

Základy složitosti a vyčíslitelnosti

NTIN090

Petr Kučera

2022/23 (7. přednáška)

Více zdrojů, více síly

Prostor (důsledky)

Věta (o deterministické prostorové hierarchii)

Pro každou prostorově konstruovatelnou funkci $f : \mathbb{N} \rightarrow \mathbb{N}$ existuje jazyk A , který je rozhodnutelný v prostoru $O(f(n))$, nikoli však v prostoru $o(f(n))$.

Důsledek

Jsou-li $f_1, f_2 : \mathbb{N} \rightarrow \mathbb{N}$ funkce, pro které platí, že $f_1(n) \in o(f_2(n))$ a f_2 je prostorově konstruovatelná, potom

$$\text{SPACE}(f_1(n)) \subsetneq \text{SPACE}(f_2(n))$$

Důsledek

Pro každá dvě reálná čísla $0 \leq \epsilon_1 < \epsilon_2$ platí, že

$$\text{SPACE}(n^{\epsilon_1}) \subsetneq \text{SPACE}(n^{\epsilon_2})$$

- Je-li ϵ_2 racionální číslo, pak
 - n^{ϵ_2} je prostorově konstruovatelná
 - Lze jednoduše ukázat pro přirozená čísla
 - Lze ukázat i pro racionální čísla
 - Ostrá inkluze plyne z prostorové hierarchie
- Je-li ϵ_2 iracionální číslo
 - Racionální čísla jsou hustá v reálných číslech
 - Existuje racionální číslo ϵ splňující $\epsilon_1 < \epsilon < \epsilon_2$
 - Z prostorové hierarchie a prostorové konstruovatelnosti n^ϵ

$$\text{SPACE}(n^{\epsilon_1}) \subsetneq \text{SPACE}(n^\epsilon) \subseteq \text{SPACE}(n^{\epsilon_2})$$

Logaritmický, polynomiální a exponenciální prostor

Důsledek

$$\text{NL} \subsetneq \text{PSPACE} \subsetneq \text{EXPSPACE} = \bigcup_{k \in \mathbb{N}} \text{SPACE}(2^{n^k}).$$

Definice

Savičova věta

Prostorová hierarchie

$$\text{NL} = \text{NSPACE}(\log_2 n) \subseteq \text{SPACE}((\log_2 n)^2) \subsetneq \text{PSPACE} \subsetneq \text{EXPSPACE}$$

Čas

Časová složitost Turingova stroje

Připomenutí

Nechť $f : \mathbb{N} \rightarrow \mathbb{N}$ je funkce, která je definovaná pro každý vstup

- Deterministický Turingův stroj M **pracuje v čase** $f(n)$, pokud výpočet M nad libovolným vstupem x délky $|x| = n$ skončí po provedení nejvýše $f(n)$ kroků.
- $\text{TIME}(f(n))$ je třída jazyků přijímaných Turingovými stroji, které pracují v čase $O(f(n))$

Věta (Věta o deterministické časové hierarchii)

Pro každou časově konstruovatelnou funkci $f : \mathbb{N} \rightarrow \mathbb{N}$ existuje jazyk A , který je rozhodnutelný v čase $O(f(n))$, nikoli však v čase $o(f(n)/\log_2 f(n))$.

Definice

Funkci $f : \mathbb{N} \rightarrow \mathbb{N}$, kde $f(n) \in \Omega(n \log_2 n)$, nazveme **časově konstruovatelnou**, je-li funkce, která zobrazuje 1^n na binární reprezentaci $f(n)$ vyčíslitelná v čase $O(f(n))$.

- Funkce obvykle používané pro měření časové složitosti jsou časově konstruovatelné, například
 - $\lceil n \log_2 n \rceil$
 - $\lceil n \sqrt{n} \rceil$
 - polynomy
 - 2^n

Efektivní počítání kroků

Předpokládejme, že $f(n)$ je časově konstruovatelná strojem M_f

- 1 Se vstupem x
- 2 Sestav řetězec $w = 1^n$
 - Každý znak x změň na 1
- 3 Vypočítej $k = f(n)$
 - Spusť $M_f(w)$
- 4 Inicializuj binární čítač hodnotou k
 - Používá $\lceil \log_2 k \rceil$ bitů
- 5 Sniž hodnotu čítače o jedna po každém kroku a skonči pokud dosáhne hodnota čítače nuly

Pracuje v čase $O(f(n) \cdot t(\lceil \log_2 k \rceil))$, kde $t(\lceil \log_2 k \rceil)$ je čas potřebný k aktualizaci hodnoty čítače s $\lceil \log_2 k \rceil$ bity.

Věta (Věta o deterministické časové hierarchii)

Pro každou časově konstruovatelnou funkci $f : \mathbb{N} \rightarrow \mathbb{N}$ existuje jazyk A , který je rozhodnutelný v čase $O(f(n))$, nikoli však v čase $o(f(n)/\log_2 f(n))$.

Idea důkazu

- Podobný postup jako v případě prostoru
- Je potřeba simulovat $M(x)$ s počítáním kroků
- Manipulace s čítačem kroků přidává faktor $\Theta(\log_2 f(n))$

Předpokládáme, že všechny stroje mají jednu pásku.

Výpočet D se vstupem x

- 1 $n \leftarrow |x|$
 - 2 Vypočti $f(n)$ pomocí časové konstruovatelnosti
 - 3 Inicializuj binární čítač hodnotou $\lceil f(n)/\log_2 f(n) \rceil$
 - 4 Sniž hodnotu čítače o 1 po každém kroku při provádění kroků 7-8
 - 5 **if** čítač dosáhne nulové hodnoty **then** odmítni
 - 6 **if** x není tvaru $\langle M \rangle 1 0^*$ **then** odmítni
 - 7 Simuluj $M(x)$
 - 8 **if** M přijal **then** odmítni **else** přijmi
-

Definujeme $A = L(D)$

Čas výpočtu D

- $f(n)$ lze vypočítat v čase $O(f(n))$ díky časové konstruovatelnosti
- $\lceil f(n)/\log_2 f(n) \rceil$ (binárně) lze vypočítat v čase $O(f(n))$
- Ověření, zda x má tvar $\langle M \rangle 10^*$ lze provést v čase $O(n) = O(f(n))$

Implementační detaily

- 1 Jak provést simulaci $M(x)$ tak, aby jedna instrukce M byla provedena v c_M krocích simulace
 - kde c_M je konstanta závisající na M
- 2 Jak snížit hodnotu čítače o 1 po každém kroku D v čase $O(\log_2 f(n))$

Simulace s konstantním zpožděním

- Předpokládejme $M = (Q, \Sigma, \delta, q_0, F)$
- Předpokládejme vstup x tvaru $\langle M \rangle 10^*$
- $\langle M \rangle$ kóduje přechodovou funkci δ
- $|\langle M \rangle|$ je konstantní, je-li M zafixovaný
- Při simulaci jednoho kroku $M(x)$
 - D hledá v $\langle M \rangle$ přechod pro aktuální displej
 - Nejprve musí hlava D najít začátek $\langle M \rangle$ na pásce
 - D potřebuje také rychlý přístup k aktuálnímu stavu M

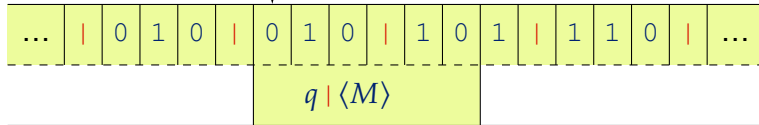
$\langle M \rangle$ a stav stroje M musí být v každém okamžiku poblíž hlavy D .

Páska D

Páska stroje D má dvě stopy

Hlava D nad blokem
pod hlavou M

Stopa 1: páska M
každý znak zakódován
 $b = \lceil \log_2 |\Sigma| \rceil$ bity



Stopa 2:
aktuální stav q a
přechodová funkce $\langle M \rangle$

Stopa 2 je vždy zarovnaná
s blokem pod hlavou M

Simulace s konstantním zpožděním

- Páska D má dvě stopy:

Stopa 1 páska M , každý znak je zakódován $b = \lceil \log_2 |\Sigma| \rceil$ bity

Stopa 2 aktuální stav q stroje M a přechodová funkce $\langle M \rangle$

- Vždy zarovnaná s hlavou M
 - Po odsimulování jednoho kroku M může být potřeba obsah stopy 2 posunout
- Simulace jednoho kroku M pak zabere čas c_M pro nějakou konstantu c_M , která závisí na M
 - Nalezení instrukce v čase $O(|\langle M \rangle|^2)$
 - Posunutí stopy 2 v čase $O(|\langle M \rangle|^2)$
 - Konstantní čas pro fixní TS M

Pokud M pracuje v čase $g(n)$, jeho simulaci lze provést $O(g(n))$.

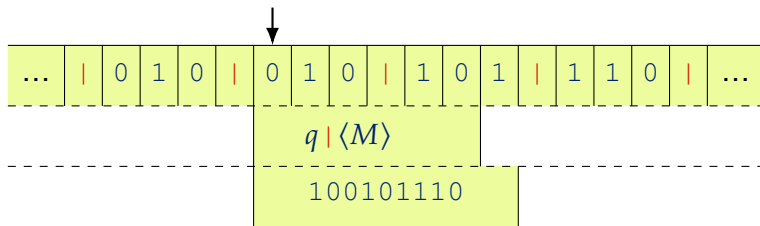
Aktualizace čítače kroků

- Čítač má $O(\log_2 f(n))$ bitů
- Snížení hodnoty o 1 lze provést v lineárním čase vzhledem k počtu bitů
- Nejprve však musí hlava M přejít k čítači na pásce

Čítač musí být neustále poblíž hlavy D

- Přidáme novou stopu pro uložení čítače
- Po každém kroku simulace je stopa posunuta posunuta, aby čítač byl u hlavy D

Třetí stopa pásky stroje D



Stopa 3: Binární čítač
 $O(\log_2 f(n))$ bitů

Zarovnaná s hlavou D

Posouvá se po každém kroku simulace

Zarovnání stop

Stopa 2 obsahuje dvojici $q \mid \langle M \rangle$

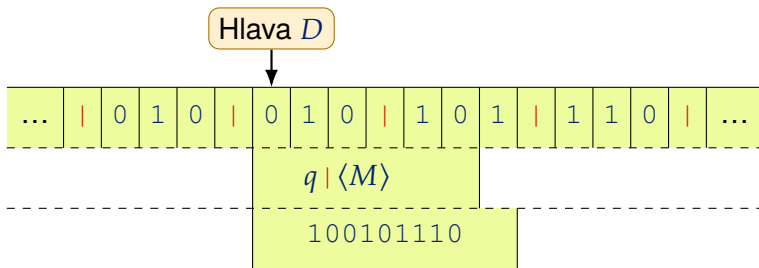
- Na začátku simulace kroku M musí být zarovnaná s blokem pod hlavou M
- Poté, co D dokončí simulaci kroku M , je stopa posunuta podle toho, kam se pohne hlava M

Stopa 3 obsahuje čítač

- Na začátku každého kroku D v rámci simulace musí být zarovnaná s hlavou D
- Jeden krok M je proveden pomocí c_M kroků simulace
- Po každém kroku simulace dojde k posunu čítače
- Čítač se tedy posune c_M -krát během simulace jednoho kroku M

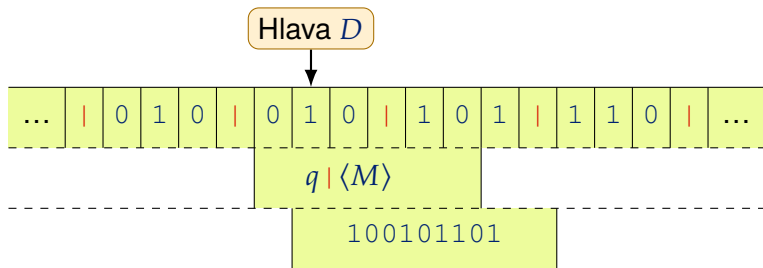
Zarovnání stop

Stopa 3 je zarovnaná s hlavou D při simulaci



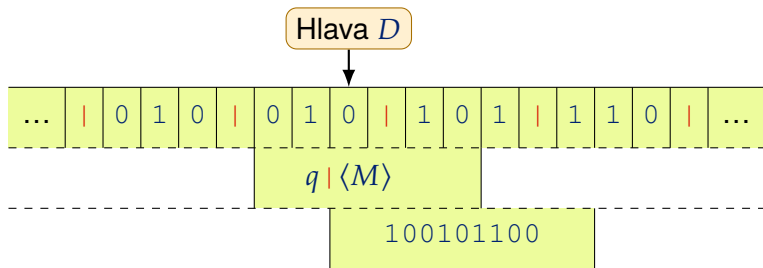
Zarovnání stop

Stopa 3 se posouvá s hlavou D po každém kroku simulace



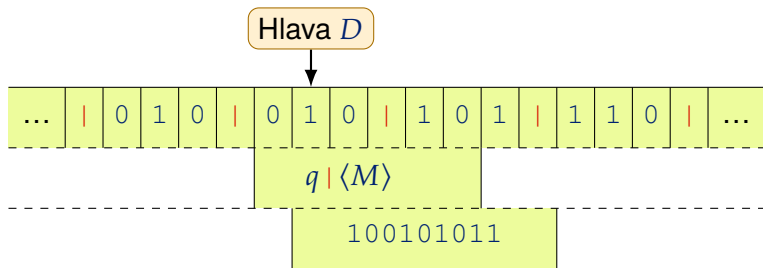
Zarovnání stop

Stopa 3 se posouvá s hlavou D po každém kroku simulace



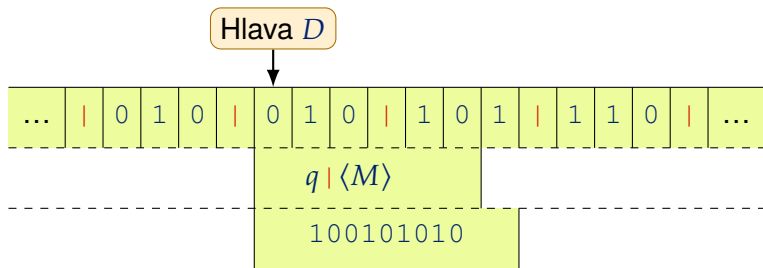
Zarovnání stop

Stopa 3 se posouvá s hlavou D po každém kroku simulace



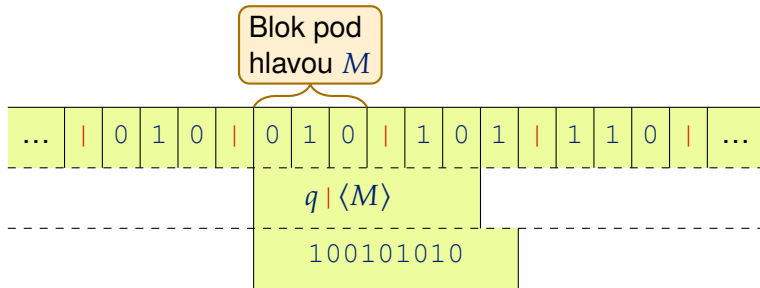
Zarovnání stop

Stopa 3 se posouvá s hlavou D po každém kroku simulace



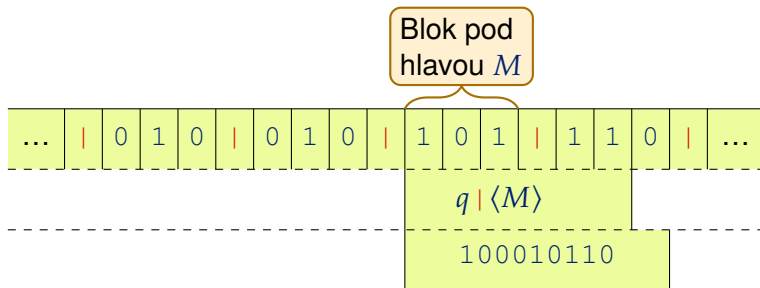
Zarovnání stop

Stopa 2 je zarovnaná s blokem pod hlavou M



Zarovnání stop

Pokud se hlava M pohne, stopa 2 je posunuta



Stopa 3 se posunula spolu s hlavou D

Časová složitost D

- Simulace M je ukončena nejpozději po provedení $\lceil f(n)/\log_2 f(n) \rceil$ kroků simulace
 - D tedy odsimuluje zhruba $\frac{f(n)}{c_M \log_2 f(n)}$ kroků M
- Hodnota čítače se snížší o 1 a případně je posunut po každém kroku simulace
 - $O(\log_2 f(n))$ kroků stačí pro snížení hodnoty
 - $O(\log_2 f(n))$ kroků stačí na posunutí
- Dohromady dostáváme čas

$$O\left(\frac{f(n)}{\log_2 f(n)} \log_2 f(n)\right) = O(f(n))$$

TS D pracuje v čase $O(f(n))$.

Časová složitost rozhodování A (horní odhad)

Jazyk $A = L(D)$ lze rozhodnout v čase $O(f(n))$.

Ukážeme, že A nelze rozhodnout v čase $o(f(n)/\log_2 f(n))$.

Menší čas nestačí

- Nechť $M = (Q, \Sigma, \delta, q_0, F)$ je TS, který pracuje v čase $g(n) = o(f(n)/\log_2 f(n))$
- Ukážeme, že $A \neq L(M)$
- Dříve jsme ukázali, že

Simulaci $M(x)$ lze provést pomocí $c_M g(n)$ kroků, kde c_M je konstanta, jež závisí na M .

Menší čas nestačí

- Z toho, že $g(n) = o(f(n)/\log_2 f(n))$, plyne

$$(\exists n_0 \in \mathbb{N})(\forall n \geq n_0)[c_M g(n) \leq f(n)/\log_2 f(n)]$$

- Předpokládejme vstup $x = \langle M \rangle 10^{n_0}$
- Tedy $|x| > n_0$
- D simuluje M po $f(n)/\log_2 f(n)$ kroků
 - Simulace M se vstupem $x = \langle M \rangle 10^{n_0}$ skončí a
 - $D(x)$ přijme, právě když $M(x)$ odmítne

$$L(D) \neq L(M)$$

Věta (Věta o deterministické časové hierarchii)

Pro každou časově konstruovatelnou funkci $f : \mathbb{N} \rightarrow \mathbb{N}$ existuje jazyk A , který je rozhodnutelný v čase $O(f(n))$, nikoli však v čase $o(f(n)/\log_2 f(n))$.

Důsledek

Jsou-li $f_1, f_2 : \mathbb{N} \rightarrow \mathbb{N}$ funkce, pro které platí, že $f_1(n) \in o(f_2(n)/\log_2 f_2(n))$ a f_2 je časově konstruovatelná, potom

$$\text{TIME}(f_1(n)) \subsetneq \text{TIME}(f_2(n))$$

Důsledek

Pro každá dvě reálná čísla $1 \leq \epsilon_1 < \epsilon_2$,

$$\text{TIME}(n^{\epsilon_1}) \subsetneq \text{TIME}(n^{\epsilon_2})$$

- Je-li ϵ_2 racionální číslo, pak
 - n^{ϵ_2} je časově konstruovatelná
 - Lze jednoduše ukázat pro přirozená čísla
 - Lze ukázat i pro racionální čísla
 - Ostrá inkluze plyne z časové hierarchie
- Je-li ϵ_2 iracionální číslo
 - Racionální čísla jsou hustá v reálných číslech
 - Existuje racionální číslo ϵ splňující $\epsilon_1 < \epsilon < \epsilon_2$
 - Z časové hierarchie a časové konstruovatelnosti n^ϵ

$$\text{TIME}(n^{\epsilon_1}) \subsetneq \text{TIME}(n^\epsilon) \subseteq \text{TIME}(n^{\epsilon_2})$$

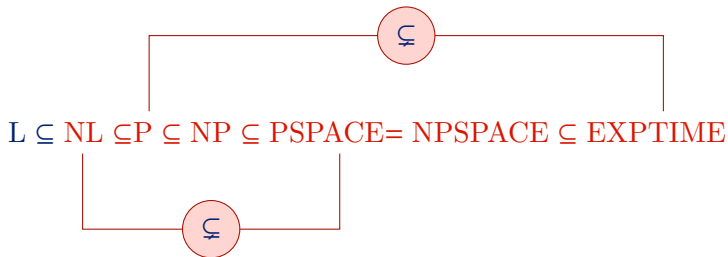
Důsledek

$P \subsetneq \text{EXPTIME}$

- Pro každé $k \in \mathbb{N}$ platí $\text{TIME}(n^k) \subseteq \text{TIME}(2^n)$
- Podle časové hierarchie tedy

$$P \subseteq \text{TIME}(2^n) \subsetneq \text{TIME}(2^{n^2}) \subseteq \text{EXPTIME}$$

Vztahy mezi třídami



- Jedna z inkluzí $NL \subseteq P \subseteq NP \subseteq PSPACE$ musí být ostrá
- Jedna z inkluzí $P \subseteq NP \subseteq PSPACE \subseteq EXPTIME$ musí být ostrá

Nevíme, která z inkluzí je ostrá

Další věty o hierarchii

- Věta o deterministické časové hierarchii lze ukázat i pro k -páskové stroje
 - Faktor $\log_2 f(n)$ je způsoben redukcí počtu pásek z k na 2
- Věty o hierarchiích platí též pro RAM nebo pro nedeterministické třídy složitosti
 - Ve větách o časové hierarchii není třeba faktor $\log_2 f(n)$

Splnitelnost a Coo- kova-Levinova věta

Výroková formule

- Spočetná množina výrokových proměnných $\{x_0, x_1, \dots\}$
- Logické spojky $\{\neg, \wedge, \vee, \rightarrow, \leftrightarrow\}$

Definice (Výroková formule)

- Proměnná je formule
- Jsou-li φ a ψ formule, pak $\neg\varphi$, $\varphi \wedge \psi$, $\varphi \vee \psi$, $\varphi \rightarrow \psi$ a $\varphi \leftrightarrow \psi$ jsou též formule
- Nic jiného nejsou formule

Ohodnocení a splnitelnost

- Ohodnocení \mathbf{a} přiřazuje výrokovým proměnným hodnoty **true/false** ($1/0$, \top/\perp)
- Ohodnocení \mathbf{a} **splňuje formuli** φ pokud se $\varphi(\mathbf{a})$ vyhodnotí na **true**
- Říkáme též, že jde o **model** formule φ
- Formule je **splnitelná**, pokud má model
- V opačném případě je formule **nesplnitelná**

Příklad

Formule

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

je splněná ohodnocením $\{x \mapsto 1, y \mapsto 0, z \mapsto 1\}$

Konjunktivní normální forma

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

Klauzule je disjunkce literálů

- Například $x \vee \neg y \vee \neg z$
- Prázdná klauzule je sporná (false, \perp)

KNF formule je v **konjunktivní normální formě** pokud jde o konjunkci klauzulí

Příklad

Následující formule je v KNF

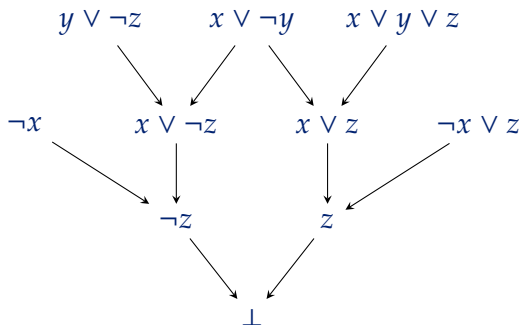
$$\varphi = \neg x \wedge (y \vee \neg z) \wedge (x \vee \neg y) \wedge (x \vee y \vee z) \wedge (\neg x \vee z)$$

Nesplnitelná KNF

Následující formule v KNF je nesplnitelná

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

Nesplnitelnost lze ukázat **rezolucí**



SPLNITELNOST (SAT)

Instance: Formule φ v KNF.

Otázka: Je φ splnitelná?

SAT patří do NP

- V polynomiálním čase lze ověřit, zda dané ohodnocení splňuje danou formuli



Patří SAT do P?

Cookova-Levinova věta

Věta (Cookova-Levinova)

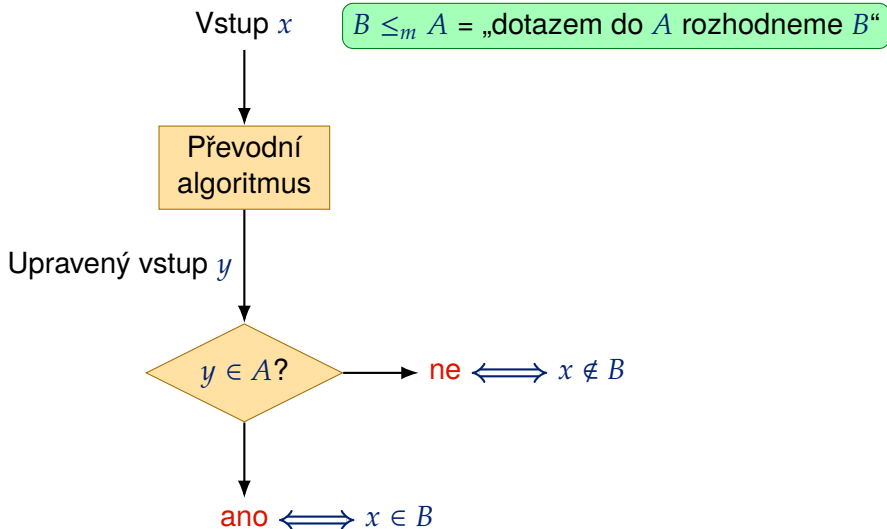
SAT patří do P právě když $P = NP$.

Idea důkazu:

- 1 Zavedeme pojem polynomiální převoditelnosti
 - m -převoditelnost, kde převodní algoritmus pracuje v polynomiálním čase
- 2 Zavedeme pojem NP -úplného problému
 - Nejtěžší problémy v NP vzhledem k polynomiální převoditelnosti
 - Je-li nějaký NP -úplný problém v P , pak $P = NP$
- 3 Ukážeme, že SAT je NP -úplný problém

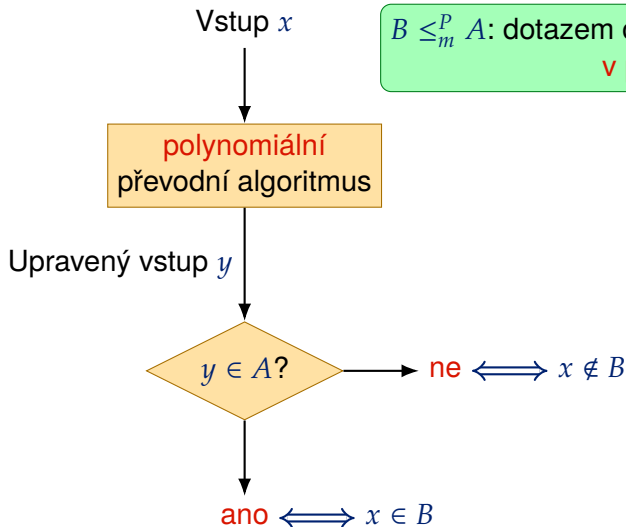
Polynomiální převoditelnost

m -převoditelnost (princip)



Polynomiální převoditelnost (princip)

$B \leq_m^P A$: dotazem do A rozhodneme B
v polynomiálním čase

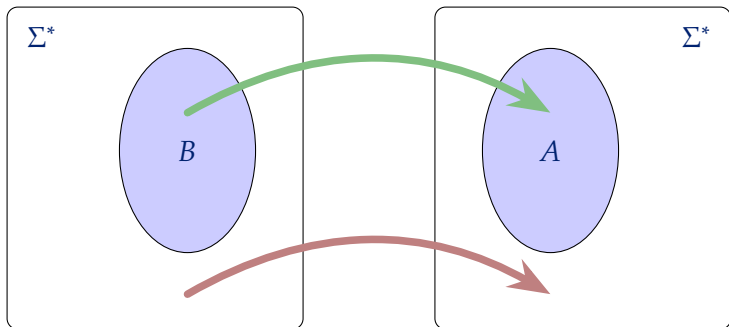


Polynomiální převoditelnost (definice)

Definice

Jazyk B je **polynomiálně převoditelný** na jazyk A ($B \leq_m^P A$), pokud existuje funkce $f : \Sigma^* \rightarrow \Sigma^*$ vyčíslitelná v polynomiálním čase, pro kterou platí

$$(\forall w \in \Sigma^*) [w \in B \iff f(w) \in A]$$



Polynomiální převoditelnost (vlastnosti)

Definice

Jazyk B je **polynomiálně převoditelný** na jazyk A ($B \leq_m^P A$), pokud existuje funkce $f : \Sigma^* \rightarrow \Sigma^*$ vyčíslitelná v polynomiálním čase, pro kterou platí

$$(\forall w \in \Sigma^*) [w \in B \iff f(w) \in A]$$

- \leq_m^P je reflexivní a tranzitivní (**kvaziuspořádání**)
- Je-li $B \leq_m^P A$ a $A \in P$, pak $B \in P$.
- Je-li $B \leq_m^P A$ a $A \in NP$, pak $B \in NP$.

Neznáme-li žádný polynomiální algoritmus pro B , pak neumíme zkonstruovat ani polynomiální algoritmus pro A .

Polynomiální převoditelnost (reflexivita)

Lemma (Reflexivita polynomiální převoditelnosti)

Pro každý jazyk A platí, že $A \leq_m^P A$.

Důkaz.

- Funkce identity $\text{id}(x) = x$ je vyčíslitelná v polynomiálním čase
- Pro každý řetězec $x \in \Sigma^*$ platí

$$x \in A \Leftrightarrow \text{id}(x) \in A$$



Polynomiální převoditelnost (tranzitivita)

Lemma (Tranzitivita polynomiální převoditelnosti)

Pro každé tři jazyky A , B a C : $A \leq_m^P B \wedge B \leq_m^P C \implies A \leq_m^P C$

- $A \leq_m^P B$ funkcí g
- $B \leq_m^P C$ funkcí h
- Definujme funkci $f(x) = h(g(x))$
 - f je vyčíslitelná v polynomiálním čase
- Pro každý řetězec $x \in \Sigma^*$ platí

$$x \in A \underbrace{\iff}_{A \leq_m^P B} g(x) \in B \underbrace{\iff}_{B \leq_m^P C} h(g(x)) \in C \underbrace{\iff}_{f(x)=h(g(x))} f(x) \in C$$

- $A \leq_m^P C$ funkcí f

Polynomiální převoditelnost srovnává dle obtížnosti

Lemma

Pokud $A \in P$ a $B \leq_m^P A$, pak $B \in P$.

- Uvažme TS M_A , který rozhoduje A v polynomiálním čase
- Popíšeme TS M_B , který rozhoduje B v polynomiálním čase

Výpočet M_B se vstupem x

```
1  $y \leftarrow f(x)$            //  $f$  ukazuje, že  $B \leq_m^P A$ 
2 Pust'  $M_A(y)$ 
3 if  $M_A$  přijal then
4   | přijmi
5 else
6   | odmítni
```

Polynomiální převoditelnost srovnává dle obtížnosti

Lemma

Pokud $A \in \text{NP}$ a $B \leq_m^P A$, pak $B \in \text{NP}$.

- Uvažme polynomiální verifikátor $V_A(x, y)$ pro jazyk A
- Popíšeme polynomiální verifikátor $V_B(x, y)$ pro jazyk B

Výpočet verifikátoru V_B se vstupem x a certifikátem y

```
1  $z \leftarrow f(x)$                                 //  $f$  ukazuje, že  $B \leq_m^P A$ 
2 Pust'  $V_A(z, y)$ 
3 if  $V_A$  přijal then
4   | přijmi
5 else
6   | odmítni
```

3-SAT

3-KNF formule φ je v **3-KNF**, pokud je v KNF a každá klauzule obsahuje právě 3 literály

3-SAT

Instance: Formule φ v 3-KNF.

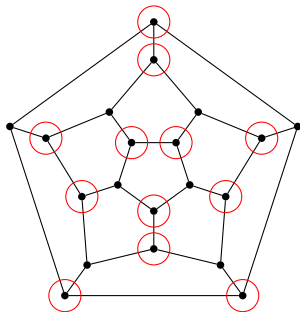
Otázka: Je φ splnitelná?

Vrcholové pokrytí

VRCHOLOVÉ POKRYTÍ

Instance: Neorientovaný graf $G = (V, E)$ a celé číslo $k \geq 0$.

Otázka: Existuje množina vrcholů $S \subseteq V$ velikosti nejvýš k , která obsahuje alespoň jeden vrchol z každé hrany $\{u, v\} \in E$ (tedy $\{u, v\} \cap S \neq \emptyset$)?

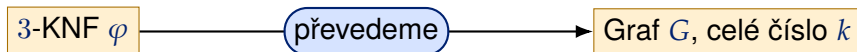


Desetistěn má vrcholové
pokrytí velikosti $k = 12$

3-SAT a VRCHOLOVÉ POKRYTÍ

Věta

3-SAT je polynomiálně převoditelný na VRCHOLOVÉ POKRYTÍ.



φ je splnitelná \longleftrightarrow G má vrcholové pokrytí velikosti k

Převod 3-SAT na VRCHOLOVÉ POKRYTÍ

- Nechť φ je 3-KNF, která má
 - n proměnných x_1, \dots, x_n
 - m klauzulí C_1, \dots, C_m
 - Každá klauzule má právě 3 literály
- Popíšeme, jak sestavit graf $G = (V, E)$ a číslo k

Příklad

Uvažme například 3-KNF

$$\begin{array}{c} \varphi = \overbrace{(x_1 \vee \neg x_2 \vee x_3)}^{C_1} \wedge \overbrace{(x_2 \vee \neg x_3 \vee x_4)}^{C_2} \\ \wedge \underbrace{(\neg x_1 \vee x_2 \vee x_4)}_{C_3} \wedge \underbrace{(\neg x_2 \vee x_3 \vee \neg x_4)}_{C_4} \end{array}$$

Konstrukce grafu (krok 1)

Pro každou proměnnou x_i , $i = 1, \dots, n$

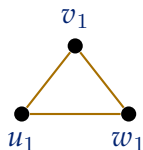
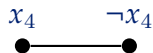
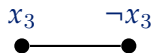
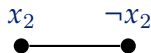
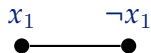
- přidáme dva vrcholy pro literály x_i , $\neg x_i$ a hranu $\{x_i, \neg x_i\}$



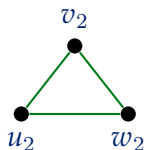
Konstrukce grafu (krok 2)

Pro každou klauzuli C_j , $j = 1, \dots, m$

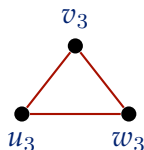
- Přidáme trojúhelník na nově přidané trojici vrcholů u_j, v_j, w_j



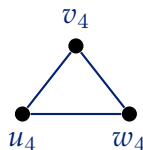
$$x_1 \vee \neg x_2 \vee x_3$$



$$x_2 \vee \neg x_3 \vee x_4$$



$$\neg x_1 \vee x_2 \vee x_4$$

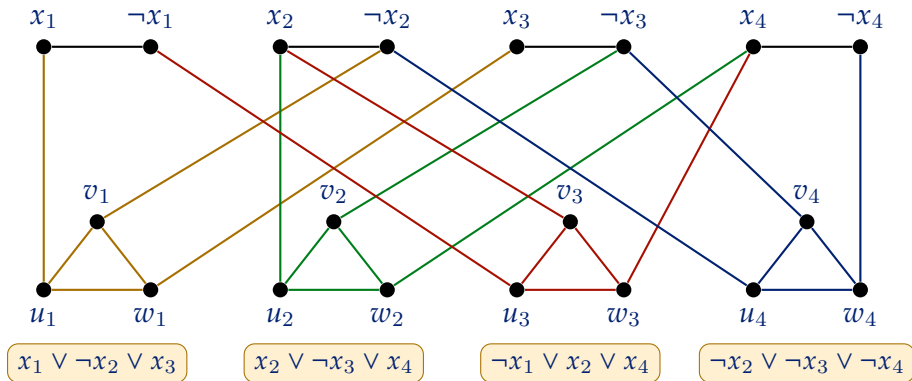


$$\neg x_2 \vee \neg x_3 \vee \neg x_4$$

Konstrukce grafu (krok 3)

Pro každou klauzuli $C_j = (l_1 \vee l_2 \vee l_3)$ s literály l_1, l_2, l_3

- Přidáme hrany $\{u_j, l_1\}, \{v_j, l_2\}, \{w_j, l_3\}$



$$k = 2m + n$$

- 1 vrchol je třeba k pokrytí každé hrany $\{x_i, \neg x_i\}$, $i = 1, \dots, n$
- 2 vrcholy jsou třeba k pokrytí každého trojúhelníku $\{u_j, v_j, w_j\}$, $j = 1, \dots, m$
- Vrcholové pokrytí musí obsahovat alespoň $2m + n$ vrcholů

Graf G má vrcholové pokrytí velikosti $k = 2m + n$ právě když formule φ je splnitelná.

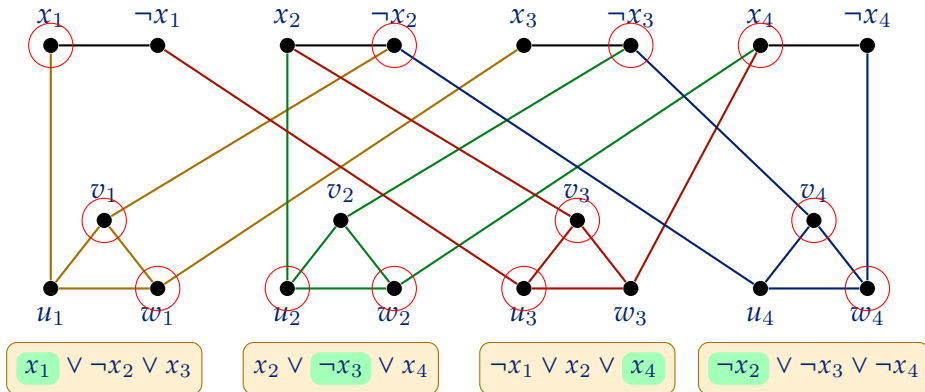
Důkaz „ \Rightarrow “

- Nechť $S \subseteq V$ je vrcholové pokrytí velikosti $k = 2m + n$
- Každá hrana $\{x_i, \neg x_i\}$, $i = 1, \dots, n$ je pokryta právě jedním vrcholem
- Definujeme ohodnocení \mathbf{a} tak, že pro každý index $i = 1, \dots, n$

$$\mathbf{a}(x_i) = \begin{cases} 1 & x_i \in S \\ 0 & \neg x_i \in S \end{cases}$$

Model daný vrcholovým pokrytím

Model daný pokrytím: $\{x_1 \mapsto 1, x_2 \mapsto 0, x_3 \mapsto 0, x_4 \mapsto 1\}$



Každá klauzule obsahuje splněný literál

\mathbf{a} splňuje φ

- Uvažme klauzuli $C_j, j \in \{1, \dots, m\}$
- Předpokládejme, že $C_j = (l_1 \vee l_2 \vee l_3)$
- S obsahuje dva vrcholy trojúhelníku $\{u_j, v_j, w_j\}$
- Jeden vrchol trojúhelníku $\{u_j, v_j, w_j\}$ není v S
- Nechť $u_j \notin S$
 - Případy se zbylými dvěma vrcholy jsou symetrické
- Uvažme hranu $\{u_j, l_1\}$
- Protože $u_j \notin S$, l_1 musí být v S

$\Rightarrow \mathbf{a}(l_1) = 1$

$\Rightarrow C_j$ je splněna ohodnocením \mathbf{a}

Ohodnocení \mathbf{a} splňuje každou klauzuli $C_j \in \varphi$, jde o model.

Důkaz „ \Leftarrow “

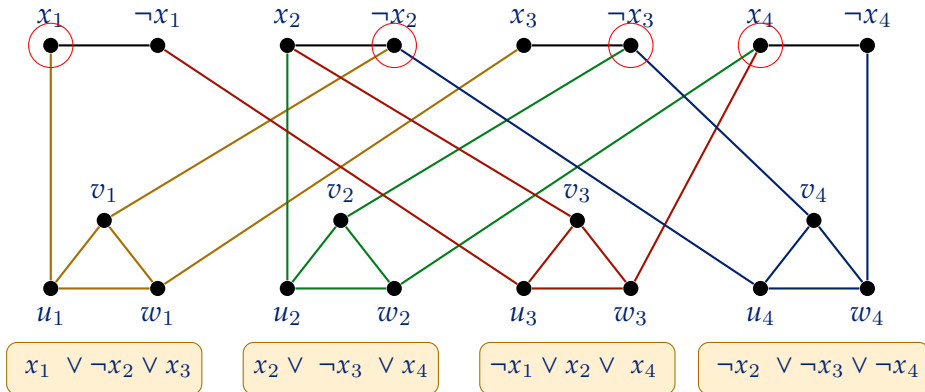
K danému splňující ohodnocení \mathbf{a} definujeme množinu vrcholů S takto

- Pro každé $i = 1, \dots, n$
 - Přidáme do S vrchol x_i pokud $\mathbf{a}(x_i) = 1$
 - Přidáme do S vrchol $\neg x_i$ pokud $\mathbf{a}(x_i) = 0$
- Pro každý trojúhelník $\{u_j, v_j, w_j\}$, $j = 1, \dots, m$
 - Uvažme klauzuli $C_j = (l_1 \vee l_2 \vee l_3)$
 - Jeden z literálů je splněný ohodnocením \mathbf{a}
 - Předpokládejme, že literál l_1 je splněný
 - Případy s ostatními literály jsou symetrické
 - Hrana $\{l_1, u_j\}$ je již pokrytá vrcholem $l_1 \in S$
 - Zbylé vrcholy v_j, w_j přidáme do S

S je vrcholové pokrytí G velikosti $k = 2m + n$.

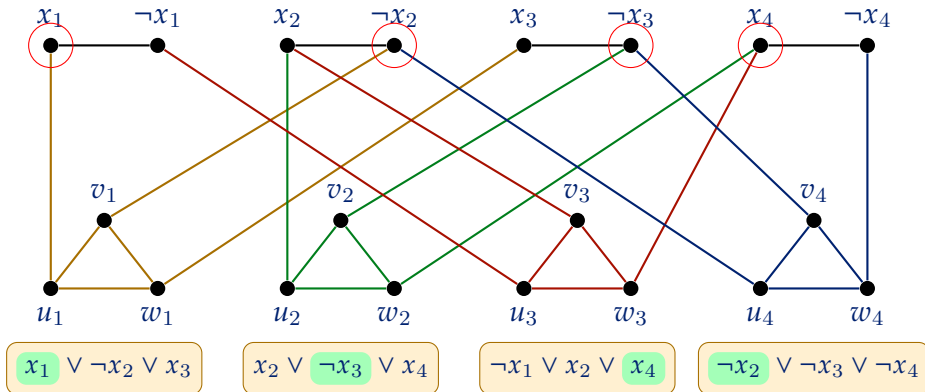
Vrcholové pokrytí dané modelem

Předpokládejme model: $\{x_1 \mapsto 1, x_2 \mapsto 0, x_3 \mapsto 0, x_4 \mapsto 1\}$



Vrcholové pokrytí dané modelem

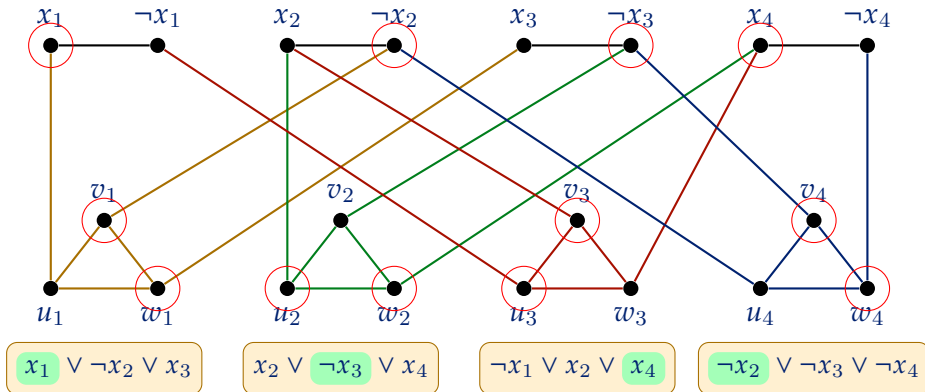
Předpokládejme model: $\{x_1 \mapsto 1, x_2 \mapsto 0, x_3 \mapsto 0, x_4 \mapsto 1\}$



V každé klauzuli vybereme splněný literál

Vrcholové pokrytí dané modelem

Předpokládejme model: $\{x_1 \mapsto 1, x_2 \mapsto 0, x_3 \mapsto 0, x_4 \mapsto 1\}$



Trojúhelníky pokryjeme zbylými vrcholy