Vlastnosti tříd složitosti

k-páskové Turingovy stroje

Věta 7.1 Je-li jazyk L přijímán nějakým k-páskovým DTS M_k v čase $t(n)$, pak je také přijímán nějakým 1-páskovým DTS M_1 v čase $O(t(n)^2)$.

Důkaz. (idea)

- Obsah k pásek stroje M_k můžeme zapsat na jednu pásku stroje M_1 za sebe a oddělit je vhodným oddělovačem. Pozici hlav M_k na pásce můžeme zakódovat vhodným označením symbolů, pod kterými se hlavy aktuálně nacházejí.
- Při simulaci kroku stroje M_k stroj M_1 dvakrát projde páskou – při jednom průchodu zjistí znaky pod hlavami M_k, při druhém je patřičně upraví.
- Jestliže je na některé simulované pásce stroje M_k nedostatek prostoru, je zapotřebí provést posun zbytku pásky stroje M_1, což si může opět vyžádat dva průchody touto páskou.
- Užitečná délka pásek stroje M_k je ovšem omezena na $t(n)$ a tudíž užitečná délka pásky stroje M_1 je omezena na $k.t(n) + k$, tj. $O(t(n))$.
- Simulace jednoho kroku M_k strojem M_1 je proto v $O(t(n))$.
- Takových kroků se provede $t(n)$ a tudíž M_1 přijímá L v čase $O(t(n)^2)$.

□

Determinismus a nedeterminismus

Věta 7.2 Je-li jazyk L přijímán nějakým NTS M_n v čase $t(n)$, pak je také přijímán nějakým DTS M_d v čase $2^{O(t(n))}$.

Důkaz. (idea) Ukážeme, jak může M_d simulovat M_n v uvedeném čase:

- Očíslujeme si přechody M_n jako $1, 2, ..., k$.
- M_d bude postupně simulovat veškeré možné posloupnosti přechodů M_n (obsah vstupní pásky si uloží na pomocnou pásku, aby ho mohl vždy obnovit; na jinou pásku si vygeneruje posloupnost přechodů z $\{1, 2, ..., k\}^*$ a tu simuluje).
- Vzhledem k možnosti nekonečných výpočtů M_n nelze procházet jeho možné výpočty do hloubky – budeme-li je ale procházet do šířky (tj. nejdřív všechny řetězce z $\{1, 2, ..., k\}^*$ délky 1, pak 2, pak 3, ...), určitě nalezneme nejkratší přijímající posloupnost přechodů pro M_n, existuje-li.
- Takto projdeme nanejvýš $O(k^{t(n)})$ cest, simulace každé z nich je v $O(t(n))$ a tudíž celkem využijeme nanejvýš čas $O(k^{t(n)})O(t(n)) = 2^{O(t(n))}$.

□

❖ Doposud nikdo nebyl schopen přijít s polynomiální simulací NTS pomocí DTS. **Zdá se** tedy, že nedeterminismus přináší značnou výhodu z hlediska časové složitosti výpočtů.

❖ Zatímco *se zdá*, že nedeterminismus přináší značnou výhodu z hlediska časové složitosti výpočtů, v případě prostorové složitosti je situace jiná:

Věta 7.3 (Savitchův teorém) $NSpace[s(n)] \subseteq DSpace[s^2(n)]$ pro každou prostorově zkonstruovatelnou funkci $s(n) \geq lg\ n$.

Důkaz. Uvažme NTS $M = (Q, \Sigma, \Gamma, \delta, q_0, q_F)$ rozhodující $L(M)$ v prostoru $s(n)$:

- Existuje $k \in \mathbb{N}$ závislé jen na $|Q|$ a $|\Gamma|$ takové, že pro libovolný vstup w, M projde nanejvýš $k^{s(n)}$ konfigurací o délce max. $s(n)$.
- To implikuje, že M provede pro w nanejvýš $k^{s(n)} = 2^{s(n)lg\ k}$ kroků.
- Pomocí DTS můžeme snadno implementovat proceduru $test(c, c', i)$, která otestuje, zda je možné v M dojít z konfigurace c do c' v 2^i krocích:
  procedure $test(c, c', i)$
  if $(i = 0)$ **then return** $((c = c') \vee (c \underset{M}{\vdash} c'))$
  else for all configurations c'' such that $|c''| \leq s(n)$ **do**
  if $(test(c, c'', i-1) \wedge test(c'', c', i-1))$ **then return** $true$
  return $false$

Důkaz pokračuje dále.

Pokračování důkazu.

- Všimněme si, že rekurzivním vyvoláváním $test$ vzniká strom o výšce i simulující posloupností svých listů posloupnost 2^i výpočetních kroků.
- Nyní k deterministické simulaci M postačí projít všechny akceptující konfigurace c_F takové, že $|c_F| \leq s(n)$, a ověřit, zda $test(c_o, c_f, \lceil s(n) lg\ k \rceil)$, kde c_0 je počáteční konfigurace.
- Každé vyvolání $test$ zabere prostor $O(s(n))$, hloubka rekurze je $\lceil s(n) lg\ k \rceil = O(s(n))$ a tedy celkově deterministicky simulujeme M v prostoru $O(s^2(n))$.
- Dodejme, že $s(n)$ může být zkonstruováno v prostoru $O(s(n))$ (jedná se o prostorově zkonstruovatelnou funkci) a tudíž neovlivňuje výše uvedené úvahy.

□

❖ Důsledkem Savitchova teorému jsou již dříve uvedené rovnosti:

- **PSPACE** $\equiv$ **NPSPACE**,
- **k-EXPSPACE** $\equiv$ **k-NEXPSPACE**.

Prostor kontra čas

❖ Intuitivně můžeme říci, že zatímco prostor může růste relativně pomalu, čas může růst výrazně rychleji, neboť můžeme opakovaně procházet týmiž buňkami pásky – opačně tomu být zřejmě nemůže (nemá smysl mít nevyužitý prostor).

Věta 7.4 *$NSpace[t(n)] \subseteq DTime[O(1)^{t(n)}]$ pro každou časově zkonstruovatelnou funkci $t(n) \geq lg\ n$.

Důkaz. Dá se použít do jisté míry podobná konstrukce jako u Savitchova teorému – blíže viz literatura. □ *

Věty o kompresi prostoru a zrychlení

Věta 7.5 Nechť M je DTS. Pak existuje ekvivalentní TS N takový, že

$$S_N(n) \leq \frac{S_M(n)}{2} + 2$$

Důkaz. (Idea) Abeceda stroje N obsahuje dvojice znaků abecedy stroje M. Obsah pásky $a_1, a_2, \ldots, a_n$ stroje M pak bude reprezentován jako

$$\begin{pmatrix} a_1 \\ a_2 \end{pmatrix}, \begin{pmatrix} a_2 \\ a_3 \end{pmatrix}, \ldots$$

□

Věta 7.6 Nechť M je DTS. Pak existuje ekvivalentní TS N takový, že

$$T_N(n) \leq \frac{T_M(n)}{2} + 2n$$

Důkaz. (Idea) Stroj N si předpočítá výsledky všech možných výpočtů délky 2 stroje M.

□

Uzavřenost vůči doplňku

❖ Doplňkem třídy rozumíme třídu jazyků, které jsou doplňkem jazyků dané třídy. Tedy označíme-li doplněk třídy $\mathcal{C}$ jako $co\text{-}\mathcal{C}$, pak $L \in \mathcal{C} \Leftrightarrow \overline{L} \in co\text{-}\mathcal{C}$. U rozhodování problémů toto znamená rozhodování komplementárního problému (prázdnost x neprázdnost apod.).

❖ Prostorové třídy jsou obvykle uzavřeny vůči doplňku:

Věta 7.7 *Jestliže $s(n) \geq lg\ n$, pak $NSpace(s(n)) = co\text{-}NSpace(s(n))$.

Důkaz. Jedná se o teorém Immermana a Szelepcsényiho – důkaz viz literatura. □ *

❖ Pro časové třídy je situace jiná:

- Některé třídy jako **P** či **EXP** jsou uzavřeny vůči doplňku.
- U jiných významných tříd zůstává otázka uzavřenosti vůči doplňku otevřená. Proto má smysl hovořit např. i o třídách jako:
  - **co-NP** či
  - **co-NEXP**.

Věta 7.8 Třída **P** je uzavřena vůči doplňku.

Důkaz. (idea) Základem je to, že ukážeme, že jestliže jazyk L nad Σ může být přijat DTS M v polynomiálním čase, pak také existuje DTS M', který L rozhoduje v polynomiálním čase, tj. $L = L(M')$ a existuje $k \in \mathbb{N}$ takové, že pro každé $w \in \Sigma^*$ M' zastaví v čase $O(|w|^k)$:

- M' na začátku výpočtu určí délku vstupu w a vypočte $p(|w|)$, kde $p(n)$ je polynom určující složitost přijímání strojem M. Na speciální dodatečnou pásku uloží $p(|w|)$ symbolů.
- Následně M' simuluje M, přičemž za každý krok umaže jeden symbol z dodatečné pásky. Pokud odebere z této pásky všechny symboly a M by mezitím nepřijal, M' abnormálně ukončí výpočet (odmítne).
- M' evidentně přijme všechny řetězce, které přijme M na to mu stačí $p(n)$ simulovaných kroků a nepřijme všechny řetězce, které by M nepřijal – pokud M nepřijme v $p(n)$ krocích, nepřijme vůbec. M' však vždy zastaví v $O(p(n))$ krocích.

□

Ostrost hierarchie tříd

❖ Z dosavadního můžeme shrnout, že platí následující:

- **LOGSPACE** $\subseteq$ **NLOGSPACE** $\subseteq$ **P** $\subseteq$ **NP**
- **NP** $\subseteq$ **PSPACE** $=$ **NPSPACE** $\subseteq$ **EXP** $\subseteq$ **NEXP**
- **NEXP** $\subseteq$ **EXPSPACE** $=$ **NEXPSPACE** $\subseteq$ **2-EXP** $\subseteq$ **2-NEXP** $\subseteq$ ...
- ... $\subset$ **ELEMENTARY** $\subset$ **PR** $\subset$ **R** $\subset$ **RE**[a]

❖ Řada otázek ostrosti uvedených vztahů pak zůstává otevřená, nicméně z tzv. teorému hierarchie (nebudeme ho zde přesně uvádět, neboť je velmi technický – zájemci ho naleznou v literatuře) plyne, že exponenciální „mezery“ mezi třídami jsou „ostré“:

- **LOGSPACE**, **NLOGSPACE** $\subset$ **PSPACE**,
- **P** $\subset$ **EXP**,
- **NP** $\subset$ **NEXP**,
- **PSPACE** $\subset$ **NEXPSPACE**,
- **EXP** $\subset$ **2-EXP**, ...

[a] Bez důkazu jsme doplnili, že $\mathbf{ELEMENTARY} \subset \mathbf{PR}$.

Polynomiální redukce a SAT-problém

Polynomiální redukce

Definice 7.1 *Polynomiální redukce* jazyka L_1 nad abecedou Σ_1 na jazyk L_2 nad abecedou Σ_2 je funkce $f : \Sigma_1^* \to \Sigma_2^*$, pro kterou platí:

1. $\forall w \in \Sigma_1^* : w \in L_1 \Leftrightarrow f(w) \in L_2$
2. f je vyčíslitelná DTS v polynomiálním čase.

Existuje-li polynomiální redukce jazyka L_1 na L_2, říkáme, že L_1 se *(polynomiálně) redukuje na* L_2 a píšeme $L_1 \leq_P^m L_2$.

Věta 7.9 Je-li $L_1 \leq_P^m L_2$ a L_2 je ve třídě P, pak L_1 je ve třídě P.

Důkaz. Nechť M_f je Turingův stroj, který provádí redukci f jazyka L_1 na L_2 a nechť $p(x)$ je jeho časová složitost. Pro libovolné $w \in L_1$ výpočet $f(w)$ vyžaduje nanejvýš $p(|w|)$ kroků a produkuje výstup maximální délky $p(|w|) + |w|$.
Nechť M_2 přijímá jazyk L_2 v polynomiálním čase daném polynomem $q(x)$.
Uvažujme Turingův stroj, který vznikne kompozicí $M_f M_2$. Tento stroj přijímá jazyk L_1 tak, že pro každé $w \in L_1$ udělá stroj $M_f M_2$ maximálně $p(|w|) + q(p(|w|) + |w|)$ kroků, což je polynomem ve $|w|$ a tedy L_1 leží ve třídě P. □

Problém splnitelnosti – SAT problém

❖ Nechť $V = \{v_1, v_2, \ldots, v_m\}$ je konečná množina Booleovských proměnných (prvotních formulí výrokového počtu). *Literálem* nazveme každou proměnnou v_i nebo její negaci $\overline{v_i}$. *Klausulí* nazveme výrokovou formuli obsahující pouze literály spojené výrokovou spojkou $\vee$ (nebo).

Příklady klausulí: $v_1 \vee \overline{v_2}, \quad v_2 \vee v_3, \quad \overline{v_1} \vee \overline{v_3} \vee v_2.$

❖ Množina klausulí nad množinou proměnných V odpovídá výrokové formuli v konjunktivní normální formě.

❖ *SAT-problém* lze formulovat takto: Je dána formule Φ_V nad množinou proměnných V splnitelná?

$$L_{SAT} = \{\Phi_V \mid \Phi_V \text{ je splnitelná formule}\}$$

Problém splnitelnosti – kódování

❖ Každý konkrétní SAT-problém můžeme zakódovat jediným řetězcem takto: Nechť $V = \{v_1, v_2, \ldots, v_m\}$, každý literál v_i zakódujeme řetězcem délky m, který obsahuje samé 0 s výjimkou i-té pozice, která obsahuje symbol p, jde-li o literál v_i, nebo n, jde-li o literál $\overline{v_i}$. Klausuli reprezentujeme seznamem zakódovaných literálů oddělených symbolem $/$. SAT-problém bude seznam klausulí uzavřených v aritmetických závorkách.

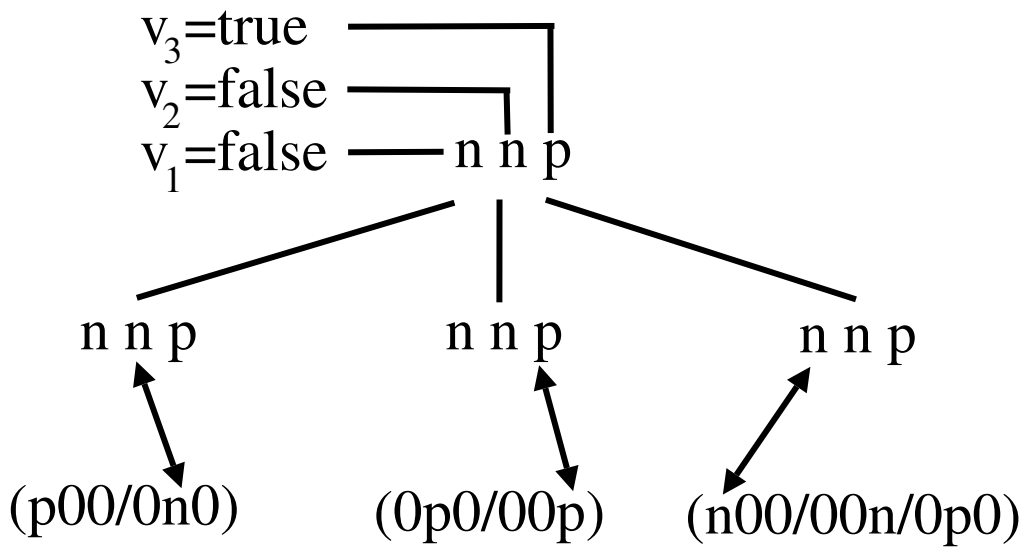
Příklad 7.1 SAT-problém obsahuje proměnné v_1, v_2, v_3 a klausule $v_1 \vee \overline{v_2}$, $v_2 \vee v_3$, $\overline{v_1} \vee \overline{v_3} \vee v_2$ bude reprezentována řetězcem: $(p00/0n0)(0p0/00p)(n00/00n/0p0)$.

❖ Označme L_{SAT} jazyk obsahující řetězce tohoto typu, které reprezentují splnitelné množiny klausulí.

Řetězec $(p00/0n0)(0p0/00p)(n00/00n/0p0)$ je prvkem L_{SAT} $(v_1 = F, v_2 = F, v_3 = T)$, na rozdíl od řetězce $(p00/0p0)(n00/0p0)(p00/0n0)(n00/0n0)$, který je kódem nesplnitelné množiny klausulí $v_1 \vee v_2$, $\overline{v_1} \vee v_2$, $v_1 \vee \overline{v_2}$, $\overline{v_1} \vee \overline{v_2}$.

❖ Přiřazení pravdivostních hodnot budeme reprezentovat řetězcem z $\{p, n\}^+$, kde p v i-té pozici představuje přiřazení $v_i \approx$ true a n v i-té pozici představuje přiřazení $v_i \approx$ false.

❖ Pak test, zda určité hodnocení je modelem množiny klausulí (množina klausulí je pro toto hodnocení splněna), je velmi jednoduchý a ilustruje ho obrázek:



L_{SAT} je **NP**-úplný

❖ Na uvedeném principu můžeme zkonstruovat nedeterministický Turingův stroj, který přijímá jazyk L_{SAT} v polynomiálním čase. Zvolíme 2-páskový Turingův stroj, který:

1. začíná kontrolou, zda vstup reprezentuje množinu klausulí,
2. na $2.$ pásku nageneruje řetězec z $\{n, p\}^m$ nedeterministickým způsobem,
3. posouvá hlavu na $1.$ pásce a testuje, zda pro dané ohodnocení (na $2.$ pásce) je množina klausulí splnitelná.

Tento proces může být snadno implementován s polynomiální složitostí přijetí v závislosti na délce vstupního řetězce a tedy $L_{SAT} \in NP$.

❖ Princip „guess & check“ použitý výše se často užívá při ukázání členství v **NP**.

Věta 7.10 *Cookův teorém:* Je-li L libovolný jazyk z NP, pak $L \leq_P^m L_{SAT}$.

Hlavní myšlenka důkazu: Protože $L \in NP$, existuje nedeterministický Turingův stroj M a polynom $p(x)$ tak, že pro každé $w \in L$ stroj M přijímá w v maximálně $p(|w|)$ krocích. Jádro důkazu tvoří konstrukce polynomiální redukce f z L na L_{SAT}: Pro každý řetězec $w \in L$ bude $f(w)$ množina klausulí, které jsou splnitelné, právě když M přijímá w.

***NP**-úplné jazyky*

❖ Po objevení Cookova teorému se ukázalo, že mnoho dalších **NP** jazyků má vlastnost podobnou jako L_{SAT}, t.j. jsou polynomiálními redukcemi ostatních **NP** jazyků.

❖ Tyto jazyky se – jak již víme – nazývají ***NP**-úplné* (**NP**-complete) jazyky.

❖ Kdyby se ukázalo, že libovolný z těchto jazyků je v **P**, pak by muselo platit **P** $=$ **NP**; naopak důkaz, že některý z nich leží mimo **P** by znamenalo **P** $\subset$ **NP**.

Význačné NP-úplné problémy

- *Satisfiability*: Je boolovský výraz splnitelný?
- *Clique*: Obsahuje neorientovaný graf kliku velikosti k?
- *Vertex cover*: Má neorientovaný graf dominantní množinu mohutnosti k?
- *Hamilton circuit*: Má neorientovaný graf Hamiltonovskou kružnici?
- *Colorability*: Má neorientovaný graf chromatické číslo k?
- *Directed Hamilton circuit*: Má neorientovaný graf Hamiltonovský cyklus?
- *Set cover*: Je dána třída množin $S_1, S_2, \ldots, S_n$. Existuje podtřída k množin $S_{i_1}, S_{i_2}, \ldots, S_{i_k}$ taková, že $\bigcup_{j=1}^{k} S_{i_j} = \bigcup_{j=1}^{n} S_j$?
- *Exact cover*: Je dána třída množin $S_1, S_2, \ldots, S_n$. Existuje množinové pokrytí (set cover) tvořené podtřídou po dvojicích disjunktních množin?

Vybrané polynomiální redukce

Problém 3SAT je NP-úplný

❖ Nechť Φ_V^3 označuje výrokovou formuli v konjunktivní normální formě nad množinou proměnných V, kde každá klauzule má přesně *3 literály*.

$$L_{3SAT} = \{\Phi_V^3 \mid \Phi_V^3 \text{ je splnitelná}\}$$

❖ $L_{3SAT} \in \mathbf{NP}$: NTS nejdříve ověří, že každá klauzule má 3 literály (lineární čas k délce formule) a pak uhádne přiřazení a ověří, zda je splnitelné (stejně jako u L_{SAT}) – opět lineární čas k délce formule.

❖ $L_{SAT} \leq_P^m L_{3SAT}$

❖ Hlavní myšlenka redukce: Transformace klauzulí, které nemají 3 literály bez změny splnitelnosti formule:

- klauzule x_1 nahradíme $x_1 \vee x_1 \vee x_1$
- klauzule $x_1 \vee x_2$ nahradíme $x_1 \vee x_2 \vee x_1$
- klauzuli $c = x_1 \vee x_2 \vee x_3 \vee x_4$ nahradíme ....

Problém 3SAT je NP-úplný

❖ Nechť Φ_V^3 označuje výrokovou formuli v konjunktivní normální formě nad množinou proměnných V, kde každá klauzule má přesně *3 literály*.

$$L_{3SAT} = \{\Phi_V^3 \mid \Phi_V^3 \text{ je splnitelná}\}$$

❖ $L_{SAT} \leq_P^m L_{3SAT}$

❖ Hlavní myšlenka redukce: Transformace klauzulí, které nemají 3 literály bez změny splnitelnosti formule:

- klauzule x_1 nahradíme $x_1 \vee x_1 \vee x_1$
- klauzule $x_1 \vee x_2$ nahradíme $x_1 \vee x_2 \vee x_1$
- klauzuli $c = x_1 \vee x_2 \vee x_3 \vee x_4$ nahradíme klauzulemi
  $c_1 = x_1 \vee x_2 \vee y$ a $c_2 = \neg y \vee x_3 \vee x_4$ kde $y \notin V$ (tj. y je čerstvá proměnná)

c je splnitelná $\iff$ $c_1 \wedge c_2$ je splnitelná

Problém 3SAT je NP-úplný

- klauzuli $c = x_1 \vee \ldots \vee x_n$ nahradíme klauzulemi

$$(x_1 \vee x_2 \vee y_1) \quad (\neg y_1 \vee x_3 \vee y_2) \quad (\neg y_2 \vee x_4 \vee y_3) \ \ldots \ (\neg y_{n-3} \vee x_{n-1} \vee x_n)$$

- transformace zachovává splnitelnost
- každá nová klauzule zmenší původní klauzuli aspoň o jeden literál.
- popsaná transformace jednou projde původní formuli – pracuje v čase $O(n)$
- formule se zvetší pouze o faktor $O(n)$

❖ Lze podobnou transformaci udělat pro jazyk L_{2SAT} (každá klauzule má 2 literály)?

Problém 3SAT je NP-úplný

- klauzuli $c = x_1 \vee \ldots \vee x_n$ nahradíme klauzulemi

$$(x_1 \vee x_2 \vee y_1) \quad (\neg y_1 \vee x_3 \vee y_2) \quad (\neg y_2 \vee x_4 \vee y_3) \ \ldots \ (\neg y_{n-3} \vee x_{n-1} \vee x_n)$$

- transformace zachovává splnitelnost
- každá nová klauzule zmenší původní klauzuli aspoň o jeden literál.
- popsaná transformace jednou projde původní formuli – pracuje v čase $O(n)$
- formule se zvetší pouze o faktor $O(n)$

❖ Lze podobnou transformaci udělat pro jazyk L_{2SAT} (každá klauzule má 2 literály)?

❖ Nelze, jelikož by jsme nemohli zkracovat původní klauzule. Na democvičení si ukážeme, že $L_{2SAT} \in \mathbf{P}$.

Problém KLIKA je NP-úplný

❖ Nechť $G = (V, E)$ je neorientovaný graf. G má k-kliku pokud obsahuje úplný podgraf (existuje hrana mezi každou dvojicí vrcholů), který má k vrcholů.

$$L_{KLIKA} = \{< G, k >| \; G \text{ má } k\text{-kliku}\}$$

Problém KLIKA je NP-úplný

❖ Nechť $G = (V, E)$ je neorientovaný graf. G má k-kliku pokud obsahuje úplný podgraf (existuje hrana mezi každou dvojicí vrcholů), který má k vrcholů.

$$L_{KLIKA} = \{< G, k >| \; G \text{ má } k\text{-kliku}\}$$

❖ $L_{KLIKA} \in \mathbf{NP}$: NTS uhádne k vrcholů a pak ověří, jestli tvoří úplný podgraf – složitost ověření záleží na reprezentaci grafu, ale očividně je v $\mathbf{P}$ vzhledem k velikost G.

Problém KLIKA je NP-úplný

❖ Nechť $G = (V, E)$ je neorientovaný graf. G má k-kliku pokud obsahuje úplný podgraf (existuje hrana mezi každou dvojicí vrcholů), který má k vrcholů.

$$L_{KLIKA} = \{< G, k >| \; G \text{ má } k\text{-kliku}\}$$

❖ $L_{3SAT} \leq_P^m L_{KLIKA}$

❖ Hlavní myšlenka: Splnitelnost formule s k klauzulemi redukujeme na problém k-kliky v grafu G, kde

- každý literál v každé klauzuli jednoznačně mapujeme na unikátní vrchol grafu
- hrany definujeme mezi "nekonfliktními" literály z různých klauzulí

Problém KLIKA je NP-úplný

❖ $L_{3SAT} \leq_P^m L_{KLIKA}$

❖ Formálně: Mějme formuli $F = C_1 \wedge C_2 \wedge \ldots \wedge C_k$ kde $\forall 1 \leq i \leq k : C_i = L_i^1 \vee L_i^2 \vee L_i^3$ a $\forall 1 \leq j \leq 3 : L_i^j$ je buď proměnná nebo její negace.

❖ Pro F sestrojíme neorientovaný graf $G_F = (V_F, E_F)$, kde

$$V_F = \{L_i^j \mid 1 \leq i \leq k \wedge 1 \leq j \leq 3\} \quad \text{a} \quad E_F = \{\{L_i^j, L_{i'}^{j'}\} \mid i \neq i' \wedge L_i^j \neq \neg L_{i'}^{j'}\}$$

❖ Je snadno vidět, že $|V_F|$ je dána velikostí F ($|E_F| < |V_F|^2$) a tudíž G_F umíme zkonstruovat v polynomiálním čase (existenci hrany mezi dvěma vrcholy ověříme v lineárním čase vzhledem k velikosti F).

❖ Korektnost: Dokážeme, že F je splnitelná $\iff$ G_F má k-kliku.

Problém KLIKA je NP-úplný

- Pro F sestrojíme $G_F = (V_F, E_F)$ kde

$$V_F = \{L_i^j \mid 1 \leq i \leq k \wedge 1 \leq j \leq 3\} \qquad E_F = \{\{L_i^j, L_{i'}^{j'}\} \mid i \neq i' \wedge L_i^j \neq \neg L_{i'}^{j'}\}$$

- Korektnost: Dokážeme, že F je splnitelná $\Longleftrightarrow$ G_F má k-kliku.

  $\Rightarrow$ Je-li F je splnitelná, tak pro každou klauzuli C_i $(1 \leq i \leq k)$ lze vybrat jeden literál L_i^j $(1 \leq j \leq 3)$ tak, že žádné dva zvolené literály L_i^j a $L_{i'}^{j'}$ nejsou vzájemně v konfliktu, tj. $L_i^j \neq \neg L_{i'}^{j'}$. Těmto k literálům odpovídá k vrcholů v G_F, které dle definice E_F tvoří úplný podgraf.

  $\Leftarrow$ Má-li G k-kliku, tak existuje v každé klauzuli literál (tj. k literálů), který není v konfliktu s žádným jiným z těchto k literálů. Z těchto literálů lze přímočaře sestrojit splnitelné přiřazení pro formuli F.

Zkouškový příklad na redukci

Uvažme jazyk

$L_{NE} = \{(\Phi_1, \Phi_2) \mid \Phi_1, \Phi_2$ jsou výrokové formule v konjunktivní normální formě,

pro které existuje valuace proměnných $\bar{v}$ taková, že $\Phi_1(\bar{v}) \neq \Phi_2(\bar{v})\}$.

Nyní ukážeme, že L_{NE} je $\mathbf{NP}$-těžký což nám s důkazem z minulé přednášky dá $\mathbf{NP}$-úplnost.

Zkouškový příklad na redukci

Uvažme jazyk

$$L_{NE} = \{(\Phi_1, \Phi_2) \mid \Phi_1, \Phi_2 \text{ jsou výrokové formule v konjunktivní normální formě,}$$
$$\text{pro které existuje valuace proměnných } \bar{v} \text{ taková, že } \Phi_1(\bar{v}) \neq \Phi_2(\bar{v})\}.$$

Nyní ukážeme, že L_{NE} je $\mathbf{NP}$-těžký což nám s předchozím důkazem dá $\mathbf{NP}$-úplnost.

- Ukážeme, že $L_{SAT} \leq_P^m L_{NE}$. Bez ztráty na obecnosti uvažme, že

  $$L_{SAT} = \{\Phi \mid \Phi \text{ je výroková formule v konjunktivní normální formě,}$$
  $$\text{pro kterou existuje valuace proměnných } \bar{v} \text{ taková, že } \Phi(\bar{v}) = \text{true}\}$$

- Sestrojíme funkci f (vyčíslitelnou DTS v polynomiálním čase), pro kterou platí $f(\Phi) = (\Phi_1, \Phi_2)$ a $\Phi \in L_{SAT} \iff (\Phi_1, \Phi_2) \in L_{NE}$.
- Stačí položit $\Phi_1 \equiv \Phi$ a $\Phi_2 \equiv x_1 \wedge \overline{x_1}$ (x_1 je proměnná a tudíž Φ_2 je kontradikce).

$$\Phi \in L_{SAT} \Rightarrow \exists \bar{v} : \Phi_1(\bar{v}) = \text{true} \wedge \Phi_2(\bar{v}) = \text{false} \Rightarrow (\Phi_1, \Phi_2) \in L_{NE}$$

$$\Phi \notin L_{SAT} \Rightarrow \forall \bar{v} : \Phi_1(\bar{v}) = \text{false} = \Phi_2(\bar{v}) \Rightarrow (\Phi_1, \Phi_2) \notin L_{NE}$$