i_m} = v_{i_1} \ldots v_{i_m}$?

Problem 3-kolorowalności jest NP-zupełny (dowód)

Redukcja 3-SAT $\leq_p$ 3-kolorowalność.

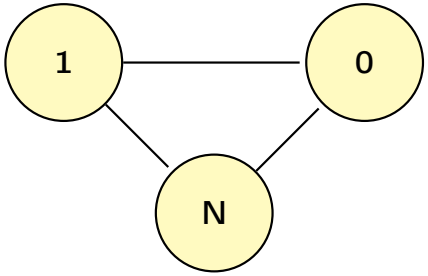
funkcja obliczalna w czasie wiel.:	ϕ w postaci 3-CNF	$\longmapsto$	G_ϕ
poprawność:	ϕ spełnialna	$\Longleftrightarrow$	G_ϕ 3-kolorowalny

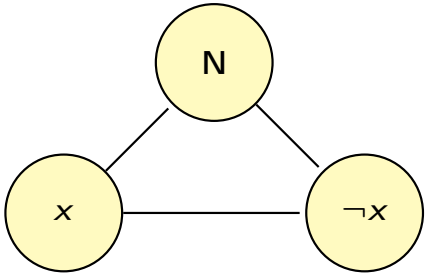
Problem 3-kolorowalności jest NP-zupełny (dowód)

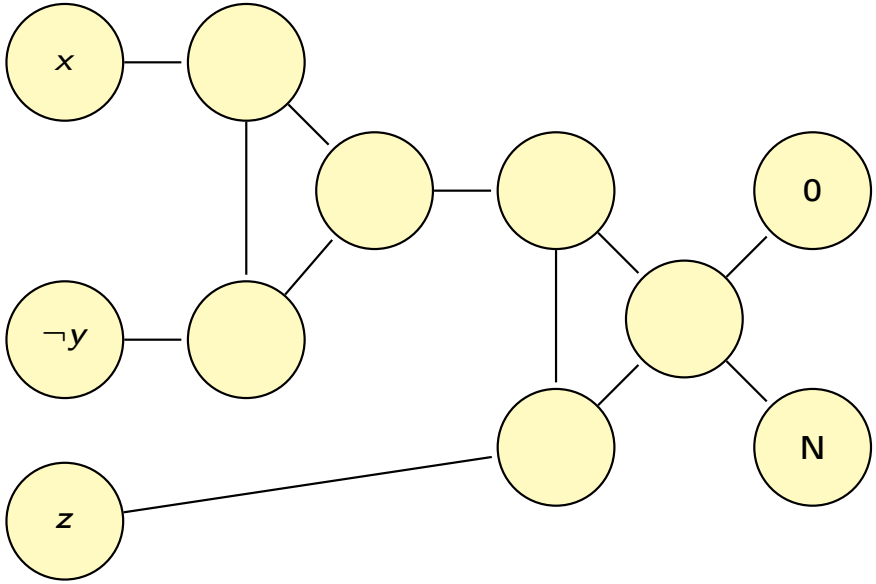
Redukcja 3-SAT $\leq_p$ 3-kolorowalność.

funkcja obliczalna w czasie wiel.:	ϕ w postaci 3-CNF	$\longmapsto$	G_ϕ
poprawność:	ϕ spełnialna	$\Longleftrightarrow$	G_ϕ 3-kolorowalny

$$\phi \equiv (x \vee \neg y \vee z) \wedge (\neg x \vee \neg z \vee u) \wedge \ldots$$







Problemy w NP, o których nie wiemy ani że są NP-zupełne, ani że są w P:

Izomorfizm grafów

Dane: Dwa grafy G, H

Wynik: Czy G i H są izomorficzne?

Problemy w NP, o których nie wiemy ani że są NP-zupełne, ani że są w P:

Izomorfizm grafów

Dane: Dwa grafy G, H

Wynik: Czy G i H są izomorficzne?

Gra parzystości:

- gracze: Parzysty, Nieparzysty
- wierzchołki grafu etykietowane liczbami
- gramy do pierwszej powtórki (cyklu)
- zwycięzca określony przez parzystość największej liczby na cyklu

P $\neq$ NP ?

Problemy w NP, o których nie wiemy ani że są NP-zupełne, ani że są w P:

Izomorfizm grafów

Dane: Dwa grafy G, H

Wynik: Czy G i H są izomorficzne?

Gra parzystości:

- gracze: Parzysty, Nieparzysty
- wierzchołki grafu etykietowane liczbami
- gramy do pierwszej powtórki (cyklu)
- zwycięzca określony przez parzystość największej liczby na cyklu

Kto wygrywa grę parzystości?

Dane: graf gry parzystości

Wynik: czy Parzysty ma strategię wygrywającą?

Problem należy do NP $\cap$ co-NP.

Pierwszość

Dane: liczba naturalna $n \in \mathbb{N}$, reprezentowana binarnie.

Wynik: czy n jest liczbą pierwszą?

Pierwszość

Dane: liczba naturalna $n \in \mathbb{N}$, reprezentowana binarnie.

Wynik: czy n jest liczbą pierwszą?

Twierdzenie (Agrawal, Kayal, Saxena 2004)

Problem pierwszości jest w P.

Plan

Problemy PSPACE-zupełne

Problem spełnialności kwantyfikowanej formuły zdaniowej (QBF)

Dane: formuła postaci $\forall x_1 \exists x_2 \forall x_3 \ldots \phi$ np. $\forall x \exists y.\ x \wedge y \vee (x \wedge (\neg y \vee z))$.

Wynik: czy ϕ jest spełnialna (tautologią)?

Problemy PSPACE-zupełne

Problem spełnialności kwantyfikowanej formuły zdaniowej (QBF)

Dane: formuła postaci $\forall x_1 \exists x_2 \forall x_3 \ldots \phi$ np. $\forall x \exists y.\ x \wedge y \vee (x \wedge (\neg y \vee z))$.

Wynik: czy ϕ jest spełnialna (tautologią)?

Kto wygrywa w Go (bez ko)?

Dane: pozycja w grze Go na dowolnie dużej planszy.

Wynik: czy białe wygrywają?

A gdy ko jest dozwolone?

Problemy PSPACE-zupełne (dla języków, automatów, itp.)

- Uniwersalność wyrażenia regularnego (albo automatu niedet.).
- Równoważność dwóch wyrażeń regularnych (albo automatów niedet.).
- Niepustość uogólnionego wyrażenia regularnego (z operacją dopełnienia).
- Niepustość przecięcia wyrażeń regularnych (albo automatów niedet.).
- Czy dane wyrażenie regularne jest równoważne wyrażeniu bezgwiazdkowemu (z operacją dopełnienia).
- Problem stopu dla maszyn liniowo ograniczonych.
- Problem słów dla gramatyk kontekstowych.
- ...