Základy složitosti a vyčíslitelnosti

NTIN090

Petr Kučera

2024/25 (10. přednáška)

Dnešní plán

- Vlastnosti třídy $\#\mathrm{P}$
- $\#\mathrm{P}$-úplnost
- Zjemnělá složitost
- Hypotézy o exponenciálním čase
- Převod k-SAT na problém ortogonality vektorů
- Převod ortogonality vektorů na shodu s regulárním výrazem

Třída $\#\mathrm{P}$
(2. část)

Počítání modelů

#SAT

Instance: Formule φ v KNF

Hodnota: Počet modelů φ

Pokud $\#\mathrm{SAT}$ lze vyčíslit v polynomiálním čase, pak $\mathrm{P} = \mathrm{NP}$

- Polynomiální algoritmus pro $\#\mathrm{SAT}$ bychom mohli použít k rozhodnutí SAT

$$\varphi \text{ je splnitelná} \iff \#\mathrm{SAT}(\varphi) > 0$$

- Těžkost $\#\mathrm{SAT}$ je způsobená těžkostí SAT

Počítání cyklů

#CYCLE

Instance: Orientovaný graf $G = (V, E)$

Hodnota: Počet cyklů G

- Snadné ověřit, zda G obsahuje cyklus
- Těžké určit počet cyklů v G

Věta

Pokud #CYCLE *lze vyčíslit v polynomiálním čase, pak* $\mathrm{P} = \mathrm{NP}$

- Ukážeme, že algoritmus vyčíslující #CYCLE lze použít k rozhodnutí problému Hamiltonovské kružnice
- Problém Hamiltonovské kružnice NP-úplný

Třída #P

Definice

Funkce $f : \Sigma^* \to \mathbb{N}$ patří do třídy #P, pokud existuje polynomiální verifikátor V a polynom $p(n)$ takové, že pro každý $x \in \Sigma^*$ platí

$$f(x) = |\{y \in \{0,1\}^{p(|x|)} \mid V(x,y) \text{ přijme}\}|.$$

Alternativně

$$f(x) = \text{počet přijímajících výpočtů } M(x)$$

pro nějaký NTS M, který pracuje v polynomiálním čase.

Třídy NP a #P

Třída NP ptáme se po existenci polynomiálního certifikátu

- Má úloha řešení?

Třída #P zajímá nás počet polynomiálních certifikátů

- Kolik řešení úloha má?

Počítání certifikátů může být těžší než hledání jednoho

Příklad

Uvažme rozdíl mezi

- ověřováním acykličnosti orientovaného grafu (lehké) a
- počítáním cyklů v orientovaného grafu (těžké)

Pozitivita funkce f

Pozitivita f

Instance: $x \in \Sigma^*$

Otázka: $f(x) > 0$?

Věta

Je-li $f \in \#\mathrm{P}$, pak Pozitivita f patří do NP.

- Nechť V je polynomiální verifikátor a $p(n)$ polynom, které splňují

$$f(x) = |\{y \in \{0,1\}^{p(|x|)} \mid V(x,y) \text{ přijme}\}|$$

- Pak V je polynomiální verifikátor pro Pozitivita f
- Pozitivita f tedy patří do NP.

Počítání pomocí rozhodování

Věta

Pro funkci $f \in \#\mathrm{P}$ je otázka náležení do množiny $S_f = \{(x, N) \mid f(x) \geq N\}$ výpočetně ekvivalentní (až na polynomiální faktor) výpočtu f.

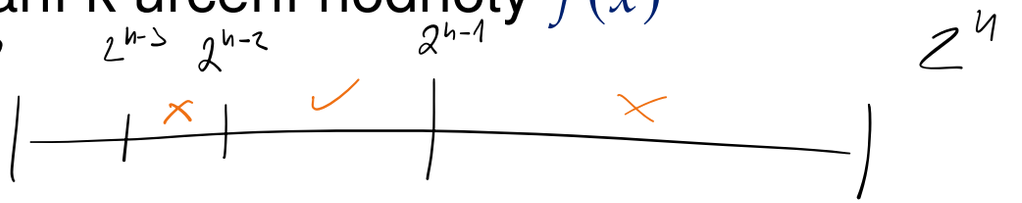
1. $(x, N) \in S_f$ lze jednoduše rozhodnout se znalostí hodnoty $f(x)$
2. $f(x)$ lze určit pomocí polynomiálního počtu dotazů typu „$(x, N) \in S_f$?“
   - Nechť V je polynomiální verifikátor a $p(n)$ polynom, které splňují

$$f(x) = |\{y \in \{0,1\}^{p(|x|)} \mid V(x, y) \text{ přijme}\}|$$

   - Tedy $0 \leq f(x) \leq 2^{p(|x|)}$
   - Použijeme binární vyhledávání k určení hodnoty $f(x)$
   - Stačí $O(p(|x|))$ dotazů

potřebujeme tolik bitů k reprezentaci

→ jinak bychom musel provádět 2^n kroků výpočtu



mám n proměnných →

Prostorová složitost $\#\mathrm{P}$

Věta

Hodnotu $f \in \#\mathrm{P}$ lze vyčíslit v polynomiálním prostoru

- Nechť V je polynomiální verifikátor a $p(n)$ polynom, které splňují

$$f(x) = |\{y \in \{0,1\}^{p(|x|)} \mid V(x,y) \text{ přijme}\}|$$

- K určení hodnoty $f(x)$ stačí pustit $V(x,y)$ pro každý řetězec $y \in \{0,1\}^{p(|x|)}$
- $f(x)$ je pak počet přijatých certifikátů

Polynomiální převoditelnost funkcí

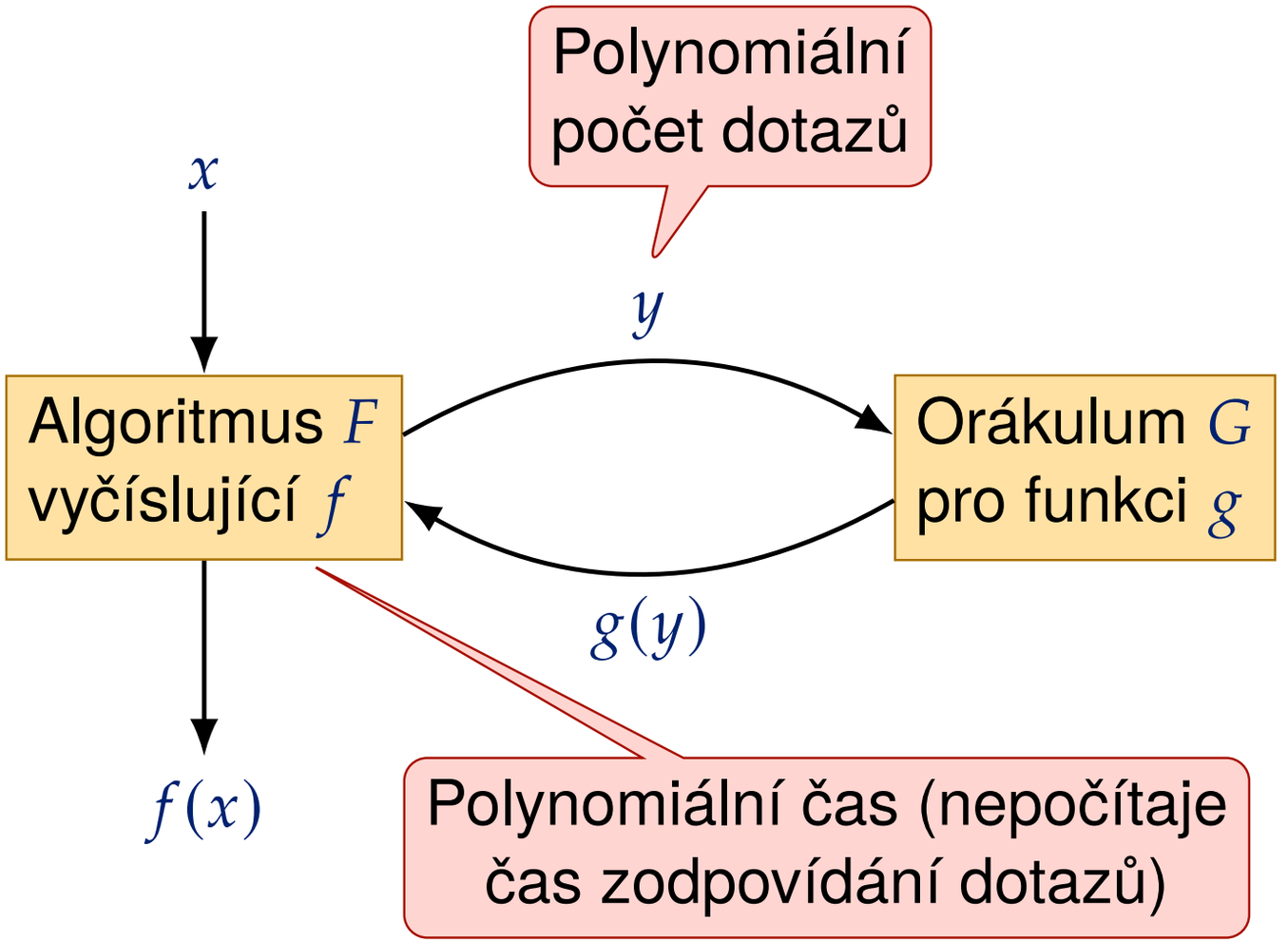
Definice

Funkce f je polynomiálně převoditelná na funkci g ($f \leq_P g$), pokud lze f vyčíslit v polynomiálním čase algoritmem, který má přístup k orákulu funkce g.

- Algoritmus vyčíslující $f(x)$
  - Má přístup k orákulu, které určí $g(y)$ pro libovolný řetězce y
  - Orákulu může položit polynomiální počet dotazů
  - Algoritmus pracuje v polynomiálním čase, za předpokladu, že orákulum odpovídá okamžitě

Převoditelnost funkcí (princip)

$f \leq_P g$ = „pomocí g umíme spočítat f v polynomiálním čase“



$\#\mathrm{P}$-úplnost

Definice

Funkce $g \in \#\mathrm{P}$ je $\#\mathrm{P}$-úplná, pokud $f \leq_P g$ pro každou funkci $f \in \#\mathrm{P}$.

Věta

Funkce $\#\mathrm{SAT}$ je $\#\mathrm{P}$-úplná.

- Plyne z převodu, kterým jsme ukazovali NP-úplnost SAT
- Podívejme se na to podrobněji

#P-úplnost #SAT

- Uvažme $f \in \#\mathrm{P}$
- Nechť V je polynomiální verifikátor a $p(n)$ polynom, které splňují

$$f(x) = |\{y \in \{0,1\}^{p(|x|)} \mid V(x,y) \text{ přijme}\}|$$

- Uvažme následující nedeterministický TS M

Výpočet M se vstupem x

---

1. Nedeterministicky vyber $y \in \{0,1\}^{p(|x|)}$
2. Pusť $V(x,y)$
3. **if** $V(x,y)$ přijal **then** přijmi **else** odmítni

$f(x) =$ počet přijímajících výpočtů $M(x)$

Vyčíslení f pomocí #SAT

- V důkazu Cookovy-Levinovy věty jsme popsali konstrukci KNF φ, která s každým hodnocením $\mathbf{a}$ splňuje

$$\varphi(\mathbf{a}) = 1 \iff \mathbf{a} \text{ reprezentuje přijímající výpočet } M(x)$$

- Navíc platí, že mezi modely $\mathbf{a}$ formule φ a přijímajícími výpočty $M(x)$ je bijekce
- Platí tedy $f(x) = \#\mathrm{SAT}(\varphi)$
- Následující algoritmus vyčísluje $f(x)$:
  1. V polynomiálním čase zkonstruuj φ
  2. Vrať $\#\mathrm{SAT}(\varphi)$

$$f \leq_P \#\mathrm{SAT}$$

Parsimonious převod

Definice

Polynomiální převod z problému A na problém B je parsimonious, pokud zachovává počet certifikátů.

- Je-li A převoditelný na B parsimonious převodem, pak $\#A \leq_P \#B$
- Převod z $A \in \mathrm{NP}$ do SAT, který jsme popsali v důkazu Cookovy-Levinovy věty, je parsimonious
- Funkce $\#\mathrm{SAT}$ je proto $\#\mathrm{P}$-úplná

Disjunktivní normální forma

Literál výroková proměnná x nebo její negace $\neg x$

Term konjunkce literálů

- Například $x \wedge \neg y \wedge \neg z$
- Prázdný term je splněný ($\top$)

DNF formule je v disjunktivní normální formě, pokud jde o disjunkci termů

Příklad

Následující formule je v DNF

$$\psi = x \vee (\neg y \wedge z) \vee (\neg x \wedge y) \vee (\neg x \wedge \neg y \wedge \neg z) \vee (x \wedge \neg z)$$

Splnitelnost DNF

- Splnitelnost DNF ψ je možné ověřit v polynomiálním čase
  - Ověříme, zda je splnitelný jeden z termů ψ
  - Term T je splnitelný $\iff$ T neobsahuje $x \wedge \neg x$ pro žádnou proměnnou x
- Pro danou KNF φ lze jednoduše zkonstruovat DNF $\psi \equiv \neg\varphi$
  - Použijeme De Morganova pravidla

$$\neg(a \wedge b) \equiv \neg a \vee \neg b \quad \text{and} \quad \neg(a \vee b) \equiv \neg a \wedge \neg b$$

Uvažme KNF

$$\varphi = \neg x \wedge (y \vee \neg z) \wedge (x \vee \neg y) \wedge (x \vee y \vee z) \wedge (\neg x \vee z)$$

Aplikací De Morganových pravidel na $\neg\varphi$ dostaneme DNF $\psi \equiv \neg\varphi$

$$\psi = x \vee (\neg y \wedge z) \vee (\neg x \wedge y) \vee (\neg x \wedge \neg y \wedge \neg z) \vee (x \wedge \neg z)$$

Počítání modelů DNF

- Definujeme funkci $\#\,\text{DNF-SAT}$, která pro každou DNF ψ splňuje

$$\#\,\text{DNF-SAT}(\psi) = |\{\mathbf{a} \mid \psi(\mathbf{a}) = 1\}|$$

- $\#\,\text{DNF-SAT}$ patří do $\#\text{P}$

Věta

$\#\text{SAT} \leq_P \#\,\text{DNF-SAT}$, *tedy funkce* $\#\,\text{DNF-SAT}$ *je* $\#\text{P}$*-úplná.*

Výpočet $\#\text{SAT}(\varphi)$ s pomocí $\#\,\text{DNF-SAT}$

Vstup: KNF φ

1 $n \leftarrow$ počet proměnných v φ
2 $\psi \leftarrow$ DNF ekvivalentní $\neg\varphi$
3 **return** $2^n - \#\,\text{DNF-SAT}(\psi)$

Počet perfektních párování v bipartitním grafu

Perfektní párování v bipartitním grafu (BPM)

Instance: Bipartitní graf $G = (V = A \cup B, E \subseteq A \times B)$, kde $|A| = |B|$.

Otázka: Existuje v G párování velikosti $|A| = |B|$?

Věta (Bez důkazu)

Funkce $\#\,\mathrm{BPM}$ *je* $\#\mathrm{P}$*-úplná.*

Permanent matice

Definice

Je-li A matice typu $n \times n$ definujeme permanent A jako

$$\mathrm{perm}(A) = \sum_{\pi \in S(n)} \prod_{i=1}^{n} a_{i,\pi(i)},$$

kde $S(n)$ je množina permutací množiny $\{1, \ldots, n\}$.

- „Determinant“, kde neuvažujeme znaménko permutace.
- Je-li A matice sousednosti bipartitního grafu G, pak $\mathrm{perm}(A)$ určuje počet perfektních párování G.

Věta (Bez důkazu)

Funkce perm *je* $\#\mathrm{P}$*-úplná.*

Zjemnělá složitost

Složitost uvnitř P

- NP-úplnost je příliš hrubá pro studium složitosti problémů uvnitř třídy P
- Rozdíl mezi $O(n)$ a $O(n^5)$
- Někdy je i kvadratický čas příliš
  - velká data (big data)
- Zajímají nás dolní odhady složitosti problémů řešitelných v polynomiálním čase
- Netriviální nepodmíněné dolní odhady je velmi těžké dokázat
  - je-li to vůbec možné

fakticky už neexistuje efektivní alg.

To ale nejde s jistotou nikdy říct. RAM model do toho háží vidle.

Zjemnělá složitost

Podmíněné dolní odhady

- Dolní odhad složitosti je založen na široce přijímané hypotéze
- Používáme zjemnělou převoditelnost

Výpočetní model

RAM s logaritmickou cenou operací

Klíčové hypotézy

k-SAT

k-KNF klauzule obsahují nejvýš k literálů

k-SAT

Instance: Formule φ v k-KNF na n proměnných.

Otázka: Je φ splnitelná?

- Lze vyřešit v čase $O(2^n n^k)$
- Pro každé $k \in \mathbb{N}$ definujeme

$$s_k = \inf \{\delta \mid k\text{-SAT lze vyřešit v čase } O(2^{\delta n} \operatorname{poly}(n))\}$$

Hypotézy o exponenciálním čase

$$s_k = \inf \{\delta \mid k\text{-SAT lze vyřešit v čase } O(2^{\delta n} \operatorname{poly}(n))\}$$

Hypotéza o exponenciálním čase (ETH)

- Pro nějaké k je $s_k > 0$ → tohle říkám: SAT nejrůstím lépe než v $2^n \cdot$ polynom → tohle říkám že ta exponenciála hnije volí ☺
- Ekvivalentně $s_3 > 0$

Silná hypotéza o exponenciálním čase (SETH)

$\lim_{k \to \infty} s_k = 1$ → nakonec stejně ten problém bude těžce exponenciální.

A zároveň to je hypotéza, kterou se snaží dokázat, že jsou alg., které nejde řešit více efektivně.

! Závěr mezi polynomy a exponenciálou je ještě spousta funkcí, proto všechno co hned není polynomiální neznamená, že to je stejně složitý jak ostatní exp. problémy.

3-SAT lze řešit v $O((\frac{4}{3})^n)$

Ortogonální vektory

Ortogonální vektory (OV)

Instance: Množiny A a B obsahující po n vektorech z $\{0,1\}^d$.

Otázka: Existují navzájem ortogonální vektory $a \in A$ a $b \in B$ (tj. $a[i] \cdot b[i] = 0$ pro všechna $i = 1, \ldots, d$)?

- Lze vyřešit v čase $O(n^2 d)$ vypočtením skalárních součinů všech dvojic vektorů

↳ tohle je direct. přístup

Hypotéza o ortogonálních vektorech (OV)

OV nelze vyřešit v čase $O(n^{2-\delta} d^c)$ pro žádné $\delta, c > 0$.

Nejkratší cesty mezi všemi dvojicemi vrcholů

Nejkratší cesty mezi všemi dvojicemi vrcholů (APSP)

Instance: Orientovaný graf G s n vrcholy a kladnými délkami hran.

Cíl: Pro každou dvojici vrcholů určit délku nejkratší cesty, která je spojuje.

- Lze vyřešit v čase $O(n^3)$ Floydovým-Warshallovým algoritmem

Hypotéza o nejkratších cestách mezi všemi dvojicemi vrcholů

APSP nelze vyřešit v čase $O(n^{3-\delta})$ pro žádné $\delta > 0$.

3SUM

3SUM

Instance: Množina X s n celými čísly

Otázka: Platí $a + b + c = 0$ pro nějakou trojici $a, b, c \in X$?

- Lze vyřešit v čase $O(n^2)$ → $\forall a, b$ si zahešuju $(a+b)$, pak se jen ptám, zdali $\exists c : a+b = -c$

Hypotéza o 3SUM

3SUM nelze vyřešit v čase $O(n^{2-\delta})$ pro žádné $\delta > 0$.

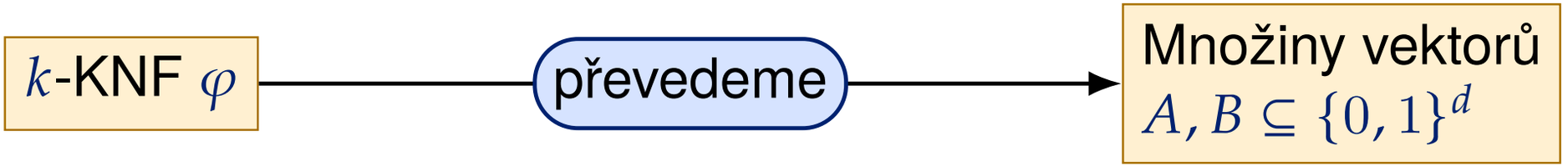
SETH implikuje OV

SETH implikuje OV

Věta

Silná hypotéza o exponenciálním čase implikuje hypotézu o ortogonálních vektorech.

- Popíšeme převod k-SAT na problém Ortogonálních vektorů



φ je splnitelná $\iff$ Nějaké $a \in A$, $b \in B$ jsou ortogonální

Navíc: Zrychlení OV implikuje zrychlení k-SAT.

Převod k-SAT na OV

- Předpokládejme k-KNF

$$\varphi = C_1 \wedge C_2 \wedge \cdots \wedge C_m$$

- n proměnných $X = \{x_1, \ldots, x_n\}$
- Proměnné rozdělíme do dvou částí (předpokládáme sudé n)

$$X_1 = \{x_1, \ldots, x_{n/2}\}$$
$$X_2 = \{x_1, \ldots, x_{n/2}\}$$

$x_{n/2}$ x_n

Množina A

- Předpokládejme ohodnocení $\alpha : X_1 \to \{0, 1\}$
- Definujeme vektor délky $d = m$

$$a^\alpha[j] = \begin{cases} 0 & C_i(\alpha) = 1 \\ 1 & \text{jinak} \end{cases}$$

$$A = \{a^\alpha \mid \alpha : X_1 \to \{0, 1\}\}$$

- Velikost množiny A je $N = 2^{n/2}$

1 2 3 4

$$(X_1 \vee X_2 \vee \neg X_4) \wedge (\neg X_1 \vee \neg X_2) \wedge (\neg X_3 \vee \neg X_4) \wedge (X_2 \vee \neg X_3 \vee X_4)$$

X_1	X_2	1	2	3	4
0	0	1	0	1	0
0	1	0	0	1	0
1	0	0	0	1	1
1	1	1	1	1	0

první polovina proměnných

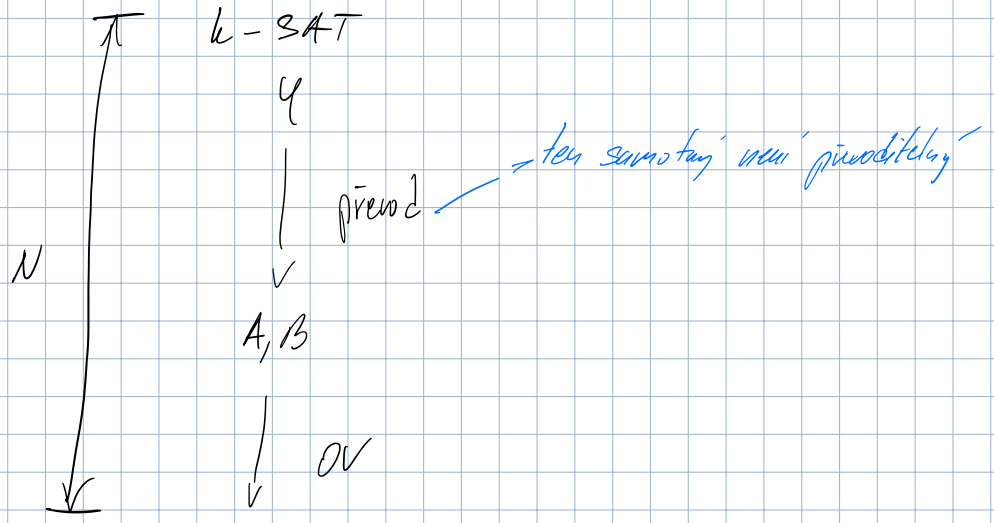
X_3	X_4	1	2	3	4
0	0	0	1	0	0
0	1	1	1	0	0
1	0	0	1	1	1
1	1	1	1	0	0

druhá polovina proměnných

m

$2^{n/2} = N$

Složením těch dvou dostanu $\overline{0}$, tedy je sh. součin roven 0.



„ten samotný není původitelný"

Množina B

- Předpokládejme ohodnocení $\beta : X_2 \to \{0, 1\}$
- Definujeme vektor délky $d = m$

$$b^\beta[j] = \begin{cases} 0 & C_i(\beta) = 1 \\ 1 & \text{jinak} \end{cases}$$

$$B = \left\{ b^\beta \mid \beta : X_2 \to \{0, 1\} \right\}$$

- Velikost množiny B je $N = 2^{n/2}$

Korektnost převodu

Formule φ je splnitelná, právě když existuje dvojice ortogonálních vektorů $a^\alpha \in A$ a $b^\beta \in B$.

- Pro dvojici vektorů $a^\alpha \in A$, $b^\beta \in B$ platí

$$a^\alpha \cdot b^\beta = 0 \iff \varphi(\alpha \circ \beta) = 1$$

Konstrukci lze provést v čase $O(2^{n/2} n^k)$.

- Platí $m = O(n^k)$
- Délka formule φ je tedy $O(kn^k) = O(n^k)$

Zrychlení OV implikuje zrychlení k-SAT

- Předpokládejme algoritmus pro OV se složitostí $O(N^{2-\delta}d^c)$ pro nějaké konstanty $\delta, c > 0$
- Platí $d = m = O(n^k)$

$\Longrightarrow$ $d^c = O(n^{kc}) = o(2^{n(\delta/10)})$ pro každé k

- Platí $N = 2^{n/2}$, tedy

$$O(N^{2-\delta}d^c) = O(2^{n(1-\delta/2)}2^{n(\delta/10)}) = O(2^{n(1-\delta/2+\delta/10)}) = O(2^{n(1-\delta')})$$

pro vhodné $\delta' > 0$

- Čas konstrukce: $O(2^{n/2}n^k) = O(2^{n(1-\delta'')})$ pro vhodné $\delta'' > 0$
- Pro vhodné $\gamma > 0$ pro každé k platí $s_k \leq 1 - \gamma$

Spor s SETH ($\lim_{k\to\infty} s_k = 1$)

Shoda s regulárním výrazem

Regulární výrazy

- Omezíme se na velmi jednoduché regulární výrazy
- Regulární výraz R reprezentuje množinu řetězců $L(R)$

Definice

Předpokládejme abecedu Σ.

1. Pro každý znak $c \in \Sigma$ je $R = c$ regulární výraz s $L(R) = \{c\}$
2. Jsou-li R_1, R_2 regulární výrazy, pak
   1. $R = R_1|R_2$ je regulární výraz s
   $$L(R) = L(R_1) \cup L(R_2)$$
   2. $R = R_1 \cdot R_2$ je regulární výraz s
   $$L(R) = L(R_1) \cdot L(R_2) = \{uv \mid u \in L(R_1) \wedge v \in L(R_2)\}$$

Shoda podřetězce s regulárním výrazem

Shoda podřetězce s regulárním výrazem (REG)

Instance: Regulární výraz R a text $T \in \Sigma^*$.

Otázka: Obsahuje T podřetězec $t \in L(R)$?

- Lze vyřešit v čase $O(nm)$, kde $n = |T|$ a $m = |R|$

Věta

Problém shody podřetězce s regulárním výrazem nelze vyřešit v čase $O((n+m)^{2-\varepsilon})$ pro žádné $\varepsilon > 0$, pokud platí hypotéza o ortogonálních vektorech.

Převod OV na REG

- Předpokládejme instanci OV
  - Množiny vektorů délky d

$$A = \{a_1, \ldots, a_n\}$$
$$B = \{b_1, \ldots, b_n\}$$

- Popíšeme konstrukci řetězce T a regulárního výrazu R, pro které platí:
  - $|T| = O(nd)$
  - $|R| = O(nd)$
  - T obsahuje podřetězec $t \in L(R)$, právě když A, B obsahují dvojici ortogonálních vektorů
  - Konstrukci lze provést v čase $O(nd)$

Rychlejší algoritmus pro REG lze použít pro rychlejší rozhodnutí OV.

Konstrukce textu a regulárního výrazu

- Pro hodnotu $v \in \{0, 1\}$ definujeme regulární výraz

$$C(v) = \begin{cases} 0 & v = 1 \\ (0|1) & v = 0 \end{cases}$$

- Pro vektor $a \in A$ definujeme regulární výraz

$$V(a) = C(a[0])C(a[1]) \dots C(a[d])$$

- Regulární výraz R definujeme takto

$$R = V(a_1)\#V(a_2)\# \dots \#V(a_n)$$

- Text T definujeme takto

$$T = b_1\#b_2\# \dots \#b_n$$

Zjemnělá převoditelnost

Zjemnělá převoditelnost

Pro jednoduchost uvažujeme jen many-one převoditelnost.

Definice

Uvažme rozhodovací problémy A a B a funkce časové složitosti t_A a t_B. Zjemnělým převodem (A, t_A) na (B, t_B) míníme funkci $f : \Sigma^* \to \Sigma^*$ splňující následující vlastnosti:

1. Pro každý řetězec $x \in \Sigma^*$ platí $x \in A \iff f(x) \in B$
2. Pro každé $\varepsilon > 0$ existuje $\delta > 0$, pro něž platí

$$t_B(|f(x)|)^{1-\varepsilon} = O(t_A(|x|)^{1-\delta})$$

3. $f(x)$ je vyčíslitelná v čase $O(t_A(|x|)^{1-\gamma})$ pro nějaké $\gamma > 0$

Použití zjemnělé převoditelnosti

- Mějme zjemnělý převod z (A, t_A) do (B, t_B)
- Předpokládejme algoritmus rozhodující B v čase $O(t_B(n)^{1-\varepsilon})$ pro nějaké $\varepsilon > 0$
- Pak otázku, zda $x \in A$, umíme rozhodnout v čase

$$\begin{aligned} & O(t_A(|x|)^{1-\gamma} + t_B(|f(x)|)^{1-\varepsilon}) \\ &= O(t_A(|x|)^{1-\gamma} + t_A(|x|)^{1-\delta}) \\ &= O(t_A(|x|)^{1-\delta'}) \end{aligned}$$

for some $\delta' > 0$

Zlepšení složitosti pro B implikuje zlepšení složitosti pro A.