Obsah

Poznámky

Tento dokument obsahuje **neoficiální** řešení sbírky z předmětu Teoretická informatika[1]. Najdete zde pouze ty příklady, které nejsou vyřešené v samotné sbírce.

Pokud budete mít pocit, že jste našli nějakou chybu, nebo pokud nebudete některému řešení rozumět, napište mi na discord kocotom#5300. Ozvěte se prosím i tehdy, pokud dojde k aktualizaci sbírky (např. přidání nových příkladů nebo změna zadání stávajících příkladů). Kontaktujte mě prosím i v případě, že se vám podaří vyřešit příklad $21/1d$ (podúkol o **NP**-těžkosti), který se mi dosud vyřešit nepodařilo, a tudíž zde chybí. Rád doplním váš postup.

Chtěl bych poděkovat všem, kteří mi pomohli s kontrolou řešení příkladů. Jmenovitě bych rád vyjádřil vděk následujícím kolegům:

TommY	6.5b
Maestro	2.0b
tomiju	2.0b
MaybeRookie	1.5b
Lakoc	1.0b
Lada	1.0b
Čoveče	1.0b
Pheter	1.0b
dedSnidane	1.0b
mickey	1.0b
stupid boisbb	0.5b
Bc. plov_ec	0.5b
Healtonn	0.5b
jetlebovec	0.5b
Mojo	0.5b

[1] http://www.fit.vutbr.cz/study/courses/TIN/public/Sbirka/all-2021.pdf

Sekce 1: Formální jazyky a základní operace

Úloha 4

Pomocí jazyků $L_0 = \{0\}$ a $L_1 = \{1\}$ nad abecedou $\Sigma = \{0, 1\}$ a základních jazykových operací zkonstruujte následující jazyky:

(b) $L_3 = \{w \in \Sigma^* \mid w$ vyjadřuje liché číslo v binární notaci (bez vedoucích $0)\ \}$
(c) $L_4 = \{w \in \Sigma^* \mid w$ obsahuje buď jeden anebo sudý počet symbolů $1\}$
(d) $L_5 = \{w \in \Sigma^* \mid w$ obsahuje sudý počet podřetězců 01 nebo $10\}$

b)

$$L_3 = L_1(L_0 \cup L_1)^* L_1 \cup L_1$$

c)

$$L_4 = L_0^* \cup L_0^* L_1 L_0^* \cup (L_0^* L_1 L_0^* L_1 L_0^*)^*$$

d)

$$L_5 = L_0^*(L_1^+ L_0^+ L_1^+ L_0^+)^+ L_1^* \cup L_1^*(L_0^+ L_1^+ L_0^+ L_1^+)^+ L_0^* \cup L_1^* \cup L_0^*$$

Úloha 5

Uvažme $\Sigma = \{a, b\}$. Rozhodněte a dokažte, zda platí následující tvrzení:

(c) $\forall L_1, L_2 \subseteq \Sigma^* : L_1$ a L_2 jsou konečné $\iff L_1 \cdot L_2$ je konečný.
(d) $\exists L_1, L_2 \subseteq \Sigma^* : L_1$ a L_2 jsou konečné $\iff L_1 \cdot L_2$ je konečný.
(e) $\forall L_1 \subseteq \Sigma^* : L_1^*$ je nekonečný
(f) $\exists L_1 \subseteq \Sigma^* : L_1^*$ je nekonečný
(g) $\forall L_1, L_2 \subseteq \Sigma^* : L_1^* \cup L_2^* = (L_1 \cup L_2)^*$
(h) $\exists L_1, L_2 \subseteq \Sigma^* : L_1^* \cup L_2^* = (L_1 \cup L_2)^*$
(i) $\forall L_1 \subseteq \Sigma^* : (L_1^*)^* = L_1^*$
(j) $\exists L_1 \subseteq \Sigma^* : (L_1^*)^* = L_1^*$
(k) $\forall L_1 \subseteq \Sigma^* : L_1^* = L_1^+ \iff \epsilon \in L_1$
(l) $\exists L_1 \subseteq \Sigma^* : L_1^* = L_1^+ \iff \epsilon \in L_1$

c)

Jistě neplatí. Jako protipříklad zvolme jazyky $L_1 = \Sigma^*, L_2 = \{\}$. Pak $L_1 \cdot L_2 = \{\}$, tedy konečný jazyk, ale $L_1 = \Sigma^*$ není konečný jazyk.

d)

Jistě platí. Zvolme $L_1 = \{\}, L_2 = \{\}$, oba tyto jazyky jsou konečné a $L_1 \cdot L_2 = \{\}$ je rovněž konečný jazyk. Lze snadno nahlédnout, že platí jak směr $\Leftarrow$, tak směr $\Rightarrow$.

e)

Jistě neplatí. V rámci protipříkladu uvažujme o jazyce $L_1 = \{\}$, pak $L_1^* = \{\epsilon\}$, což je konečný jazyk.

f)

Jistě platí. Uvažujme o jazyce $L_1 = \{a\}$, kde $a \in \Sigma$. Pak $L_1^* = \{a^n \mid n \geq 0\}$, což je nekonečný jazyk.

g)

Jistě neplatí. V rámci protipříkladu uvažujme o jazycích $L_1 = \{a\}, L_2 = \{b\}$, kde $a, b \in \Sigma$. Pak $ab \in (L_1 \cup L_2)^*$, ale $ab \notin L_1^* \cup L_2^*$. Nejde tedy o tytéž jazyky.

h)

Jistě platí. Uvažujme o jazycích $L_1 = \{\}, L_2 = \{\}$, pak $(L_1 \cup L_2)^* = \{\epsilon\}$ a rovněž $L_1^* \cup L_2^* = \{\epsilon\}$.

i)

Jistě platí. K důkazu využijeme definici iterace jazyka, podle které platí:

$$L_1^* = \bigcup_{k=0}^{\infty} L_1^k$$

Můžeme tedy výrok ze zadání upravit do podoby

$$\bigcup_{k=0}^{\infty} L_1^k = \left(\bigcup_{k=0}^{\infty} L_1^k \right)^*$$

neboli

$$\bigcup_{k=0}^{\infty} L_1^k = \bigcup_{l=0}^{\infty} \left(\bigcup_{k=0}^{\infty} L_1^k \right)^l$$

Tuto rovnost dokážeme postupně z obou stran.
Směr $\subseteq$:

$$\bigcup_{k=0}^{\infty} L_1^k \subseteq \bigcup_{l=0}^{\infty} \left(\bigcup_{k=0}^{\infty} L_1^k \right)^l$$

Rozepíšeme pravou stranu

$$\underline{\bigcup_{k=0}^{\infty} L_1^k} \subseteq \{\epsilon\} \cup \underline{\left(\bigcup_{k=0}^{\infty} L_1^k \right)} \cup \left(\bigcup_{k=0}^{\infty} L_1^k \right)^2 \cup \left(\bigcup_{k=0}^{\infty} L_1^k \right)^3 \cup \dots$$

Podtržené části jsou zjevně totožné, pročež inkluze platí.
Směr $\supseteq$:

$$\bigcup_{k=0}^{\infty} L_1^k \supseteq \bigcup_{l=0}^{\infty} \left(\bigcup_{k=0}^{\infty} L_1^k \right)^l$$

Ukážeme, že platí následující výrok:

$$w \in \bigcup_{l=0}^{\infty} \left(\bigcup_{k=0}^{\infty} L_1^k \right)^l \implies w \in \bigcup_{k=0}^{\infty} L_1^k$$

Jelikož množina uvedená v levé části implikace vznikla pomocí nekonečného sjednocení množin, je jasné, že w leží v některé z nich:

$$w \in \bigcup_{l=0}^{\infty} \left(\bigcup_{k=0}^{\infty} L_1^k \right)^l \implies \exists n \in \mathbb{N} : w \in \left(\bigcup_{k=0}^{\infty} L_1^k \right)^n$$

Tento řetězec w tedy musí být prvkem konkatenace n jazyků. Je tomu tak proto, že w lze zapsat jako $w_1 \cdot w_2 \cdot \cdots \cdot w_n$, kde $\forall i \in \mathbb{N} : 1 \leq i \leq n \Rightarrow w_i \in \bigcup_{k=0}^{\infty} L_1^k$. Jelikož $\bigcup_{k=0}^{\infty} L_1^k$ je opět sjednocením nekonečného množství jazyků, je zřejmé, že každé w_i musí být prvkem některého z nich, jak shrnuje následující výrok:

$$\exists n \in \mathbb{N} : w \in \left(\bigcup_{k=0}^{\infty} L_1^k \right)^n \implies \exists n \in \mathbb{N}\, \exists m_1, m_2, \ldots m_n \in \mathbb{N} : w \in L_1^{m_1} \cdot L_1^{m_2} \cdot L_1^{m_3} \ldots L_1^{m_n}$$

Konkatenaci mocnin téhož jazyka lze upravit:

$$\begin{aligned} &\exists n \in \mathbb{N}\, \exists m_1, m_2, \ldots m_n \in \mathbb{N} : w \in L_1^{m_1} \cdot L_1^{m_2} \cdot L_1^{m_1} \ldots L_1^{m_n} \implies \\ &\exists n \in \mathbb{N}\, \exists m_1, m_2, \ldots m_n \in \mathbb{N} : w \in L_1^{m_1 \cdot m_2 \cdot \cdots \cdot m_n} \end{aligned}$$

Pokud $w \in L_1^{m_1 \cdot m_2 \cdot \cdots \cdot m_n}$, pak jistě patří do libovolné nadmnožiny $L_1^{m_1 \cdot m_2 \cdot \cdots \cdot m_n}$.

$$\begin{aligned} &\exists n \in \mathbb{N}\, \exists m_1, m_2, \ldots m_n \in \mathbb{N} : w \in L_1^{m_1 \cdot m_2 \cdot \cdots \cdot m_n} \implies \\ &\exists n \in \mathbb{N}\, \exists m_1, m_2, \ldots m_n \in \mathbb{N} : w \in L_1^0 \cup L_1^1 \cup L_1^2 \cup \cdots \cup L_1^{m_1 \cdot m_2 \cdot \cdots \cdot m_n} \cup \ldots \end{aligned}$$

Což lze upravit:

$$\exists n \in \mathbb{N}\, \exists m_1, m_2, \ldots m_n \in \mathbb{N} : w \in L_1^0 \cup L_1^1 \cup L_1^2 \cup \cdots \cup L_1^{m_1 \cdot m_2 \cdot \cdots \cdot m_n} \cup \cdots \implies w \in \bigcup_{k=0}^{\infty} L_1^k$$

Dostáváme tedy

$$\bigcup_{k=0}^{\infty} L_1^k \supseteq \bigcup_{l=0}^{\infty} \left(\bigcup_{k=0}^{\infty} L_1^k \right)^l$$

Zadaný výrok $L_1^* = (L_1^*)^*$ tedy platí.

j)

Jistě platí. Toto triviálně vyplývá z **i)**, neboť pokud pro každý jazyk $L_1 \subseteq \Sigma^*$ platí nějaká vlastnost, pak určitě existuje jazyk $L_1 \subseteq \Sigma^*$, pro který daná vlastnost platí.

k)

Jistě platí. Dokážeme postupně jednotlivé směry implikace.
Směr $\Rightarrow$:

$$L_1^* = L_1^+ \Rightarrow \epsilon \in L_1$$

Jelikož platí $L_1^* = L_1^+ \cup \{\epsilon\}$, píšeme

$$L_1^+ \cup \{\epsilon\} = L_1^+ \Rightarrow \epsilon \in L_1$$

Implikace platí, neboť rovnost na levé straně implikace je splněna jen v případě, že $\epsilon \in L_1^+$
Směr $\Leftarrow$:

$$L_1^* = L_1^+ \Leftarrow \epsilon \in L_1$$

Obecně pokud $a \in A$, pak $A \cup \{a\} = A$, proto můžeme psát:

$$\epsilon \in L_1 \Rightarrow L_1^+ = L_1^+ \cup \{\epsilon\} \Rightarrow L_1^+ = L_1^*$$

l)

Jistě platí. Toto triviálně vyplývá z **k)**, neboť pokud pro každý jazyk $L_1 \subseteq \Sigma^*$ platí nějaká vlastnost, pak určitě existuje jazyk $L_1 \subseteq \Sigma^*$, pro který daná vlastnost platí.

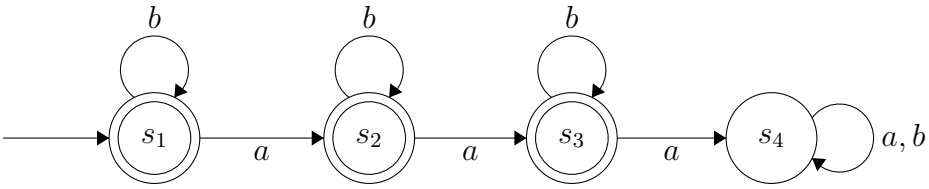
Sekce 2: Základní konstrukce pro regulární jazyky

Úloha 1

Zkonstruujte konečné automaty a regulární gramatiky pro tyto jazyky:

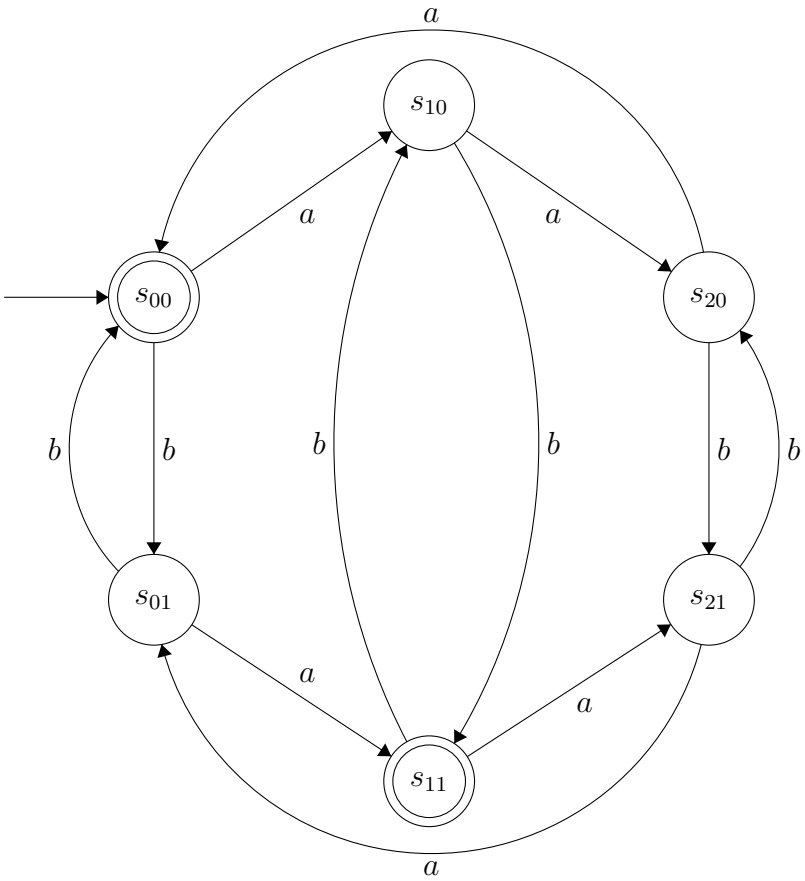
(b) $L = \{w \in \{a,b\}^* \mid \#_a(w) < 3\}$
(c) $L = \{w \in \{a,b\}^* \mid \#_a(w) \bmod 3 = \#_b(w) \bmod 2\}$
(d) $L = a^* \cdot (bb + cc)^*$

b)



$G = (\{S, S_1, S_2\}, \{a,b\}, P, S)$, kde
$P : S \to aS_1 \mid bS \mid \epsilon$
$S_1 \to aS_2 \mid bS_1 \mid \epsilon$
$S_2 \to bS_2 \mid \epsilon$

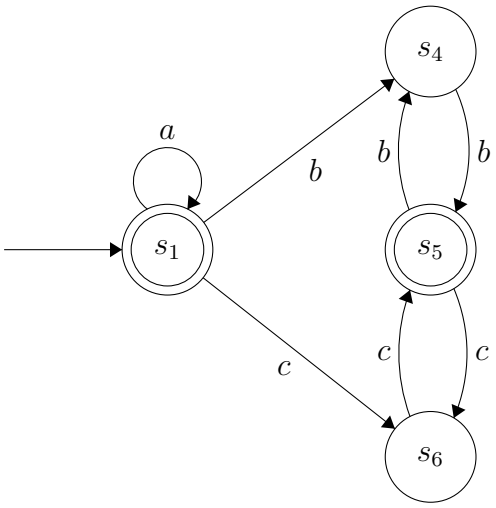
c)



$G = (\{S_{00}, S_{01}, S_{10}, S_{11}, S_{20}, S_{21}\}, \{a,b\}, P, S_{00})$, kde

$P : S_{00} \to aS_{10} \mid bS_{01} \mid \epsilon$
$S_{01} \to aS_{11} \mid bS_{00}$
$S_{10} \to aS_{20} \mid bS_{11}$
$S_{11} \to aS_{21} \mid bS_{10} \mid \epsilon$
$S_{20} \to aS_{00} \mid bS_{21}$
$S_{21} \to aS_{01} \mid bS_{20}$

d)



$G = (\{S, B, C, X\}, \{a, b\}, P, S)$, kde
$P : S \to aS \mid bB \mid cC \mid \epsilon$
$B \to bX$
$C \to cX$
$X \to bB \mid cC \mid \epsilon$

Úloha 2

Pomocí regulárních výrazů popište tyto jazyky:

(b) $L = \{w \in \{a, b, c\}^* \mid \#_a(w) < 2\}$
(c) $L = \{w \in \{a, b, c\}^* \mid \#_a(w) \bmod 2 = 0\}$

b)

$$L = (b + c)^* + (b + c)^* a (b + c)^*$$

c)

$$L = (b + c)^* + ((b + c)^* a (b + c)^* a (b + c)^*)^*$$

Sekce 3: Transformace konečných automatů a regulárních gramatik

Úloha 1

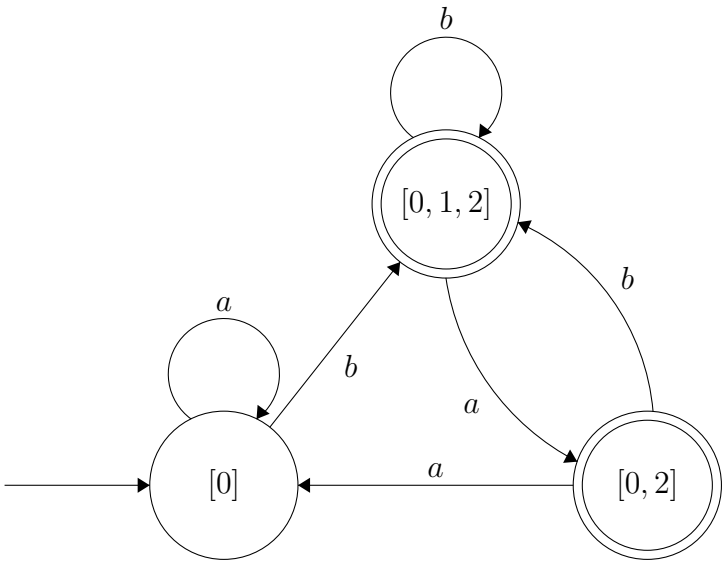
Převeďte následující nedeterministické automaty na ekvivalentní deterministické automaty. Postupujte dle algoritmu z přednášky. Není třeba uvádět nedosažitelné stavy.

(b) $A = (\{q_0, q_1, q_2\}, \{a, b\}, \delta, q_0, \{q_2\})$, kde δ je definována jako

$\delta(q_0, a) = \{q_0\},\ \delta(q_0, b) = \{q_0, q_1, q_2\}$
$\delta(q_1, a) = \{q_2\},\ \delta(q_1, b) = \{q_0\}$
$\delta(q_2, a) = \{\},\ \delta(q_2, b) = \{q_2\}$

b)

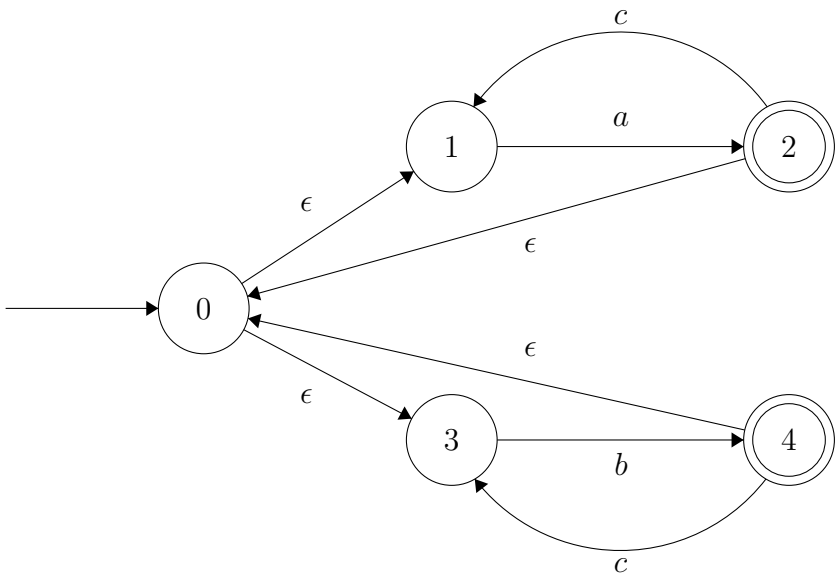
	a	b
$\rightarrow \{q_0\}$	$\{q_0\}$	$\{q_0, q_1, q_2\}$
$\leftarrow \{q_0, q_1, q_2\}$	$\{q_0, q_2\}$	$\{q_0, q_1, q_2\}$
$\leftarrow \{q_0, q_2\}$	$\{q_0\}$	$\{q_0, q_1, q_2\}$



Úloha 2

Převeďte následující rozšířené automaty (tj. automaty s $\epsilon-$kroky) na ekvivalentní deterministické automaty. Postupujte dle algoritmu z přednášky. Není třeba uvádět nedosažitelné stavy.

(b) Automat na obrázku níže.



b)

Počáteční stav nového automatu nazvěme A.

ϵ-uzávěr$(\{0\}) = \{0, 1, 3\} = A$

$\delta(A, a) = \epsilon$-uzávěr$(\{2\}) = \{2, 0, 1, 3\} = B$
$\delta(A, b) = \epsilon$-uzávěr$(\{4\}) = \{4, 0, 1, 3\} = C$
$\delta(A, c) = \epsilon$-uzávěr$(\{\}) = \{\} = D$

$\delta(B, a) = \epsilon$-uzávěr$(\{2\}) = \{2, 0, 1, 3\} = B$
$\delta(B, b) = \epsilon$-uzávěr$(\{4\}) = \{4, 0, 1, 3\} = C$
$\delta(B, c) = \epsilon$-uzávěr$(\{1\}) = \{1\} = E$

$\delta(C, a) = \epsilon$-uzávěr$(\{2\}) = \{2, 0, 1, 3\} = B$
$\delta(C, b) = \epsilon$-uzávěr$(\{4\}) = \{4, 0, 1, 3\} = C$
$\delta(C, c) = \epsilon$-uzávěr$(\{3\}) = \{3\} = F$

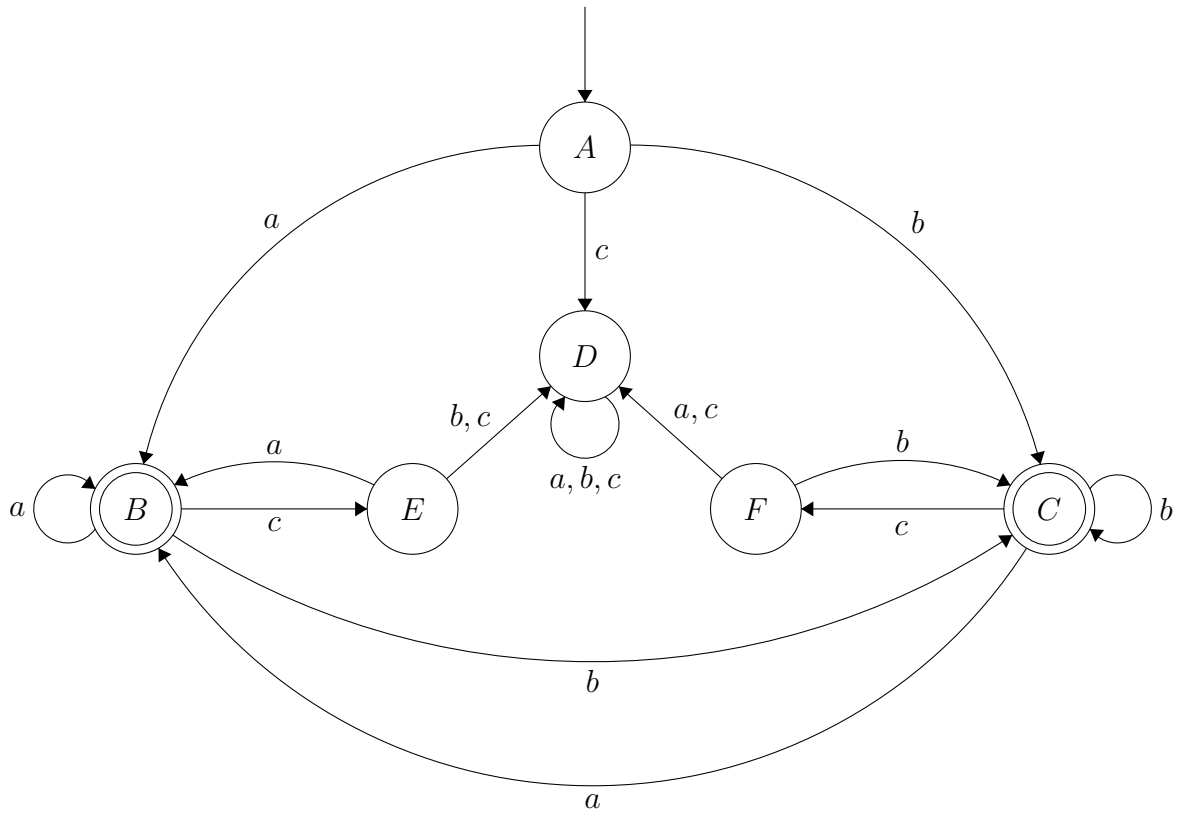
$\delta(D, a) = \delta(D, b) = \delta(D, c) = D$

$\delta(E, a) = \epsilon$-uzávěr$(\{2\}) = \{2, 0, 1, 3\} = B$
$\delta(E, b) = \epsilon$-uzávěr$(\{\}) = \{\} = D$
$\delta(E, c) = \epsilon$-uzávěr$(\{\}) = \{\} = D$

$\delta(F, a) = \epsilon$-uzávěr$(\{\}) = \{\} = D$
$\delta(F, b) = \epsilon$-uzávěr$(\{4\}) = \{4, 0, 1, 3\} = C$
$\delta(F, c) = \epsilon$-uzávěr$(\{\}) = \{\} = D$

$F_A = \{B, C\}$ (množina koncových stavů)

Následující obrázek zobrazuje deterministický konečný automat, který vznikl převodem z rozšířeného konečného automatu.



Lze snadno nahlédnout, že stav D je neukončující.

Úloha 3

Následující automaty zadané tabulkou převeďte do redukované formy (tj. zkonstruujte ekvivalentní minimální automat)

(b)

stav	a	b
$\leftrightarrow 1$	3	2
2	6	4
3	3	5
$\leftarrow 4$	4	2
5	10	8
6	6	7
$\leftarrow 7$	7	5
$\leftarrow 8$	8	2
$\leftarrow 9$	11	2
10	10	9
11	11	5

b)

	stav	a	b
I	2	I	II
	3	I	I
	5	I	II
	6	I	II
	10	I	II
	11	I	I
$\longleftrightarrow$ II	1	I	I
	4	II	I
	7	II	I
	8	II	I
	9	I	I

$\overset{0}{\equiv}$

	stav	a	b
I	2	I	IV
	5	I	IV
	6	I	IV
	10	I	III
II	3	II	I
	11	II	I
$\longleftrightarrow$ III	1	II	I
	9	II	I
$\longleftarrow$ IV	4	IV	I
	7	IV	I
	8	IV	I

$\overset{1}{\equiv}$

	stav	a	b
I	2	I	V
	5	II	IV
	6	I	V
II	10	II	IV
III	3	III	I
	11	III	I
$\longleftrightarrow$ IV	1	III	I
	9	III	I
$\longleftarrow$ V	4	V	I
	7	V	I
	8	V	I

$\overset{2}{\equiv}$

	stav	a	b
I	2	I	VI
	6	I	VI
II	5	III	VI
III	10	III	V
IV	3	IV	II
	11	IV	II
$\longleftrightarrow$ V	1	IV	I
	9	IV	I
$\longleftarrow$ VI	4	VI	I
	7	VI	II
	8	VI	I

$\overset{3}{\equiv}$

	stav	a	b
I	2	I	VI
	6	I	VII
II	5	III	VI
III	10	III	V
IV	3	IV	II
	11	IV	II
$\longleftrightarrow$ V	1	IV	I
	9	IV	I
$\longleftarrow$ VI	4	VI	I
	8	VI	I
$\longleftarrow$ VII	7	VII	II

$\overset{4}{\equiv}$

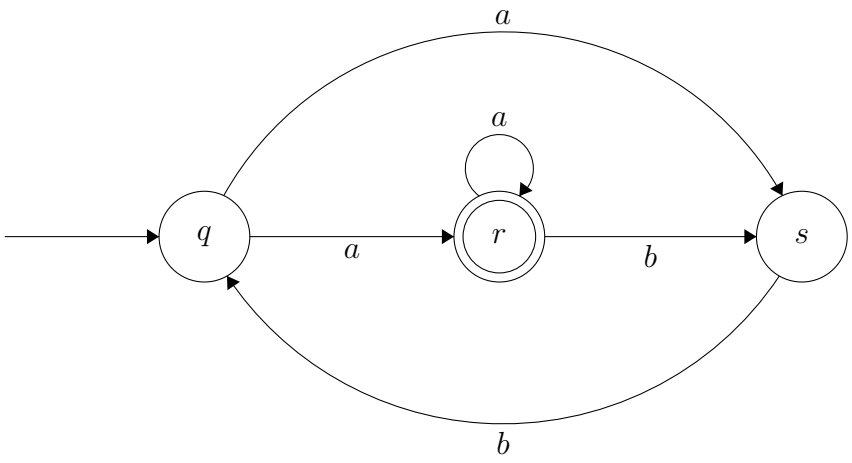
	stav	a	b
I	2	II	VII
II	6	II	VIII
III	5	IV	VII
IV	10	IV	VI
V	3	V	III
	11	V	III
$\longleftrightarrow$ VI	1	V	I
	9	V	I
$\longleftarrow$ VII	4	VII	I
	8	VII	I
$\longleftarrow$ VIII	7	VIII	III

$\overset{5}{\equiv}=\overset{6}{\equiv}$

Úloha 4

Následující konečné automaty převeďte na ekvivalentní regulární gramatiky.

(b) Automat uvedený níže:



b)

$G = (\{S_1, S_2, S_3\}, \{a, b\}, P, S_1)$, kde:

$P : S_1 \to aS_2 \mid aS_3$
$S_2 \to aS_2 \mid bS_3 \mid \epsilon$
$S_3 \to bS_1$

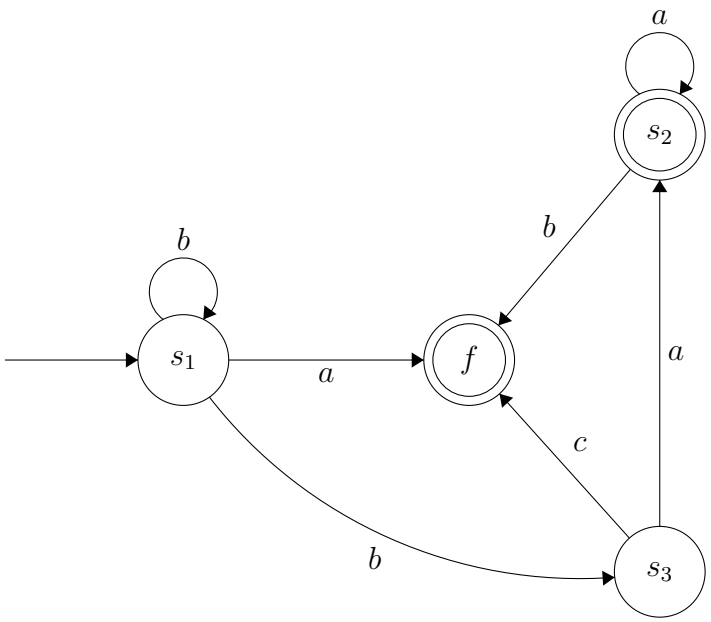
Úloha 5

Následující regulární gramatiky převeďte na ekvivalentní konečné automaty.

(b) $G = (\{S_1, S_2, S_3\}, \{a, b, c\}, P, S_1)$, kde

$P : S_1 \to a \mid bS_1 \mid bS_3$
$S_2 \to \epsilon \mid aS_2 \mid b$
$S_3 \to aS_2 \mid c$

b)



Úloha 6

Následující regulární výrazy převeďte na ekvivalentní regulární gramatiky.

(b) $((a+b)^+c)^*$

b)

$G = (\{S, A\}, \{a, b, c\}, P, S)$, kde

$P : S \to \epsilon \mid aA \mid bA$
$A \to aA \mid bA \mid cS$

Správnost gramatiky lze snadno ověřit řešením soustavy rovnic nad regulárními výrazy

$$S = \epsilon + aA + bA = \epsilon + (a+b)A$$

$$A = aA + bA + cS = (a+b)A + cS = (a+b)^*cS$$

Po dosazení A do první rovnice dostáváme:

$$S = \epsilon + (a+b)(a+b)^*cS = \epsilon + (a+b)^+cS = ((a+b)^+c)^*,$$

což je původní regulární výraz.

Úloha 7

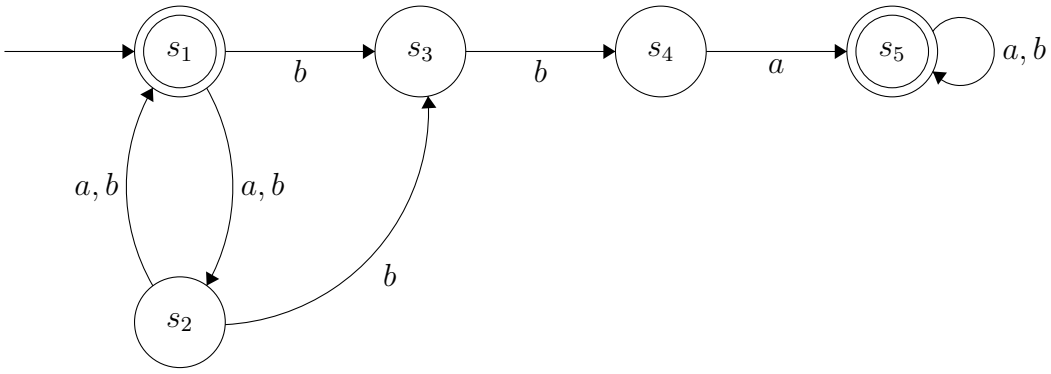
Pro následující jazyky

- Sestrojte NKA pro jazyk (pokuste se efektivně využít nedeterminismus).
- Převeďte algoritmicky tento NKA na ekvivalentní DKA.
- Získaný DKA převeďte algoritmicky do redukované podoby, nebo ukažte, že již v redukované podobě je.

(b) $L = \{w \in \{a,b\}^* \mid w \text{ obsahuje podslovo } bba\} \cup \{w \in \{a,b\}^* \mid |w| \bmod 2 = 0\}$

b)

Nedeterministický automat:



Determinizace (poslední sloupec uvádí nové jméno pro stav v prvním sloupci kvůli snažšímu zápisu při následné minimalizaci):

	a	b	nové jméno pro stav
$\leftrightarrow \{s_1\}$	$\{s_2\}$	$\{s_2, s_3\}$	1
$\{s_2\}$	$\{s_1\}$	$\{s_1, s_3\}$	2
$\{s_2, s_3\}$	$\{s_1\}$	$\{s_1, s_3, s_4\}$	3
$\leftarrow \{s_1, s_3\}$	$\{s_2\}$	$\{s_2, s_3, s_4\}$	4
$\leftarrow \{s_1, s_3, s_4\}$	$\{s_2, s_5\}$	$\{s_2, s_3, s_4\}$	5
$\{s_2, s_3, s_4\}$	$\{s_1, s_5\}$	$\{s_1, s_3, s_4\}$	6
$\leftarrow \{s_2, s_5\}$	$\{s_1, s_5\}$	$\{s_1, s_3, s_5\}$	7
$\leftarrow \{s_1, s_5\}$	$\{s_2, s_5\}$	$\{s_2, s_3, s_5\}$	8
$\leftarrow \{s_1, s_3, s_5\}$	$\{s_2, s_5\}$	$\{s_2, s_3, s_4, s_5\}$	9
$\leftarrow \{s_2, s_3, s_5\}$	$\{s_1, s_5\}$	$\{s_1, s_3, s_4, s_5\}$	10
$\leftarrow \{s_2, s_3, s_4, s_5\}$	$\{s_1, s_5\}$	$\{s_1, s_3, s_4, s_5\}$	11
$\leftarrow \{s_1, s_3, s_4, s_5\}$	$\{s_2, s_5\}$	$\{s_2, s_3, s_4, s_5\}$	12

Minimalizace:

stav	a	b
$\leftrightarrow$ 1	2	3
2	1	4
3	1	5
$\leftarrow$ 4	2	6
$\leftarrow$ 5	7	6
6	8	5
$\leftarrow$ 7	8	9
$\leftarrow$ 8	7	10
$\leftarrow$ 9	7	11
$\leftarrow$ 10	8	12
$\leftarrow$ 11	8	12
$\leftarrow$ 12	7	11

Přechodová tabulka

	stav	a	b
$\leftrightarrow$ I	1	II	II
	4	II	II
	5	I	II
	7	I	I
	8	I	I
	9	I	I
	10	I	I
	11	I	I
	12	I	I
II	2	I	I
	3	I	I
	6	I	I

$\overset{0}{\equiv}$

	stav	a	b
$\leftrightarrow$ I	1	IV	IV
	4	IV	IV
$\leftarrow$ II	5	III	IV
$\leftarrow$ III	7	III	III
	8	III	III
	9	III	III
	10	III	III
	11	III	III
	12	III	III
IV	2	I	I
	3	I	II
	6	III	II

$\overset{1}{\equiv}$

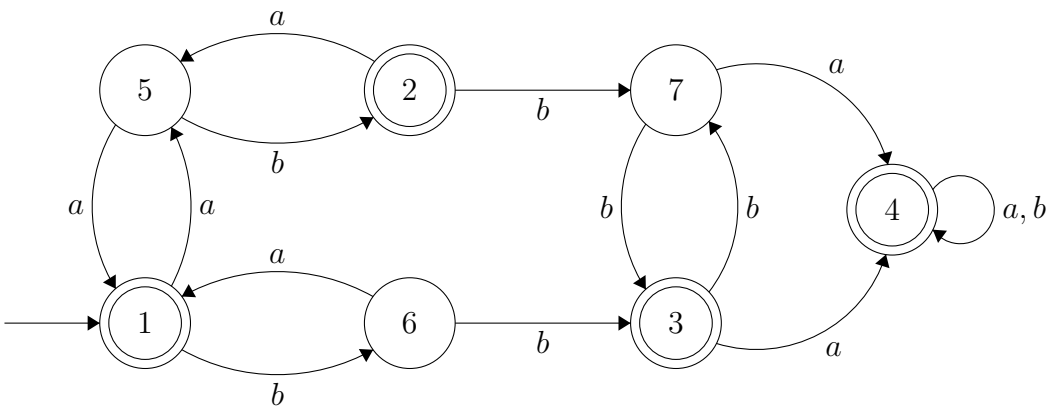
	stav	a	b
$\leftrightarrow$ I	1	IV	V
	4	IV	VI
$\leftarrow$ II	5	III	VI
$\leftarrow$ III	7	III	III
	8	III	III
	9	III	III
	10	III	III
	11	III	III
	12	III	III
IV	2	I	I
V	3	I	II
VI	6	III	II

$\overset{2}{\equiv}$

	stav	a	b
$\leftrightarrow$ I	1	V	VI
$\leftarrow$ II	4	V	VII
$\leftarrow$ III	5	IV	VII
$\leftarrow$ IV	7	IV	IV
	8	IV	IV
	9	IV	IV
	10	IV	IV
	11	IV	IV
	12	IV	IV
V	2	I	II
VI	3	I	III
VII	6	IV	III

$\overset{3}{\equiv}$

Ukázka minimálního automatu:



Úloha 8

Navrhněte a formálně popište algoritmus pro operaci průnik nad dvěma

(b) nedeterministickými KA (nepoužívejte determinizaci).

b)

Vstup: Nedeterministické konečný automaty $A_1 = (Q_1, \Sigma_1, \delta_1, s_1, F_1), A_2 = (Q_2, \Sigma_2, \delta_2, s_2, F_2)$
Výstup: Nedeterministický konečný automat $A_\cap = (Q_\cap, \Sigma_\cap, \delta_\cap, s_\cap, F_\cap)$
Postup:

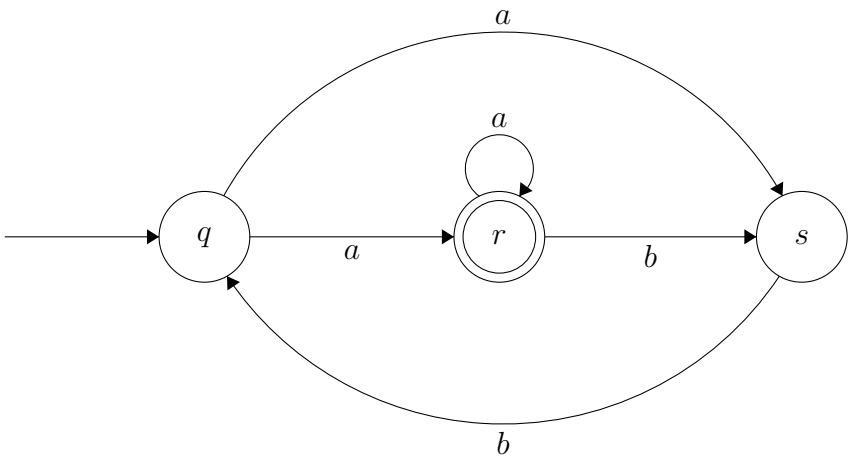
1. $Q_\cap = Q_1 \times Q_2$ (předpokládáme, že $Q_1 \cap Q_2 = \{\}$, jinak přejmenujeme stavy)
2. $\Sigma_\cap = \Sigma_1 \cap \Sigma_2$ (pokud $\Sigma_1 \cap \Sigma_2 = \{\}$, pak algoritmus vrátí automat $(\{s\}, \{a\}, s, \{\}, \{\})$)
3. $\delta_\cap : \forall (q_1, q_2) \in Q_\cap \, \forall \sigma \in \Sigma_\cap : \delta_\cap((q_1, q_2), \sigma) = \delta_1(q_1, \sigma) \times \delta_2(q_2, \sigma)$
4. $s_\cap = (s_1, s_2)$
5. $F_\cap = F_1 \times F_2$

Sekce 4: Rovnice nad regulárními výrazy

Úloha 1

Řešením rovnic nad regulárními výrazy sestavte ekvivalentní regulární výraz k těmto automatům.

(b) Automat zakreslený níže



b)

Jednotlivé stavy si nazvěme Q, R, S (zleva doprava dle obrázku). Sestavme jednotlivé rovnice nad regulárními výrazy následovně:

$$Q = aR + aS$$

$$R = aR + bS + \epsilon$$

$$S = bQ$$

Třetí rovnici dosaďme do dvou předchozích:

$$Q = aR + abQ$$

$$R = aR + bbQ + \epsilon$$

Nyní pro úpravu druhé rovnice využijeme té vlastnosti, že nejmenší pevný bod rovnice nad regulárními výrazy $X = pX + q$ je $X = p^*q$.

$$Q = aR + abQ$$

$$R = a^*(bbQ + \epsilon)$$

Nyní dosadíme druhou rovnici do první.

$$Q = aa^*(bbQ + \epsilon) + abQ$$

Pravou stranu rovnice roznásobíme.

$$Q = aa^*bbQ + aa^* + abQ$$

Na pravé straně rovnice zprava vytkneme Q z těch členů, v nichž se Q vyskytuje.

$$Q = (aa^*bb + ab)Q + aa^*$$

Pro další úpravu opět využijeme pevný bod rovnice nad regulárními výrazy $X = pX + q$.

$$Q = (aa^*bb + ab)^*aa^*$$

Jelikož Q byl náš startující stav, odpovídá regulární výraz $(aa^*bb+ab)^*aa^*$ zadanému automatu. Tento výraz můžeme dále upravit například do podoby $(a(a^*b + \epsilon)b)^*a^+$.

Sekce 5: Pumping lemma

Úloha 1

Rozhodněte a dokažte, zda jsou následující jazyky regulární. Při dokazování, že je jazyk regulární, stačí uvést odpovídající gramatiku či automat. Při dokazování, že jazyk není regulární, použijte Pumping lemma.

(c) $L_3 = \{w \mid w \in \{a,b,c\}^* \wedge \#_a(w) = \#_b(w) \wedge \#_c(w) = 1\}$
(d) $L_4 = \{a^i b^{2i} c^j \mid i > 0 \wedge i < j < 2 \cdot i\}$

c)

Jazyk L_3 není regulární. Dokážeme to pomocí Pumping lemmatu v podobě obměněné věty.

$$\forall k > 0\ \exists w \in L : |w| \geq k\ \wedge (\forall x,y,z \in \Sigma^* : w = xyz \wedge |xy| \leq k \wedge |y| \neq 0 \Rightarrow \exists i \geq 0 : xy^i z \notin L) \Rightarrow L \notin \mathcal{L}_3$$

Předpokládejme libovolné $k > 0$. Zvolme slovo $w = a^k c b^k$. Jistě $w \in L_3$. Uvažujme o všech možných rozděleních $w = xyz$, kde $y \neq \epsilon \wedge |xy| \leq k$.

$$\begin{aligned} x &= a^{\varphi_1} \\ y &= a^{\varphi_2} \\ z &= a^{k-\varphi_1-\varphi_2} c b^k \end{aligned}$$

Zavedeme podmínky $\varphi_1 + \varphi_2 \leq k \wedge \varphi_2 \neq 0 \wedge \varphi_1, \varphi_2 \in \mathbb{N}$. Zvolme nyní $i = 2$. Pak $w' = xy^2z = a^{\varphi_1} a^{2\cdot\varphi_2} a^{k-\varphi_1-\varphi_2} c b^k = a^{k+\varphi_2} c b^k$. Platí tedy $\#_a(w') = k + \varphi_2$ a $\#_b(w') = k$. My jsme však zavedli podmínku $\varphi_2 \neq 0$, pročež $\#_a(w') > \#_b(w')$, což je v rozporu s podmínkami jazyka, tedy $w' \notin L$, z čehož plyne, že $L_3 \notin \mathcal{L}_3$.

d)

Jazyk L_4 není regulární. Dokážeme to pomocí Pumping lemmatu v podobě obměněné věty.
Předpokládejme libovolné $k > 0$. Zvolme slovo $w = a^{2k} b^{4k} c^{3k}$. Jistě $w \in L_4$. Uvažujme nyní o všech možných rozděleních $w = xyz$, kde $y \neq \epsilon \wedge |xy| \leq k$.

$$\begin{aligned} x &= a^{\varphi_1} \\ y &= a^{\varphi_2} \\ z &= a^{2k-\varphi_1-\varphi_2} b^{4k} c^{3k} \end{aligned}$$

Zavedeme podmínky $\varphi_1 + \varphi_2 \leq k \wedge \varphi_2 \neq 0 \wedge \varphi_1, \varphi_2 \in \mathbb{N}$. Zvolme nyní $i = 2$. Pak $w' = xy^2z = a^{\varphi_1} a^{2\cdot\varphi_2} a^{2\cdot k-\varphi_1-\varphi_2} b^{4k} c^{3k} = a^{2\cdot k+\varphi_2} b^{4k} c^{3k}$. Platí tedy $\#_a(w') = 2 \cdot k + \varphi_2$ a $\#_b(w') = 4 \cdot k$. Jelikož jsme zavedli podmínku $\varphi_2 \neq 0$, určitě $2 \cdot \#_a(w') \neq \#_b(w')$, což je v rozporu s podmínkami jazyka, pročež $w' \notin L$, z čehož plyne, že $L_4 \notin \mathcal{L}_3$.

Úloha 2

Dokažte, že následující jazyky splňují pravou stranu Pumping lemmatu.

(b) $L = \{w \in \{a,b\}^* \mid |w| > 3\}$
(c) $L = \{w \in \{a,b\}^* \mid |w| < 3\}$

b)

Zvolme $k = 5$. Každé slovo $w \in L$, kde $|w| \geq k$, pak lze zapsat jako $w = \sigma w'$, kde $\sigma \in \{a, b\} \wedge |w'| \geq 4$. Zvolme rozdělení $x = \epsilon, y = \sigma, z = w'$. Pak pro $i = 0$ platí $xy^0z = xz = w' \in L$, neboť $|w'| > 3$. Pro $i > 0$ pak nemůže být počet znaků nižší. Ukázali jsme, že jazyk L splňuje pravou stranu Pumping lemmatu.

c)

Zvolme $k = 3$ (nebo libovolné vyšší přirozené číslo). Jelikož všechna slova $w \in L$ mají délku kratší než 3, pak neexistují žádná slova, pro něž by byla levá strana implikace splněna. Levá strana implikace má tedy vždy pravdivostní hodnotu 0, pročež L splňuje pravou stranu Pumping lemmatu.

Sekce 6: Myhill-Nerodova věta

Úloha 1

S využitím Myhill-Nerodovy věty dokažte, že následující jazyky nejsou regulární.

(b) $L_2 = \{a^{2^n} \mid n > 0\}$
(c) $L_3 = \{w \in \{a, b\}^* \mid w = w^R\}$

b)

Jazyk $L_2 = \{a^{2^n} \mid n > 0\}$ není regulární. Dokážeme to pomocí Myhill-Nerodovy věty. Pro spor předpokládejme, že regulární je. Pak $\exists n : |\Sigma^*/_{\sim_L}| = n$, tedy index relace prefixové ekvivalence je konečný. Uvažujme nyní o $n + 1$ slovech:

$$a^{2^0}, a^{2^1}, \ldots, a^{2^n}$$

Uvažujme dále o dvou různých přirozených číslech i, j, kde $0 \leq i \leq n \wedge 0 \leq j \leq n$. Pokud $a^{2^i} \sim_L a^{2^j}$, pak $\forall w \in \Sigma^* : a^{2^i} w \in L \iff a^{2^j} w \in L$. Zvolme $w = a^{2^i}$. Pak $a^{2^i} a^{2^i} = a^{2^i + 2^i} = a^{2 \cdot 2^i} = a^{2^{i+1}} \in L$, nicméně $a^{2^j} a^{2^i} = a^{2^j + 2^i} \notin L$. Ukažme nyní, proč platí $i \neq j \Rightarrow a^{2^i + 2^j} \notin L$. Předpokládejme, že $j > i$. Můžeme jistě psát $2^i + 2^j = 2^i + 2^{i+j-i} = 2^i + 2^i \cdot 2^{j-i} = 2^i \cdot (1 + 2^{j-i})$. Protože $i \neq j$, pak $2^{j-i} \neq 1$, a tudíž $2^i + 2^j = 2^i \cdot (1 + 2^{j-i})$ ve svém prvočíselném rozkladu obsahuje nějaké liché číslo. Nemůže tedy být kladnou mocninou čísla 2. Totéž platí v případě, že $j < i$. Jistě tedy $a^{2^i} \not\sim_L a^{2^j}$.
Ukázali jsme, že žádné z těchto $n + 1$ slov nemůže být ve stejné třídě rozkladu. To je ale spor s naším předpokladem, že $|\Sigma^*/_{\sim_L}| = n$. Z toho plyne, že jazyk L není regulární.

c)

Jazyk $L_3 = \{w \in \{a, b\}^* \mid w = w^R\}$ není regulární. Dokážeme to pomocí Myhill-Nerodovy věty. Pro spor předpokládejme, že regulární je. Pak $\exists n : |\Sigma^*/_{\sim_L}| = n$, tedy index relace prefixové ekvivalence je konečný. Uvažujme nyní o $n + 1$ slovech:

$$\epsilon, ab, a^2b^2, \ldots, a^nb^n$$

Uvažujme dále o dvou různých přirozených číslech i, j, kde $0 \leq i \leq n \wedge 0 \leq j \leq n$. Pokud $a^ib^i \sim_L a^jb^j$, pak $\forall w \in \Sigma^* : a^ib^iw \in L \iff a^jb^jw \in L$. Zvolme $w = b^ia^i$. Pak $a^ib^{2i}a^i = (a^ib^{2i}a^i)^R \Rightarrow a^ib^{2i}a^i \in L$, ale $a^jb^jb^ia^i \neq (a^jb^jb^ia^i)^R$, pročež $a^jb^jb^ia^i \notin L$.
Ukázali jsme, že žádné z těchto $n + 1$ slov nemůže být ve stejné třídě rozkladu. To je ale spor s naším předpokladem, že $|\Sigma^*/_{\sim_L}| = n$. Z toho plyne, že jazyk L není regulární.

Úloha 2

S využitím Myhill-Nerodovy věty dokažte, že následující jazyky jsou regulární.

(b) $L = \{w \in \{a,b\}^* \mid \#_a(w) < 3000\}$
(c) $L = \{w \in \{a,b\}^* \mid \#_a(w) \bmod 3 = \#_b(w) \bmod 2\}$

b)

Sestrojíme relaci pravé kongruence $\sim$ a ukážeme, že jazyk L je sjednocením některých tříd rozkladu $\Sigma^*/_\sim$.

$$u \sim v \iff (\#_a(u) = \#_a(v)) \vee (\#_a(u) \geq 3000 \wedge \#_a(v) \geq 3000)$$

Relace $\sim$ je zřejmě reflexivní, symterická i tranzitivní. Je to rovněž relace pravé kongruence.

$$\Sigma^*/_\sim \; = \{[x_i] \mid 0 \leq i \leq 2999\} \cup \{\{w \in \{a,b\}^* \mid \#_a(w) \geq 3000\}\}, \text{kde}$$

$$[x_i] = \{w \in \{a,b\}^* \mid \#_a(w) = i\}$$

Z toho plyne, že $|\Sigma^*/_\sim| = 3001$. Lze snadno nahlédnout, že $L = \bigcup\{[x_i] \mid 0 \leq i \leq 2999\}$, jazyk je tedy sjednocením 3000 tříd rozkladu.

c)

Sestrojíme relaci pravé kongruence $\sim$ a ukážeme, že jazyk L je sjednocením některých tříd rozkladu $\Sigma^*/_\sim$.

$$\begin{aligned}
&u \sim v \iff \\
&(\#_a(u) \mod 3 = \#_a(v) \mod 3 = 0 \wedge \#_b(u) \mod 2 = \#_b(v) \mod 2 = 0) \\
&\vee (\#_a(u) \mod 3 = \#_a(v) \mod 3 = 0 \wedge \#_b(u) \mod 2 = \#_b(v) \mod 2 = 1) \\
&\vee (\#_a(u) \mod 3 = \#_a(v) \mod 3 = 1 \wedge \#_b(u) \mod 2 = \#_b(v) \mod 2 = 0) \\
&\vee (\#_a(u) \mod 3 = \#_a(v) \mod 3 = 1 \wedge \#_b(u) \mod 2 = \#_b(v) \mod 2 = 1) \\
&\vee (\#_a(u) \mod 3 = \#_a(v) \mod 3 = 2 \wedge \#_b(u) \mod 2 = \#_b(v) \mod 2 = 0) \\
&\vee (\#_a(u) \mod 3 = \#_a(v) \mod 3 = 2 \wedge \#_b(u) \mod 2 = \#_b(v) \mod 2 = 1)
\end{aligned}$$

Relace $\sim$ je zřejmě reflexivní, symterická i tranzitivní. Je to rovněž relace pravé kongruence.

$$\Sigma^*/_\sim \; = \{[x_{i,j}] \mid i \in \{0,1,2\} \wedge j \in \{0,1\}\}, \text{kde}$$

$$[x_{i,j}] = \{w \in \{a,b\}^* \mid \#_a(w) \mod 3 = i \wedge \#_b(w) \mod 2 = j\}$$

Z toho plyne, že $|\Sigma^*/_\sim| = 6$. Lze snadno nahlédnout, že $L = [x_{0,0}] \cup [x_{1,1}]$, jazyk je tedy sjednocením 2 tříd rozkladu.

Úloha 3

Dokažte, že neexistuje úplný deterministický automat se 3 stavy, který akceptuje jazyk

(b) $L = \{w \in \{a,b\}^* \mid \#_a(w) \bmod 4 = 0\}$

b)

Předpokládejme, že existuje minimální deterministický konečný automat se třemi stavy, který přijímá L. Pak dle Myhill-Nerodovy věty platí, že $|\Sigma^*/_{\sim_L}| = 3$. Uvažujme o čtyřech slovech ϵ, a, aa, aaa. Pro libovolnou dvojici těchto slov a^i, a^j, kde $i \neq j \wedge i, j \in \{0, 1, 2, 3\}$ ukážeme, že $a^i \not\sim_L a^j$, což bude spor s naším předpokladem.
Pokud by platilo $a^i \sim_L a^j$, pak $\forall w \in \{a, b\}^* : a^i w \in L \iff a^j w \in L$. Zvolme $w = a^{4-i}$, pak $a^i a^{4-i} = a^4 \in L$, ale $a^j a^{4-i} \notin L$. Nyní ukážeme, proč $a^j a^{4-i} = a^{j+4-i} \notin L$. Určitě $j + 4 - i \equiv j - i \pmod 4$. Jelikož $i \neq j \wedge i, j \in \{0, 1, 2, 3\}$, pak určitě $j - i \not\equiv 0 \pmod 4$, tedy $a^j a^{4-i} \notin L$.
Určitě tedy $a^i \not\sim_L a^j$, tudíž náš předpoklad byl vyvrácen, neboť i minimální automat přijímající L musí mít více než 3 stavy. Úplný deterministický konečný automat se třemi stavy přijímající L tedy neexistuje.

Sekce 7: Uzávěrové vlastnosti regulárních jazyků

Úloha 1

Rozhodněte a dokažte, zda platí následující tvrzení

(b) $L_1 \cup L_2 \in \mathcal{L}_3 \Rightarrow L_1 \in \mathcal{L}_3 \vee L_2 \in \mathcal{L}_3$
(d) $L \in \mathcal{L}_3 \Rightarrow \Diamond L \in \mathcal{L}_3$, kde $\Diamond L = \{w \in L \mid w \in \{a,b\}^* \wedge \#_a(w) = \#_b(w) = 5\}$
(e) $L \in \mathcal{L}_3 \Rightarrow \Diamond L \in \mathcal{L}_3$, kde $\Diamond L = \{w \in L \mid w \in \{a,b\}^* \wedge \#_a(w) = \#_b(w)\}$

b)

Jistě neplatí. V rámci protipříkladu uvažujme o jazycích $L_1 = \{a^n b^n \mid n \geq 0\}, L_2 = co - L_1$. Jistě $L_1 \cup L_2 \in \mathcal{L}_3$, neboť $L_1 \cup L_2 = \Sigma^*$, avšak $L_1 \notin \mathcal{L}_3 \wedge L_2 \notin \mathcal{L}_3$, což lze ukázat pomocí Pumping lemmatu pro regulární jazyky.

d)

Jistě platí. Předpokládejme na okamžik, že $L = \Sigma^*$. Pak $\Diamond L$ obsahuje všechny takové řetězce nad abecedou $\{a,b\}$, které obsahují 5 symbolů a a 5 symbolů b. Pomocí vzorce pro permutace s opakováním můžeme ukázat, že takových řetězců existuje pouze $\frac{(5+5)!}{5!\cdot 5!} = 252$. Pokud je tedy L libovolný jazyk $L \subseteq \Sigma^*$, pak $\Diamond L$ je konečný (a tedy regulární) jazyk, neboť obsahuje nejvýše 252 řetězců.

e)

Jistě neplatí. V rámci protipříkladu uvažujme o jazyku $L = \Sigma^*$. Pak $\Diamond L = \{w \in \{a,b\}^* \mid \#_a(w) = \#_b(w)\}$, což není regulární jazyk. To dokážeme snadno pomocí Myhill-Nerodovy věty. Žádné dva různé řetězce $\epsilon, b, bb, bbb, \ldots$ totiž nemohou být v relaci $\sim$, neboť $b^k a^k \in \Diamond L$, ale $b^j a^k \notin \Diamond L$ (pro $j \neq k$), tedy $|\Sigma^*/_\sim| = \aleph_0$.

Úloha 2

Rozhodněte a dokažte, zda pro libovolný jazyk L a libovolný konečný jazyk K platí následující tvrzení:

(b) $L \in \mathcal{L}_3 \iff L \cup K \in \mathcal{L}_3$

b)

Jistě platí. Dokažme postupně jednotlivé směry ekvivalence.

$L \in \mathcal{L}_3 \Rightarrow L \cup K \in \mathcal{L}_3$: Platí, neboť K je dle zadání konečný, tudíž regulární jazyk. Třída regulárních jazyků je uzavřena na sjednocení.

$L \in \mathcal{L}_3 \Leftarrow L \cup K \in \mathcal{L}_3$: Platí, neboť můžeme psát, že $L = ((L \cup K) \setminus K) \cup (L \cap K)$. Z implikačního předpokladu víme, že $L \cup K$ je regulární jazyk, tedy i $(L \cup K) \setminus K$ je regulární jazyk, neboť třída regulárních jazyků je uzavřena na rozdíl. Současně víme, že $L \cap K$ je regulární jazyk, neboť průnik s konečným jazykem K může být nanejvýš konečný, tedy regulární jazyk. Z uzavřenosti třídy regulárních jazyků na sjednocení dále plyne, že rovněž $((L \cup K) \setminus K) \cup (L \cap K)$ je regulární jazyk. Jelikož $L = ((L \cup K) \setminus K) \cup (L \cap K)$, pak i L je regulární jazyk.

Následující obrázek znázorňuje, proč platí $L = ((L \cup K) \setminus K) \cup (L \cap K)$.

Jelikož jsme ukázali, že oba směry ekvivalence platí, je zřejmé, že platí celý původní výrok o libovolném jazyku L a o konečném jazyku K.

Sekce 8: Rozhodnutelné problémy pro regulární jazyky

Úloha 1

Dokažte, že následující problémy jsou rozhodnutelné.

(b) Pro daný konečný automat $A = (Q, \Sigma, \delta, q_0, F)$ je rozhodnutelné, zda platí toto tvrzení:

$$\forall n \in \mathbb{N} : \exists w \in L(A) : \#_a(w) > n$$

b)

Ať $A = (Q, \Sigma, \delta, q_0, F)$ je konečný automat. Můžeme předpokládat, že je deterministický, neboť každý konečný automat lze na deterministický převést. Zavedeme relace $\rightsquigarrow \subseteq Q \times Q$ a $\rightsquigarrow_a \subseteq Q \times Q$ následovně:

$$\forall q_1, q_2 \in Q : q_1 \rightsquigarrow q_2 \iff \exists \sigma \in \Sigma : ((q_1, \sigma), q_2) \in \delta$$

$$\forall q_1, q_2 \in Q : q_1 \rightsquigarrow_a q_2 \iff ((q_1, a), q_2) \in \delta$$

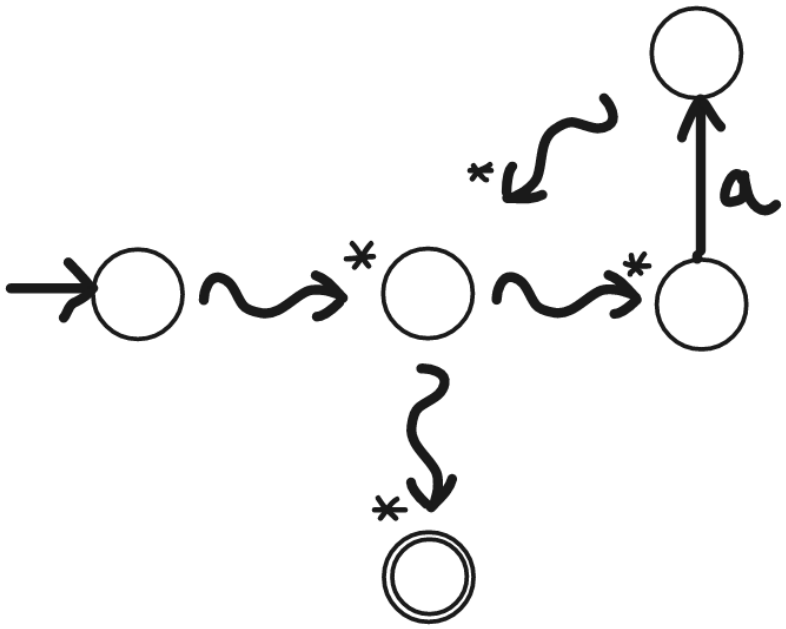
Výrok

$$\forall n \in \mathbb{N} : \exists w \in L(A) : \#_a(w) > n$$

je splněn právě tehdy, pokud

$$\exists q_1, q_2, q_3 \in Q \; \exists f \in F : q_0 \rightsquigarrow^* q_1 \wedge q_1 \rightsquigarrow^* q_2 \wedge q_2 \rightsquigarrow_a q_3 \wedge q_3 \rightsquigarrow^* q_1 \wedge q_1 \rightsquigarrow^* f.$$

Neformálně řečeno jsme vstupní problém převedli na problém, zda je z počátečního stavu q_0 dosažitelný cyklus, v němž je čten alespoň jednou symbol a, a zda je z některého stavu, jenž do cyklu patří, dosažitelný nějaký koncový stav f.



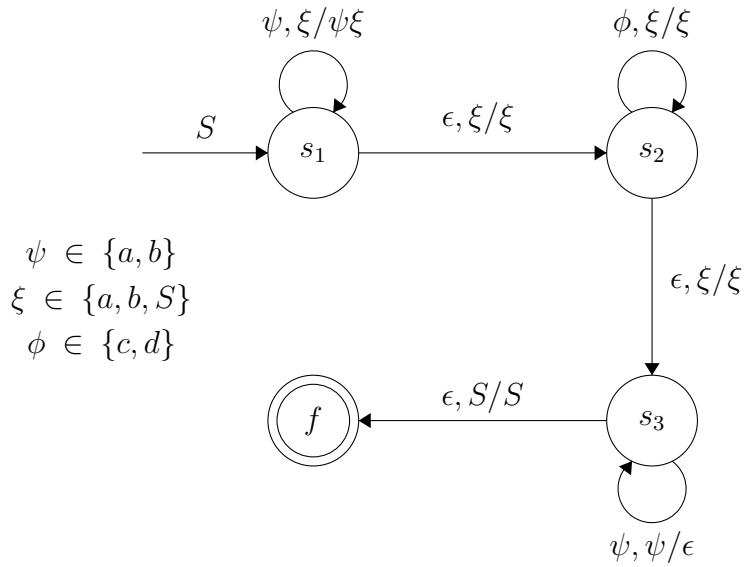
Sekce 9: Základní konstrukce pro bezkontextové jazyky

Úloha 1

Zkonstruujte zásobníkové automaty a bezkontextové gramatiky pro tyto jazyky:

(b) $L = \{ww'w^R \mid w \in \{a,b\}^* \wedge w' \in \{c,d\}^*\}$
(d) $L = \{a^i b^j c^k \mid 3 \cdot i \geq (k+j) \wedge i,j,k \geq 0\}$
(e) $L = \{a^i b^j c^k \mid 3 \cdot i \leq (k+j) \wedge i,j,k \geq 0\}$
(f) $L = \{w \in \{a,b,c\}^* \mid \#_a(w) = \#_b(w)\}$
(g) $L = \{w \in \{a,b\}^* \mid \#_a(w) > \#_b(w) + 3\}$
(h) $L = \{w \in \{a,b\}^* \mid \#_a(w) > 2 \cdot \#_b(w)\}$
(i) $L = \{w \in \{a,b,c\}^* \mid \#_a(w) > 2 \cdot (\#_b(w) + \#_c(w))\}$

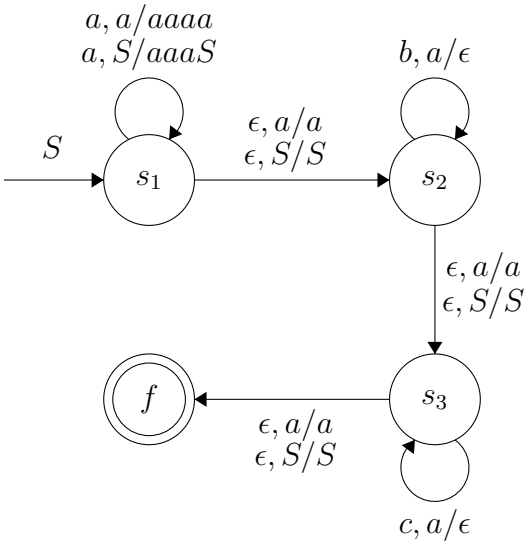
b)



$G = (\{S, C\}, \{a, b, c, d\}, P, S)$

$P : S \to aSa \mid bSb \mid \epsilon \mid C$
$C \to cC \mid dC \mid \epsilon$

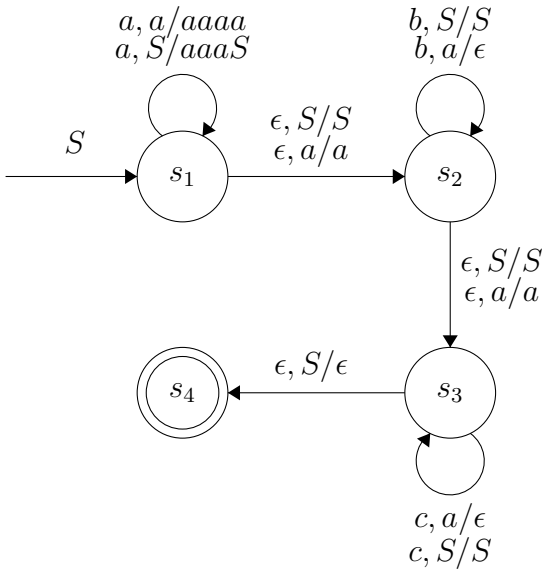
d)



$G = (\{S, K, C, B, A\}, \{a, b, c\}, P, S)$

$P : S \to ASCCC \mid AKBCC \mid AKBBC \mid AKBBB \mid \epsilon$
$K \to AKBBB \mid \epsilon$
$C \to c \mid \epsilon$
$B \to b \mid \epsilon$
$A \to aA \mid a$

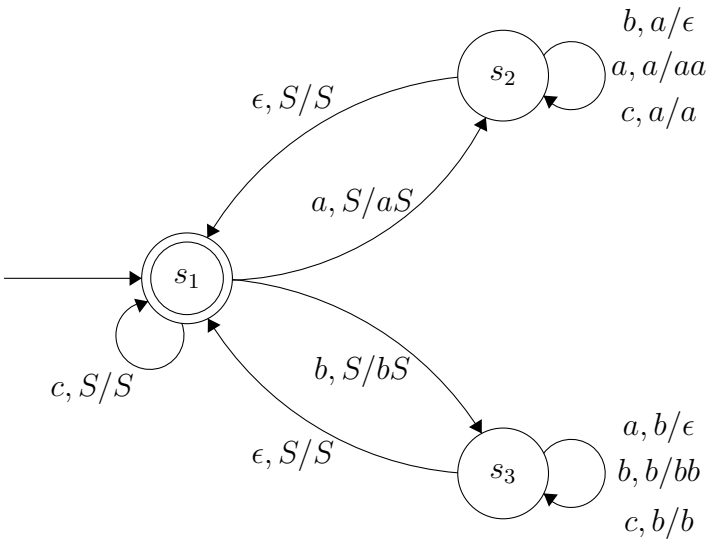
e)



$G = (\{A, B, C, K, S\}, \{a, b, c\}, P, S)$

$P : S \to ASCCC \mid AKBCC \mid AKBBC \mid AKBBB \mid B \mid C \mid BC \mid \epsilon$
$K \to AKBBB \mid \epsilon$
$C \to c \mid cC$
$B \to b \mid bB$
$A \to a \mid \epsilon$

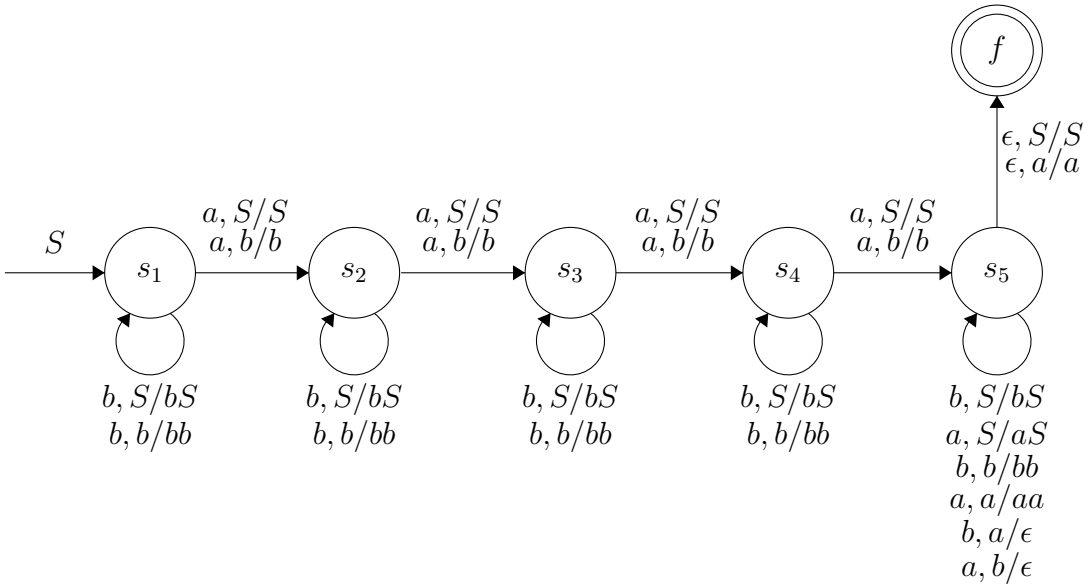
f)



$G = (\{S\}, \{a, b, c\}, P, S)$

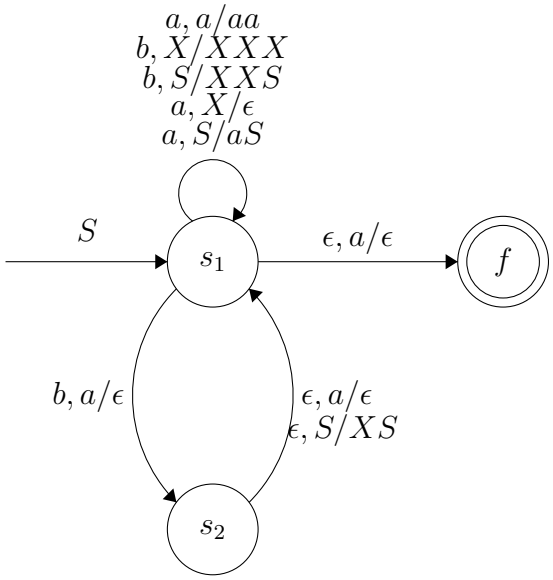
$P : S \to SaSbS \mid SbSaS \mid cS \mid \epsilon$

g)



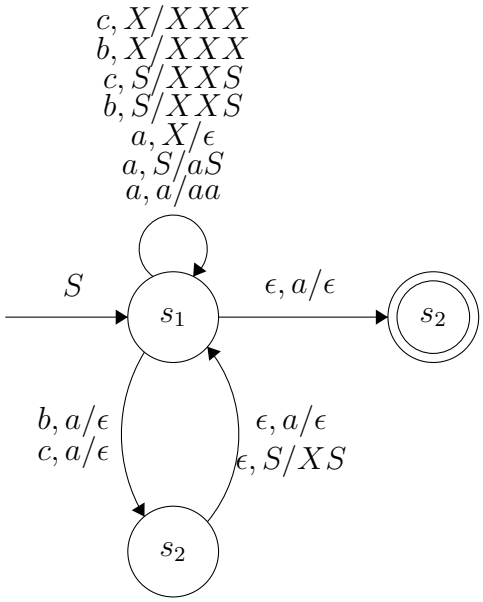
$S \to KaKaKaKaK$
$K \to KaKbK \mid KbKaK \mid aK \mid \epsilon$

h)



$S \to KaK$
$K \to KaKaKbK \mid KaKbKaK \mid KbKaKaK \mid aK \mid \epsilon$

i)



$S \to KaK$
$K \to KaKaKbK \mid KaKbKaK \mid KbKaKaK \mid KaKaKcK \mid KaKcKaK \mid KcKaKaK \mid aK \mid \epsilon$

Úloha 2

Pro následující gramatiky navrhněte (rozšířený) zásobníkový automat, který provádí syntaktickou analýzu

- shora dolů,
- zdola nahoru,

V obou případech proveďte analýzu zadaného slova w.

(b) $G = (\{S, A, B\}, \{a, b, c\}, P, S)$, kde P obsahuje pravidla:
$S \to \epsilon \mid ASBB$
$A \to a \mid Aa$
$B \to b \mid c$
Slovo $w = aabc$.

b)

Analýza shora dolů

$M = (\{q\}, \Sigma, \Sigma \cup N, \delta, q, S, \{\})$

$\delta : \delta(q, a, a) = \{(q, \epsilon)\}$
$\delta(q, b, b) = \{(q, \epsilon)\}$
$\delta(q, c, c) = \{(q, \epsilon)\}$
$\delta(q, \epsilon, S) = \{(q, \epsilon), (q, ASBB)\}$
$\delta(q, \epsilon, A) = \{(q, a), (q, Aa)\}$
$\delta(q, \epsilon, B) = \{(q, b), (q, c)\}$

Analýza slova $w = aabc$:

$(q, aabc, S) \vdash (q, aabc, ASBB) \vdash (q, aabc, AaSBB) \vdash (q, aabc, aaSBB) \vdash (q, abc, aSBB) \vdash (q, bc, SBB) \vdash (q, bc, BB) \vdash (q, bc, bB) \vdash (q, c, B) \vdash (q, c, c) \vdash (q, \epsilon, \epsilon)$

Automat přijímá vyprázdněním zásobníku.

Analýza zdola nahoru

$M = (\{q, r\}, \Sigma, \Sigma \cup N \cup \{\#\}, \delta, q, \#, \{r\})$

$\delta : \delta(q, \epsilon, \#S) = \{(r, \epsilon)\}$
$\delta(q, a, \epsilon) = \{(q, a)\}$
$\delta(q, b, \epsilon) = \{(q, b)\}$
$\delta(q, c, \epsilon) = \{(q, c)\}$
$\delta(q, \epsilon, \epsilon) = \{(q, S)\}$
$\delta(q, \epsilon, ASBB) = \{(q, S)\}$
$\delta(q, \epsilon, a) = \{(q, A)\}$
$\delta(q, \epsilon, Aa) = \{(q, A)\}$
$\delta(q, \epsilon, b) = \{(q, B)\}$
$\delta(q, \epsilon, c) = \{(q, B)\}$

Analýza slova $w = aabc$:

$(q, aabc, \#) \vdash (q, abc, \#a) \vdash (q, abc, \#A) \vdash (q, bc, \#Aa) \vdash (q, bc, \#A) \vdash (q, bc, \#AS) \vdash (q, c, \#ASb) \vdash (q, c, \#ASB) \vdash (q, \epsilon, \#ASBc) \vdash (q, \epsilon, \#ASBB) \vdash (q, \epsilon, \#S) \vdash (r, \epsilon, \epsilon)$

Vrchol zásobníku je u analýzy zdola nahoru uveden vpravo.

Sekce 10: Transformace bezkontextových gramatik

Úloha 1

Odstraňte ϵ-pravidla v následujících gramatikách.

(b) $G = (\{S, A, B, C, D\}, \{a, b, c\}, P, S)$
$P : S \to ABC$
$A \to Ab \mid BC \mid b$
$B \to bB \mid a \mid Ab \mid \epsilon$
$C \to cD \mid cS \mid Aa \mid \epsilon$
$D \to SSS \mid bc$

b)

Spočtěme množinu N_ϵ: $N_\epsilon^0 = \{\}, N_\epsilon^1 = \{B, C\}, N_\epsilon^2 = \{A, B, C\}, N_\epsilon^3 = \{A, B, C, S\}, N_\epsilon^4 = \{A, B, C, S, D\} = N_\epsilon$.

Nová gramatika $G' = (\{S', S, A, B, C, D\}, \{a, b, c\}, P', S')$ obsahuje pravidla

$P' : S' \to S \mid \epsilon$
$S \to ABC \mid AB \mid BC \mid AC \mid A \mid B \mid C$
$A \to Ab \mid b \mid BC \mid B \mid C$
$B \to bB \mid b \mid a \mid Ab$
$C \to cD \mid c \mid cS \mid Aa \mid a$
$D \to SSS \mid SS \mid S \mid bc$

Úloha 2

Odstraňte jednoduchá pravidla v následujících gramatikách a pak je převeďte do Chomského normální formy.

(b) $G = (\{S, A, B, C, D, E, F\}, \{a, b\}, P, S)$, kde P obsahuje pravidla:
$S \to A \mid F$
$A \to bS$
$B \to Sa \mid a \mid S$
$C \to aD \mid S$
$D \to Ba$
$E \to aAS \mid E$
$F \to SAb$

b)

Spočteme množiny N_X, kde $X \in N$. $N_S = \{S, A, F\}, N_A = \{A\}, N_B = \{B, S, A, F\}, N_C = \{C, S, A, F\}, N_D = \{D\}, N_E = \{E\}, N_F = \{F\}$

Vytvořme gramatiku $G' = (\{S, A, B, C, D, E, F\}, \{a, b\}, P', S)$ s pravidly

$P' : S \to bS \mid SAb$
$A \to bS$

$B \to Sa \mid a \mid bS \mid SAb$
$C \to aD \mid bS \mid SAb$
$D \to Ba$
$E \to aAS$
$F \to SAb$

Dopusťme se převodu G' do Chomského normální formy, vytvořme tedy gramatiku $G'' = (\{S, A, B, C, D, E, F, \langle a\rangle, \langle b\rangle, \langle Ab\rangle, \langle AS\rangle\}, \{a, b\}, P'', S)$ s pravidly:

$P' : S \to \langle b\rangle S \mid S\langle Ab\rangle$
$A \to \langle b\rangle S$
$B \to S\langle a\rangle \mid a \mid \langle b\rangle S \mid S\langle Ab\rangle$
$C \to \langle a\rangle D \mid \langle b\rangle S \mid S\langle Ab\rangle$
$D \to B\langle a\rangle$
$E \to \langle a\rangle\langle AS\rangle$
$F \to S\langle Ab\rangle$
$\langle a\rangle \to a$
$\langle b\rangle \to b$
$\langle Ab\rangle \to A\langle b\rangle$
$\langle AS\rangle \to AS$

Úloha 3

Odstraňte levou rekurzi v následujících gramatikách.

(b) $G = (\{S, A, B\}, \{a, b\}, P, S)$, kde P obsahuje pravidla
$S \to Ab \mid a \mid bB$
$A \to BSa \mid ab \mid SBa$
$B \to bB \mid BaB \mid Sa$

b)

Nejprve rozgenerujeme pravidlo $A \to SBa$

$S \to Ab \mid a \mid bB$
$A \to BSa \mid ab \mid AbBa \mid aBa \mid bBBa$
$B \to bB \mid BaB \mid Sa$

Povšimněme si, že se nám u pravidla $A \to AbBa$ objevila přímá rekurze. Nyní ji odstraníme.

$S \to Ab \mid a \mid bB$
$A \to BSa \mid ab \mid aBa \mid bBBa \mid BSaA' \mid abA' \mid aBaA' \mid bBBaA'$
$A' \to bBa \mid bBaA'$
$B \to bB \mid BaB \mid Sa$

Nyní rozgenerujeme pravidlo $B \to Sa$

$S \to Ab \mid a \mid bB$
$A \to BSa \mid ab \mid aBa \mid bBBa \mid BSaA' \mid abA' \mid aBaA' \mid bBBaA'$
$A' \to bBa \mid bBaA'$

$B \to bB \mid BaB \mid Aba \mid aa \mid bBa$

Pokračujeme rozgenerováním pravidla $B \to Aba$

$S \to Ab \mid a \mid bB$
$A \to BSa \mid ab \mid aBa \mid bBBa \mid BSaA' \mid abA' \mid aBaA' \mid bBBaA'$
$A' \to bBa \mid bBaA'$
$B \to bB \mid BaB \mid BSaba \mid abba \mid aBaba \mid bBBaba \mid BSaA'ba \mid abA'ba \mid aBaA'ba \mid bBBaA'ba \mid aa \mid bBa$

Nyní odstraňme levou rekurzi u neterminálu B.

$S \to Ab \mid a \mid bB$
$A \to BSa \mid ab \mid aBa \mid bBBa \mid BSaA' \mid abA' \mid aBaA' \mid bBBaA'$
$A' \to bBa \mid bBaA'$
$B \to bB \mid abba \mid aBaba \mid bBBaba \mid abA'ba \mid aBaA'ba \mid bBBaA'ba \mid aa \mid bBa \mid bBB' \mid abbaB' \mid$
$\quad aBabaB' \mid bBBabaB' \mid abA'baB' \mid aBaA'baB' \mid bBBaA'baB' \mid aaB' \mid bBaB'$
$B' \to aB \mid Saba \mid SaA'ba \mid aBB' \mid SabaB' \mid SaA'baB'$

Sekce 11: Jednoznačnost a determinismus bezkontextových gramatik

Úloha 1

Navrhněte jednoznačnou gramatiku pro jazyk

(b) $L = \{a^n b^n \mid n > 0\} \cup \{a^n b^{2m} \mid m, n > 0\}$

b)

$G = (\{S, X, C, A\}, \{a, b\}, P, S)$

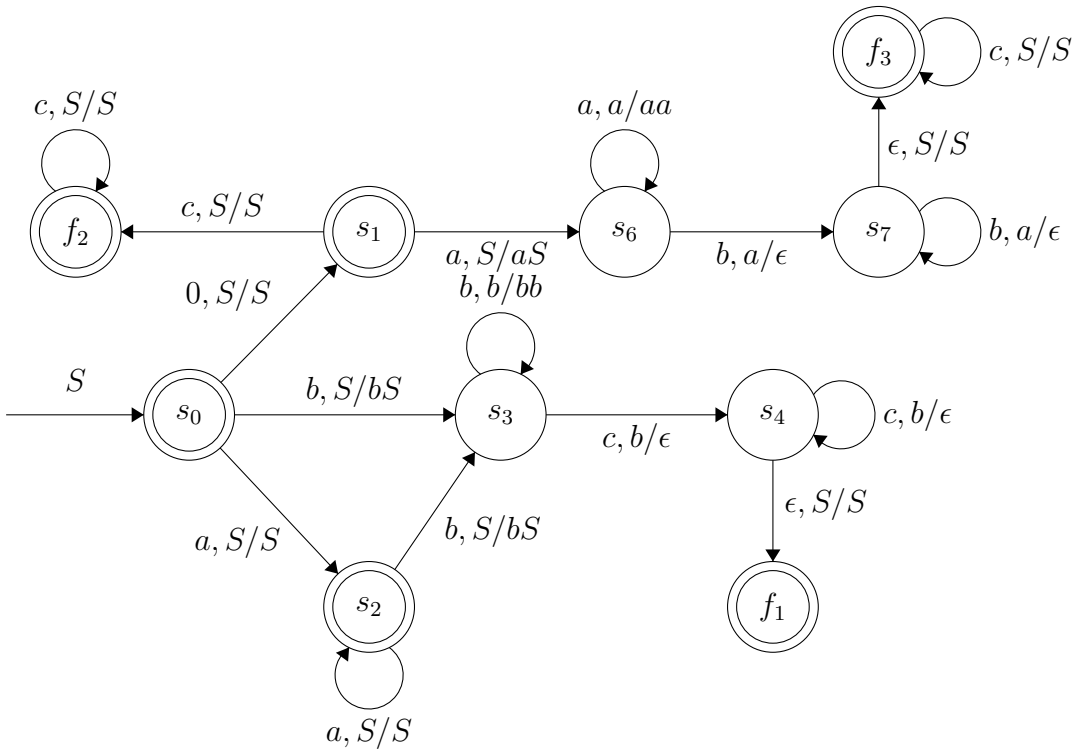
$P : S \to aXb \mid Cbb$
$X \to aaXbb \mid \epsilon$
$C \to Cbb \mid aA$
$A \to aA \mid \epsilon$

Úloha 2

Navrhněte deterministický zásobníkový automat pro jazyk

(b) $L = \{0\} \cdot L_1 \cup L_2$, kde $L_1 = \{a^n b^n c^m \mid n, m \geq 0\}$ a $L_2 = \{a^m b^n c^n \mid n, m \geq 0\}$

b)



Sekce 12: Fix-point algoritmy pro bezkontextové gramatiky

Úloha 1

Zkonstruujte algoritmus, který pro danou BG $G = (N, \Sigma, S, P)$ spočítá následující množiny:

(b) $\mathbb{A} = \{A \in N \mid \forall k \in \mathbb{N}\ \exists w \in \Sigma^* : A \Rightarrow^+ w \wedge \#_a(w) > k\}$

b)

Konstruujeme množinu takových neterminálů A vstupní gramatiky G, pro které platí, že pro každý řetězec terminálů w_1, který lze z A zderivovat, jsme schopni nalézt jiný řetězec w_2, který lze z A rovněž zderivovat, a který obsahuje více symbolů a než řetězec w_1, tj. počet symbolů a v řetězcích derivovaných z A není shora omezen.

Vstup: $G = (N, \Sigma, P, S), a \in \Sigma$
Výstup: Množina $\mathbb{A}$ všech neterminálů gramatiky G dle výše uvedené specifikace
1. Zkonstruujeme množinu $N_t = \{A \in N \mid \exists w \in \Sigma^* : A \Rightarrow^* w\}$ pomocí algoritmu pevného bodu.
2. Zkonstruujeme množinu $N_a = \{A \in N \mid \exists w_1, w_2 \in \Sigma^* : A \Rightarrow^* w_1 a w_2\}$ pomocí algoritmu pevného bodu.
3. Zavedeme relaci ξ následovně:

$$A\ \xi\ B \iff A, B \in N\ \wedge\ \exists (A \to \alpha B \beta) \in P : \alpha, \beta \in (\Sigma \cup N_t)^*.$$

4. Zavedeme relaci ξ_a následovně:

$$\begin{aligned} &A\ \xi_a\ B \iff A, B \in N\ \wedge\ \exists (A \to \alpha B \beta) \in P : \\ &(\alpha \in (\Sigma \cup N_t)^* \wedge \beta \in (\Sigma \cup N_t)^* (N_a \cup \{a\}) (\Sigma \cup N_t)^*) \vee \\ &(\beta \in (\Sigma \cup N_t)^* \wedge \alpha \in (\Sigma \cup N_t)^* (N_a \cup \{a\}) (\Sigma \cup N_t)^*) \end{aligned}$$

5. Spočteme reflexivně-tranzivitní uzávěr ξ^* relace ξ.
6. Zkonstruujeme množinu

$$\mathbb{A} = \{A \in N \mid \exists B, C, D \in N_t\ : A\ \xi^*\ B \wedge B\ \xi^*\ C \wedge C\ \xi_a\ D \wedge D\ \xi^*\ B\}.$$

7. Vrátíme $\mathbb{A}$ jakožto výstup.

Vysvětlení: Aby počet symbolů a v řetězcích derivovatelných z neterminálu A nebyl shora omezen, musí platit, že z A je dosažitelný cyklus $B \rightsquigarrow B$, který ve svém běhu generuje alespoň jeden terminál a. Současně musí platit, že B není zbytečný symbol, tedy že je možné jej zderivovat na řetězec terminálů. Toto opatření je nutné, protože kupříkladu v gramatice $G = (\{S\}, \{a\}, \{S \to aSa\}, S)$ je z S dosažitelný cyklus generující ve svém běhu a, ale $L(G) = \{\}$, tudíž $S \notin \mathbb{A}$.

Sekce 13: Pumping lemma pro bezkontextové jazyky

Úloha 1

Pomocí Pumping lemmatu dokažte, že následující jazyky nejsou bezkontextové.

(b) $L = \{uww^Ru \mid u, w \in \{a, b\}^*\}$
(c) $L = \{a^i b^{2i} c^{3i} \mid i \geq 0\}$

b)

Ať k je konstanta Pumping lemmatu. Vybereme slovo $z = a^k b^k a^k b^k$ (zde $w = w^R = \epsilon$). Určitě platí $z \in L$ a rovněž $|z| = 4 \cdot k \geq k$. Uvažujeme o všech rozděleních $z = uvwxy$, kde $|vx| > 0 \wedge |vwx| \leq k$. Mohou nastat následující případy:
Řetězec vx obsahuje pouze jediný typ symbolu (a nebo b). Potom volbou $i = 0$ docílíme toho, že $uv^iwx^iy \notin L$, neboť dojde k narušení vztahů mezi částmi $a^k b^k$ a $a^k b^k$ řetězce z, které mají být totožné.
Řetězec vx obsahuje dva typy symbolů (a i b). Potom volbou $i = 2$ docílíme toho, že $uv^iwx^iy \notin L$, neboť dojde k narušení vztahů mezi částmi $a^k b^k$ a $a^k b^k$ řetězce z, které mají být totožné.

c)

Ať k je konstanta Pumping lemmatu. Vybereme slovo $z = a^k b^{2k} c^{3k}$. Určitě platí $z \in L$ a rovněž $|z| = 6 \cdot k \geq k$. Uvažujeme o všech rozděleních $z = uvwxy$, kde $|vx| > 0 \wedge |vwx| \leq k$. Mohou nastat následující případy:
Řetězec vx obsahuje pouze jediný typ symbolu (a, b nebo c, označme jej jako σ). Potom volbou $i = 2$ docílíme toho, že $uv^iwx^iy \notin L$, neboť dojde k narušení vztahů mezi počtem symbolů σ a zbylých symbolů, které vyžaduje definice jazyka.
Řetězec vx obsahuje dva typy symbolů (zbylý třetí symbol označme σ). Potom volbou $i = 2$ opět docílíme toho, že $uv^iwx^iy \notin L$, neboť dojde k narušení vztahů mezi počtem symbolů σ a počtem zbylých symbolů, které vyžaduje definice jazyka.

Úloha 2

Dokažte, že následující jazyky splňují pravou stranu Pumping lemmatu pro bezkontextové jazyky.

(b) $L = \{w \in \{a, b\}^* \mid \#_a(w) = \#_b(w)\}$

b)

Zvolme konstantu Pumping lemmatu $k = 2$. Pak veškerá slova $z \in L$, kde $|z| \geq 2$, můžeme zapsat ve tvaru $z = a_1 a_2 \ldots a_n$, kde $n \geq 2$ a $\forall i \in \langle 0; n\rangle \cap \mathbb{N} : a_i \in \Sigma$. Je zřejmé, že n je sudé číslo, neboť z obsahuje stejný počet symbolů a a b. Potom určitě existuje takové $j \in \langle 0; n-1\rangle \cap \mathbb{N}$, kde $a_j \neq a_{j+1}$, tedy že v řetězci z spolu určitě někde sousedí dva různé symboly.
Zvolme tedy dekompizici z tak, že $v = a_j, w = \epsilon, x = a_{j+1}$, tj. $z = ua_j a_{j+1} y$. Potom pro libovolné $i \geq 0 : uv^iwx^iy \in L$, neboť a_j a a_{j+1} jsou různé symboly, tudíž rovnost počtu a a b zůstane zachována.

Sekce 14: Uzávěrové vlastnosti bezkontextových gramatik

Úloha 1

Rozhodněte a dokažte, zda platí následující tvrzení.

(c) $\exists L_1, L_2 \in \mathcal{L}_2 \setminus \mathcal{L}_3 : L_1 \cdot L_2 \in \mathcal{L}_3$
(d) $\exists L_1, L_2 \in \mathcal{L}_1 \setminus \mathcal{L}_2 : L_1 \cap L_2 \in \mathcal{L}_2 \setminus \mathcal{L}_3$
(e) $\exists L_1, L_2 \in \mathcal{L}_1 \setminus \mathcal{L}_2 : L_1 \cup L_2 \in \mathcal{L}_2 \setminus \mathcal{L}_3$

c)

Platí. Uvažujme o jazycích

$$L_1 = \{w \in \{a,b\}^* \mid \#_a(w) \geq \#_b(w)\}$$
$$L_2 = \{w \in \{a,b\}^* \mid \#_a(w) \leq \#_b(w)\}$$

Pak platí

$$L_1 \cdot L_2 = \Sigma^*$$

kde $\Sigma = \{a,b\}$, $L_1 \cdot L_2$ je tedy regulární jazyk. Jazyky L_1 a L_2 jsou jistě bezkontextové neregulární.

d)

Platí. Uvažujme o jazycích

$$L_1 = \{c^n d^n e^n \mid n > 0\} \cup \{a^n b^n \mid n > 0\}$$
$$L_2 = \{f^n g^n h^n \mid n > 0\} \cup \{a^n b^n \mid n > 0\}$$

Pak platí

$$L_1 \cap L_2 = \{a^n b^n \mid n > 0\}$$

Jazyk $L_1 \cap L_2$ je určitě bezkontextový neregulární. Jazyky L_1 a L_2 lze zřejmě konstruovat pomocí LOA a současně se jednoduše dá pomocí Pumping lemmatu dokázat, že nejsou bezkontextové.

e)

Platí. Uvažujme o jazycích

$$L_1 = \{a^n b^n c^n \mid n > 0\} \cup \{c^n b^n a^m \mid m,n > 0\}$$
$$L_2 = \{a^n b^n c^m \mid m,n > 0\} \cup \{c^n b^n a^n \mid n > 0\}$$

Pak platí

$$L_1 \cup L_2 = \{a^n b^n c^m \mid m,n > 0\} \cup \{c^n b^n a^m \mid m,n > 0\}$$

Jazyk $L_1 \cup L_2$ je jistě bezkontextový neregulární. Jazyky L_1 a L_2 lze konstruovat pomocí LOA, jsou tedy kontextové. Lze také pomocí Pumping lemmatu jednoduše dokázat, že jsou nebezkontextové.

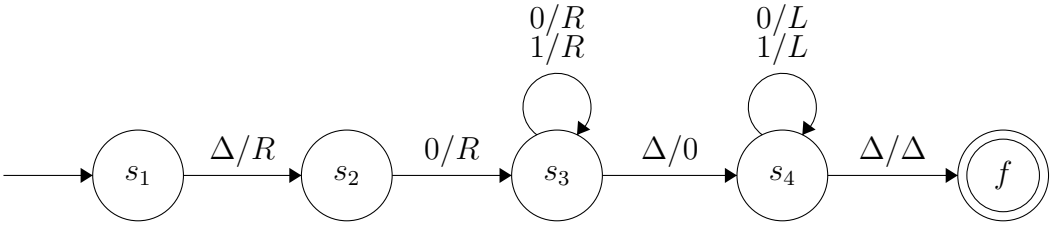
Sekce 15: Základní konstrukce Turingových strojů

Úloha 1

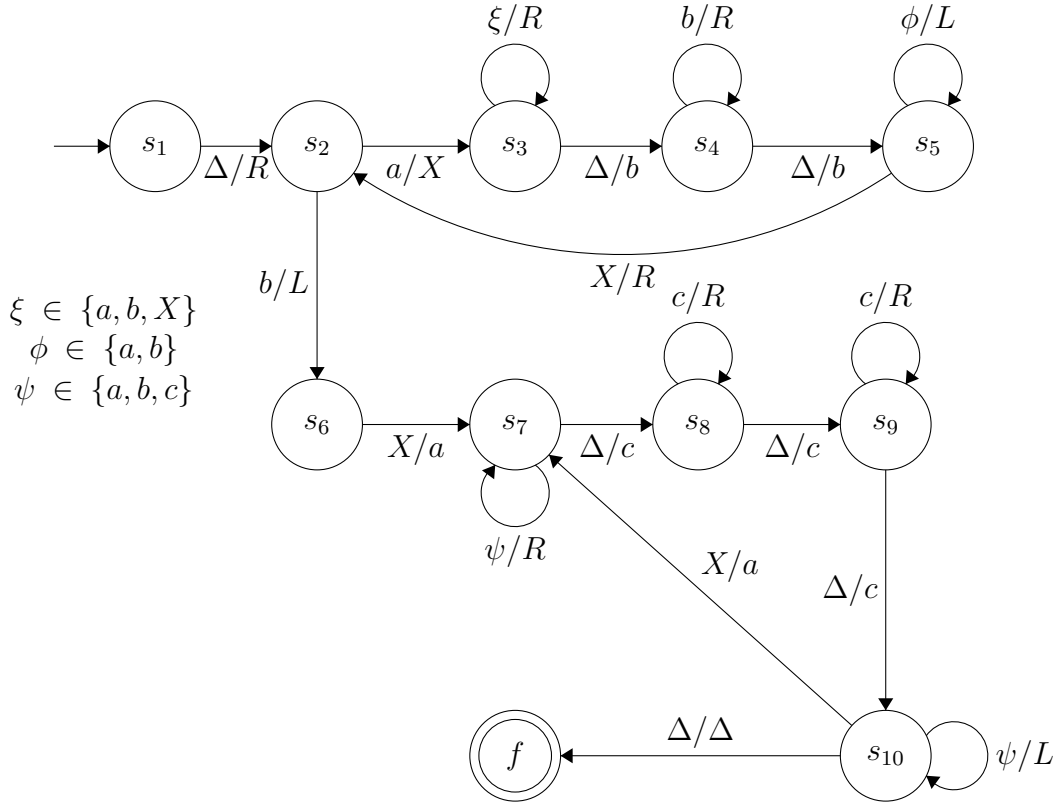
Zkonstruujte (a popište přechodovým diagramem) TS, který

(b) převádí vstupní konfiguraci pásky $\underline{\Delta}x\Delta^\omega$, kde $x \in \{0\} \cdot \{0,1\}^*$, na $\underline{\Delta}y\Delta^\omega$, kde x a y reprezentují čísla u_x a u_y v binárním kódu taková, že $u_y = 2 \cdot u_x$
(c) převádí vstupní konfiguraci pásky $\underline{\Delta}a^n\Delta^\omega$, kde $n > 0$, na $\underline{\Delta}a^n b^{2n} c^{3n}\Delta^\omega$
(d) akceptuje jazyk $L = \{w \in \{a,b,c\}^* \mid \#_a(w) = \#_b(w) = \#_c(w)\}$
(e) akceptuje jazyk $L = \{w \in \{a,b,c\}^* \mid \#_a(w) > 2 \cdot \#_b(w)\}$

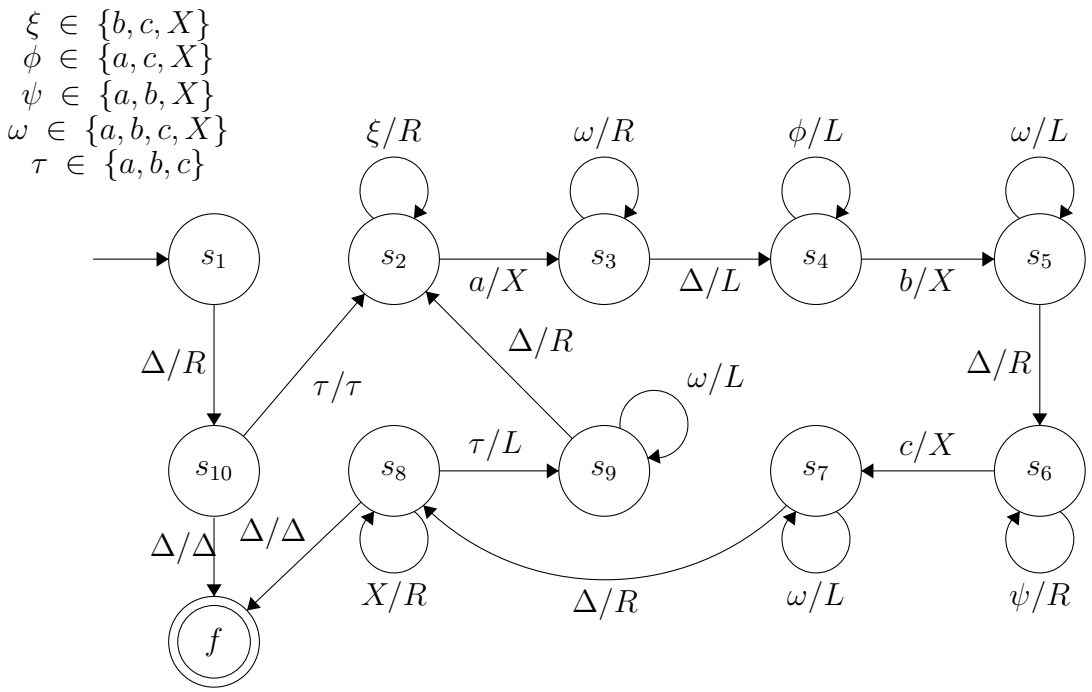
b)



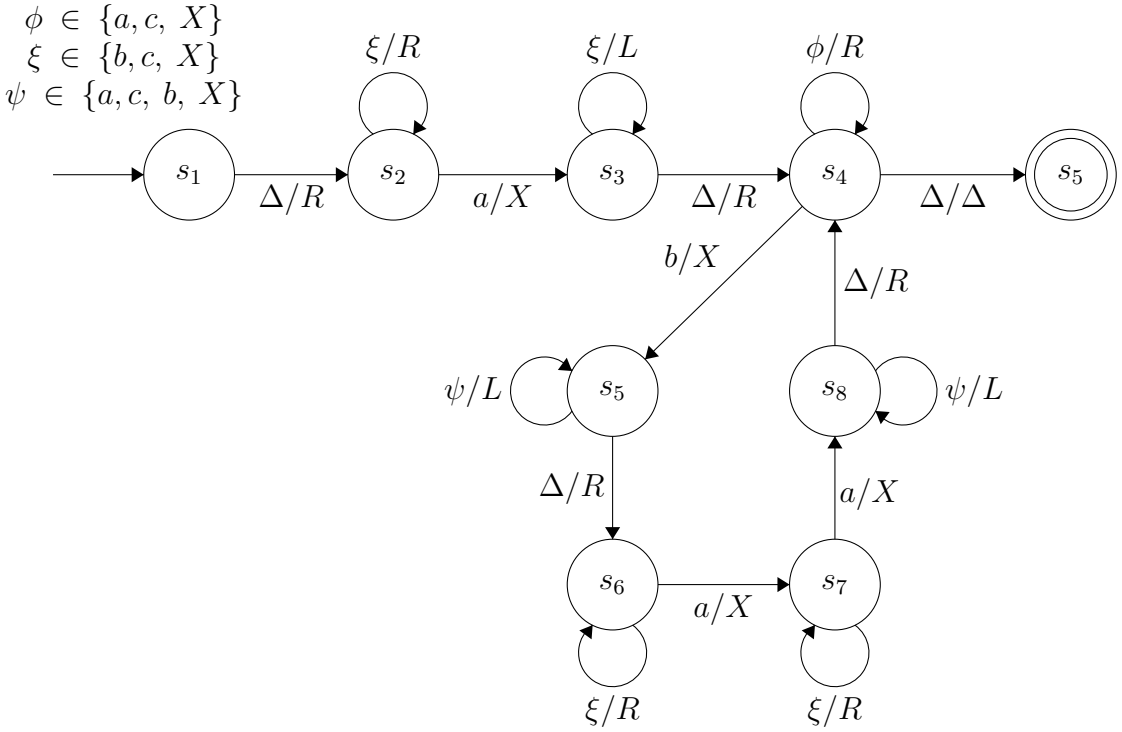
c)



d)


$\xi \in \{b, c, X\}$
$\phi \in \{a, c, X\}$
$\psi \in \{a, b, X\}$
$\omega \in \{a, b, c, X\}$
$\tau \in \{a, b, c\}$
s_1
s_2
s_3
s_4
s_5
s_6
s_7
s_8
s_9
s_{10}
f
ξ/R
ω/R
ϕ/L
ω/L
a/X
Δ/L
b/X
Δ/R
Δ/R
τ/τ
ω/L
Δ/R
τ/L
c/X
Δ/Δ
Δ/Δ
X/R
Δ/R
ω/L
ψ/R

e)


$\phi \in \{a, c, X\}$
$\xi \in \{b, c, X\}$
$\psi \in \{a, c, b, X\}$
s_1
s_2
s_3
s_4
s_5
s_5
s_6
s_7
s_8
ξ/R
ξ/L
ϕ/R
Δ/R
a/X
Δ/R
Δ/Δ
b/X
Δ/R
ψ/L
ψ/L
Δ/R
a/X
a/X
ξ/R
ξ/R

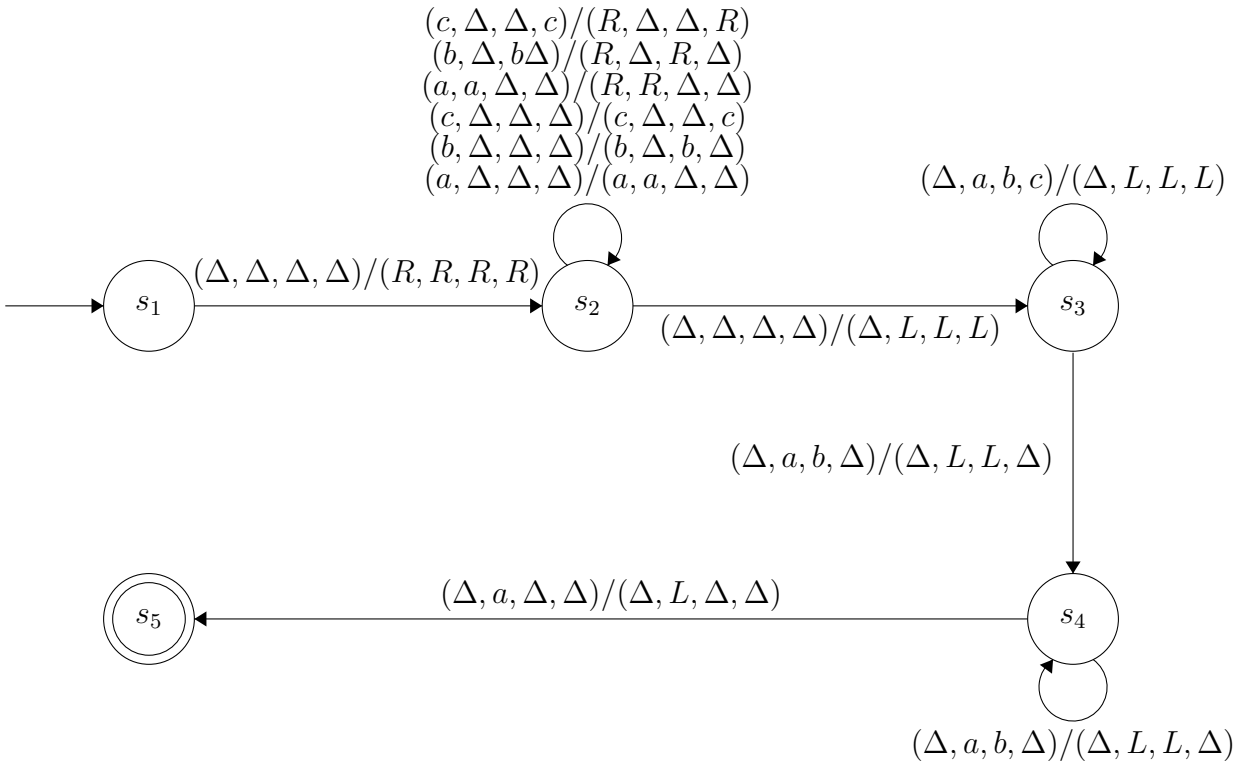
Sekce 16: Konstrukce vícepáskových Turingových strojů

Úloha 1

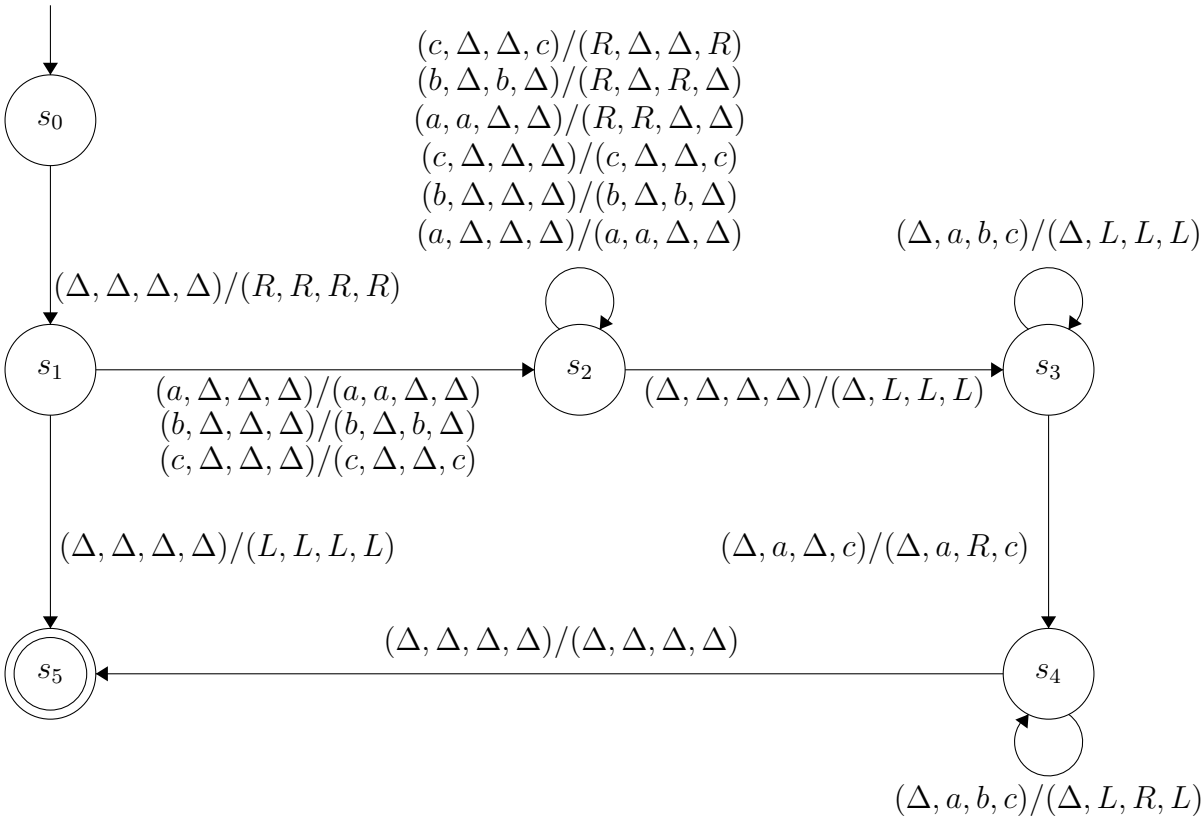
Zkonstruujte vícepáskový TS (a popište přechodovým diagramem), který akceptuje jazyk

(b) $L = \{w \in \{a, b, c\}^* \mid \#_a(w) > \#_b(w) > \#_c(w)\}$
(c) $L = \{w \in \{a, b, c\}^* \mid \#_a(w) = 2 \cdot \#_b(w) = \#_c(w)\}$

b)



c)


s_0
s_1
s_2
s_3
s_4
s_5
$(c, \Delta, \Delta, c)/(R, \Delta, \Delta, R)$
$(b, \Delta, b, \Delta)/(R, \Delta, R, \Delta)$
$(a, a, \Delta, \Delta)/(R, R, \Delta, \Delta)$
$(c, \Delta, \Delta, \Delta)/(c, \Delta, \Delta, c)$
$(b, \Delta, \Delta, \Delta)/(b, \Delta, b, \Delta)$
$(a, \Delta, \Delta, \Delta)/(a, a, \Delta, \Delta)$
$(\Delta, a, b, c)/(\Delta, L, L, L)$
$(\Delta, \Delta, \Delta, \Delta)/(R, R, R, R)$
$(a, \Delta, \Delta, \Delta)/(a, a, \Delta, \Delta)$
$(b, \Delta, \Delta, \Delta)/(b, \Delta, b, \Delta)$
$(c, \Delta, \Delta, \Delta)/(c, \Delta, \Delta, c)$
$(\Delta, \Delta, \Delta, \Delta)/(L, L, L, L)$
$(\Delta, a, \Delta, c)/(\Delta, a, R, c)$
$(\Delta, \Delta, \Delta, \Delta)/(\Delta, \Delta, \Delta, \Delta)$
$(\Delta, a, b, c)/(\Delta, L, R, L)$

Sekce 17: Důkazy (částečné) rozhodnutelnosti

Úloha 1

Pro každý níže uvedený jazyk rozhodněte, zda je rozhodnutelný či pouze částečně rozhodnutelný. Uveďte hlavní myšlenku konstrukce TS, který tuto (částečnou) rozhodnutelnost dokazuje

(b) $L = \{\langle M_1 \rangle \# \langle M_2 \rangle \mid L(M_1) \cap L(M_2) \neq \{\}\}$
(c) $L = \{\langle M \rangle \# \langle w \rangle \# \langle n \rangle \mid M$ neakceptuje w do n kroků $\}$
(d) $L = \{\langle M \rangle \# \langle n \rangle \mid \exists w \in \Sigma^* : M$ akceptuje w do n kroků $\}$
(e) $L = \{\langle M \rangle \# \langle n \rangle \mid \forall w \in \Sigma^* : \texttt{steps}(M, w) \leq n\}$
(f) $L = \{\langle M \rangle \# \langle n \rangle \mid \forall w \in \Sigma^* : \texttt{steps}(M, w) \geq n\}$

b)

Nedeterministická varianta:

Jazyk L je částečně rozhodnutelný. Uvažujme o vícepáskovém Turingově stroji M přijímajícím L, který pracuje následovně:

Na první pásce má M uložený svůj vstup. Pokud tento vstup není ve tvaru $\langle M_1 \rangle \# \langle M_2 \rangle$, kde $\langle M_1 \rangle, \langle M_2 \rangle$ jsou validní kódy Turingových strojů, stroj M ihned odmítne, jinak pokračuje. Řekněme, že stroj M_1 má abecedu Σ a stroj M_2 má abecedu Ψ. Stroj M nedeterministicky vygeneruje slovo $w \in (\Sigma \cap \Psi)^*$ a spustí simulaci stroje M_1 na tomto slově. Pokud simulace skončí odmítnutím, stroj M také odmítne. Pokud simulace cyklí, stroj M rovněž cyklí. Pokud simulace skončí akceptováním, stroj M spustí simulaci stroje M_2 na slově w. Pokud tato simulace skončí odmítnutím, odmítne i stroj M. Pokud tato simulace cyklí, cyklí i stroj M. Pokud tato simulace skončí akceptováním, akceptuje i M, neboť w je slovo ležící v jazyce $L(M_1) \cap L(M_2)$, tedy tento průnik není prázdný.

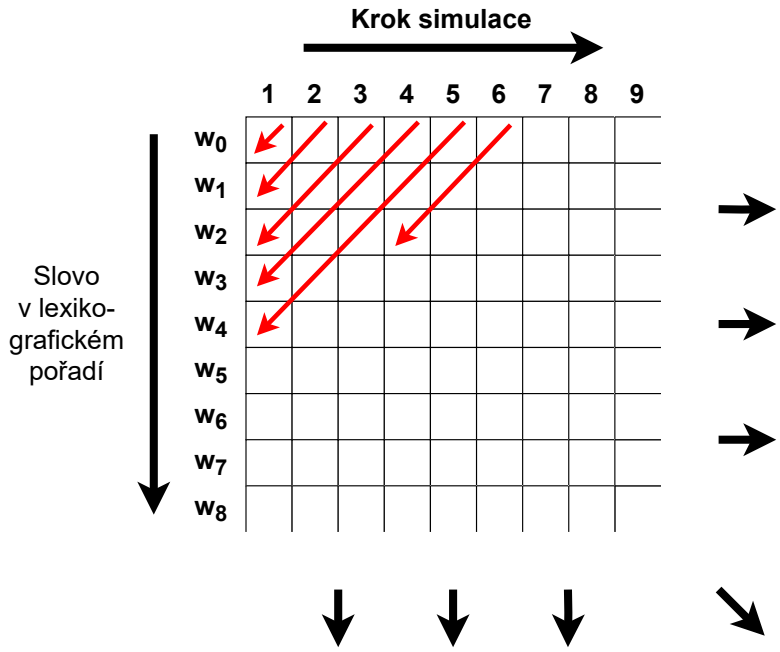
Deterministická varianta:

Jazyk L je částečně rozhodnutelný. Uvažujme o vícepáskovém Turingově stroji M přijímajícím L, který pracuje následovně:

Na první pásce má M uložený svůj vstup. Pokud tento vstup není ve tvaru $\langle M_1 \rangle \# \langle M_2 \rangle$, kde $\langle M_1 \rangle, \langle M_2 \rangle$ jsou validní kódy Turingových strojů, stroj M ihned odmítne, jinak pokračuje. Řekněme, že stroj M_1 má abecedu Σ a stroj M_2 má abecedu Ψ. Stroj M postupně generuje všechna slova z množiny $(\Sigma \cap \Psi)^*$ v jejich lexikografickém pořadí $w_0, w_1, w_2, \ldots$. Stroj M na své druhé pásce postupně provádí simulace stroje M_1 na těchto slovech v jejich lexikografickém pořadí. Nejprve provede první krok simulace stroje M_1 nad řetězcem w_0, následně pozastaví výpočet a uloží si údaje o konfiguraci rozpracované simulace. Poté postupuje dle následujícího schématu. Provede vždy jeden následující krok všech rozpracovaných simulací, uloží si údaje o konfiguraci všech rozpracovaných simulací, provede první krok simulace stroje M_1 nad dalším slovem, jehož simulace dosud nebyla spuštěna, a odstartuje další iteraci celého tohoto procesu. Pokud některá simulace skončila, v další iteraci už není znovu spouštěna. Takto stroj M eventuálně obslouží každý krok každé simulace stroje M_1 nad každým slovem. Tento proces je pro názornost vyobrazen na obrázku 1. V následujícím textu se k němu budeme odkazovat jako k *prokládaným diagonálním simulacím po krocích*.

Pokaždé, když simulace některého takového slova skončí akceptováním, stroj M si toto slovo poznačí na konec posloupnosti slov $u_0, u_1, u_2, \ldots$ sestávající právě ze slov dosud přijatých strojem M_1. Stroj M dále spouští prokládané diagonální simulace po krocích stroje M_2 na slovech $u_0, u_1, u_2, \ldots$, tedy na slovech, která byla přijata strojem M_1. Stroj M rovněž po jednom kroce střídá prokládané diagonální simulace po krocích,

které provádí stroj M_1 a stroj M_2, tj. nejprve provede stroj M_1 krok simulace dle výše uvedeného předpisu, načež provede stroj M_2 krok simulace podle tohoto postupu. Jelikož posloupnost $u_0, u_1, u_2, \ldots$ je v každém okamžiku konečná, stroj M_2 může eventuálně čekat na přidělení nového slova, pokud veškerá slova dosud schválená strojem M_1 již vyčerpal (odmítl je). Pokud stroj M_2 některé ze slov akceptuje, akceptuje i stroj M, neboť bylo nalezeno slovo, které patří do průniku $L(M_1) \cap L(M_2)$. Pokud takové slovo existuje, bude dříve či později nalezeno a stroj M akceptuje, tedy jazyk L je rekurizvně vyčíslitelný. Pokud takové slovo neexistuje, stroj M bude cyklit.
Poznámka: V případě, že $\Sigma \cap \Psi = \varnothing$, pak provádíme simulace pouze na slovech z konečné množiny $\{\epsilon\}$. Pokud v takovém případě simulace některého stroje M_1, M_2 na tomto řetězci skončí odmítnutím, odmítne i stroj M.



Obrázek 1: Prokládané diagonální simulace po krocích

c)

Deterministická varianta:

Jazyk L je rozhodnutelný. Uvažujme o vícepáskovém Turingově stroji M', který pracuje následovně:

Na první pásce má M' uložený svůj vstup. Pokud tento vstup není ve tvaru $\langle M \rangle \# \langle w \rangle \# \langle n \rangle$, kde $\langle M \rangle$ je validní kód Turingova stroje, $\langle w \rangle$ je validní kód řetězce a $\langle n \rangle$ je validní kód přirozeného čísla, stroj M' ihned odmítne, jinak pokračuje. Na druhou pásku vloží vstup w a nad touto páskou provádí simulaci stroje M. Na třetí pásce zaznamenává počet provedených kroků simulace. Pokud M provede n kroků, aniž by akceptoval nebo odmítl w, stroj M' akceptuje. Pokud M akceptuje w do n kroků, stroj M' odmítne. Pokud M odmítne w do n kroků, stroj M' akceptuje. Stroj M' je tedy úplný Turingův stroj, neboť pro libovolný vstup zastaví. Jazyk L je tudíž rekurzivní.

Nedeterminismus nám v tomto případě nijak nepomůže.

d)

Nedeterministická varianta:

Jazyk L je rozhodnutelný. Uvažujme o vícepáskovém Turingově stroji M', který pracuje následovně:

Na první pásce má M' uložený svůj vstup. Pokud tento vstup není ve tvaru $\langle M \rangle \# \langle n \rangle$, kde $\langle M \rangle$ je validní kód Turingova stroje a $\langle n \rangle$ je validní kód přirozeného čísla, stroj M' ihned odmítne, jinak pokračuje. Řekněme, že Σ je abeceda stroje M. Stroj M' nedeterministicky vygeneruje slovo nad abecedou Σ a spustí simulaci stroje M na tomto slově, přičemž provede nejvýše n kroků. Pokud simulace skončí akceptováním do n kroků, akceptuje i M. Pokud simulace skončí odmítnutím do n kroků, nebo pokud simulace ani do n kroků neskončila, stroj M' odmítne. Pokud tedy existuje nějaké slovo ze Σ^*, které tuto podmínku splňuje, bude eventuálně nalezeno a stroj M' akceptuje. Pokud neexistuje, každá nedeterministická volba skončí odmítnutím. Stroj M' tedy zastaví pro každý svůj vstup. Jazyk L je tudíž rekurzivní.

Deterministická varianta:

Jazyk L je rozhodnutelný. Uvažujme o vícepáskovém Turingově stroji M', který pracuje následovně:

Na první pásce má M' uložený svůj vstup. Pokud tento vstup není ve tvaru $\langle M \rangle \# \langle n \rangle$, kde $\langle M \rangle$ je validní kód Turingova stroje a $\langle n \rangle$ je validní kód přirozeného čísla, stroj M' ihned odmítne, jinak pokračuje. Řekněme, že Σ je abeceda stroje M. Stroj M' postupně v lexikografickém pořadí generuje veškeré řetězce nad abecedou Σ, jejichž délka je n nebo nižší a nad každým takovým řetězcem provede nejvýše n kroků simulace stroje M. Pokud simulace skončí akceptováním do n kroků, akceptuje i M. Pokud simulace skončí odmítnutím do n kroků, nebo pokud simulace ani do n kroků neskončila, přesune se stroj M' k simulaci dalšího slova dle výše uvedeného pořadí. Pokud stroj M' vyčerpá všechna uvedená slova, aniž by některé z nich akceptoval do n kroků, odmítne svůj vstup. Stroj M' tedy zastaví pro každý svůj vstup, protože buď odhalí nějaké hledané slovo, nebo učiní konečný počet kroků simulace nad konečným počtem řetězců a odmítne. Pokud totiž žádné slovo délky n a menší neakceptuje do n kroků, je jasné, že všechna delší slova by pro případné akceptování potřebovala ještě více kroků, neboť žádná dvě různá slova w_1, w_2, kde $|w_1| > n$ a w_1 je předponou w_2, nejsou v rámci n kroků Turingova stroje rozlišitelná. Jazyk L je tudíž rekurzivní.

e)

Deterministická varianta:

Jazyk L je rozhodnutelný. Uvažujme o vícepáskovém Turingově stroji M', který pracuje následovně:

Na první pásce má M' uložený svůj vstup. Pokud tento vstup není ve tvaru $\langle M \rangle \# \langle n \rangle$, kde $\langle M \rangle$ je validní kód Turingova stroje a $\langle n \rangle$ je validní kód přirozeného čísla, stroj M' ihned odmítne, jinak pokračuje. Řekněme, že Σ je abeceda stroje M. Stroj M' postupně v lexikografickém pořadí generuje veškeré řetězce nad abecedou Σ, jejichž délka je $n+1$ nebo nižší a nad každým takovým řetězcem provede nejvýše n kroků simulace stroje M. Pokud některá z těchto simulací neskončí ani do n kroků, stroj M' odmítne. Pokud simulace skončí do n kroků, stroj přejde k simulaci dalšího slova v uvedeném pořadí. Pokud stroj M' vyčerpá všechna uvedená slova, aniž by na některém z nich vykonal více než n kroků, akceptuje svůj vstup. Pokud jsou totiž všechna slova o délce $n+1$ přijata nebo odmítnuta do n kroků, je jasné, že nemohla být ani celá dočtena, pročež i všechna delší slova budou přijata nebo odmítnuta do n kroků, neboť neexistuje způsob, jak tato delší slova v rámci n kroků rozlišit od slov o délce $n+1$ nebo nižší. Stroj M' tedy zastaví pro každý svůj vstup, protože buď odhalí slovo, které vyžaduje více než n kroků simulace, nebo provede konečný počet kroků simulace nad konečnou množinou slov. Jazyk L je tudíž rekurzivní.

Nedeterminismus nám v tomto případě nijak nepomůže.

f)

Deterministická varianta:

Jazyk L je rozhodnutelný. Uvažujme o vícepáskovém Turingově stroji M', který pracuje následovně:

Na první pásce má M' uložený svůj vstup. Pokud tento vstup není ve tvaru $\langle M \rangle \# \langle n \rangle$, kde $\langle M \rangle$ je validní kód Turingova stroje a $\langle n \rangle$ je validní kód přirozeného čísla, stroj M' ihned odmítne, jinak pokračuje. Řekněme, že Σ je abeceda stroje M. Stroj M' postupně v lexikografickém pořadí generuje veškeré řetězce nad abecedou Σ, jejichž délka je $n + 1$ nebo nižší a nad každým takovým řetězcem provede nejvýše n kroků simulace stroje M. Pokud některá z těchto simulací skončí do n kroků, stroj M' odmítne. Pokud simulace neskončí ani do n kroků, stroj přejde k simulaci dalšího slova v uvedeném pořadí. Pokud stroj M' vyčerpá všechna uvedená slova, aniž by na některém z nich zastavil v méně než n krocích, akceptuje svůj vstup. Pokud totiž pro žádné slovo délky $n+1$ ani kratší nestačí n kroků k ukončení simulace stroje M nad daným slovem, pak je jasné, že i na všech delších slovech bude provedeno více než n kroků simulace, neboť neexistuje způsob, jak tato delší slova v rámci n kroků rozlišit od slov o délce $n + 1$ nebo nižší. Stroj M' tedy zastaví pro každý svůj vstup, protože buď odhalí slovo, jehož simulace strojem M skončí dříve než v n krocích, nebo provede konečný počet kroků simulace nad konečnou množinou slov. Jazyk L je tudíž rekurzivní.

Nedeterminismus nám v tomto případě nijak nepomůže.

Sekce 18: Nerozhodnutelnost, redukce a diagonalizace

Úloha 1

Rozhodněte, zda jsou následující jazyky rozhodnutelné, nerozhodnutelné ale částečně rozhodnutelné nebo nejsou ani částečně rozhodnutelné. Nerozhodnutelnost, případně skutečnost, že jazyk není ani částečně rozhodnutelný, dokažte pomocí redukce. Rozhodnutelnost a částečnou rozhodnutelnost nemusíte dokazovat.

(c) $L = \{\langle M \rangle \# \langle n \rangle \mid M$ je TS t.ž. $\texttt{card}(L(M)) > n\}$
(d) $L = \{\langle M \rangle \# \langle n_1 \rangle \# \langle n_2 \rangle \mid M$ je TS t.ž. $n_1 \leq \texttt{card}(L(M)) \leq n_2\}$
(e) $L = \{\langle M \rangle \# \langle w_1 \rangle \# \langle w_2 \rangle \mid M$ je TS t.ž. $w_1, w_2 \in L(M) \wedge \texttt{steps}(M, w_1) > \texttt{steps}(M, w_2)\}$
(f) $L = \{\langle M_1 \rangle \# \langle M_2 \rangle \mid M_1, M_2$ jsou TS t.ž. $L(M_1) \cup L(M_2) \neq \{\}\}$
(g) ${}^*L = \{\langle M \rangle \mid M$ je TS t.ž. $L(M) = \Sigma^*\}$

c)

Jazyk L je nerozhodnutelný, ale je částečně rozhodnutelný. Nerozhodnutelnost dokážeme pomocí redukce z HP, tedy ukážeme, že $HP \leq L$. Požadovanou funkci $\sigma : \{0, 1, \#\}^* \to \{0, 1, \#\}^*$ definujeme takto:

$$\sigma(\langle M \rangle \# \langle w \rangle) = \langle M' \rangle \# \langle n \rangle$$

Pokud $\langle M \rangle \# \langle w \rangle$ není korektní instance HP, vrací σ kód takového TS M', že $L(M') = \{\}$. Předpokládáme totiž, že n je přirozené číslo, a tudíž určitě pro $card(L(M')) = 0$ není podmínka jazyka splněna. Jinak vrací kód TS M', který pracuje následovně:

- M' spustí simulaci M na řetězci w (ignoruje svůj vstup w'). Poznamenejme, že kódy $\langle M \rangle$ a $\langle w \rangle$ jsou zakódovány ve stavovém řízení stroje M'. Pokud simulace cyklí, cyklí i M' pro každý svůj vstup.
- Pokud simulace stroje M skončí, pak M' akceptuje svůj vstup w', pročež M' akceptuje každý svůj vstup w'.

Pro M' tedy platí:

$$\langle M \rangle \# \langle w \rangle \in HP \Rightarrow L(M') = \Sigma^* \Rightarrow \langle M' \rangle \# \langle n \rangle \in L$$

$$\langle M \rangle \# \langle w \rangle \notin HP \Rightarrow L(M') = \{\} \Rightarrow \langle M' \rangle \# \langle n \rangle \notin L$$

$$\langle M \rangle \# \langle w \rangle \in HP \iff \sigma(\langle M \rangle \# \langle w \rangle) \in L$$

Tato redukce je správná, neboť σ je totální, rekurzivně vyčíslitelná funkce zachovávající příslušnost do jazyka.

d)

Jazyk L není ani částečně rozhodnutelný. Dokážeme to pomocí redukce z $co - HP$, tedy ukážeme, že $co - HP \leq L$. Požadovanou funkci $\sigma : \{0, 1, \#\}^* \to \{0, 1, \#\}^*$ definujeme takto:

$$\sigma(\langle M \rangle \# \langle w \rangle) = \langle M' \rangle \# \langle 0 \rangle \# \langle 0 \rangle$$

Kód $\langle 0 \rangle$ odpovídá kódu přirozeného čísla 0. Pokud $\langle M \rangle \# \langle w \rangle$ není korektní instance $co - HP$, vrací σ kód takového TS M', že $L(M') = \Sigma^*$, tj. $\langle M' \rangle \# \langle 0 \rangle \# \langle 0 \rangle \notin L$, neboť mohutnost Σ^* přesahuje hodnotu 0. Jinak vrací kód TS M', který pracuje následovně:

- M' spustí simulaci M na w. Poznamenejme, že kódy $\langle M\rangle$ i $\langle w\rangle$ jsou zakódovány v M'.
- Pokud simulace M cyklí, cyklí i M' pro každý svůj vstup w'.
- Pokud simulace M zastaví, M' akceptuje, a tudíž akceptuje každý svůj vstup w'.

Pro M' tedy platí:

$$\langle M\rangle\#\langle w\rangle \in co-HP \Rightarrow L(M') = \{\} \Rightarrow \langle M'\rangle\#\langle 0\rangle\#\langle 0\rangle \in L$$

$$\langle M\rangle\#\langle w\rangle \notin co-HP \Rightarrow L(M') = \Sigma^* \Rightarrow \langle M'\rangle\#\langle 0\rangle\#\langle 0\rangle \notin L$$

$$\langle M\rangle\#\langle w\rangle \in co-HP \iff \sigma(\langle M\rangle\#\langle w\rangle) \in L$$

Tato redukce je správná, neboť σ je totální, rekurzivně vyčíslitelná funkce zachovávající příslušnost do jazyka.

e)

Jazyk L není rozhodnutelný, ale je částečně rozhodnutelný. Nerozhodnutelnost ukážeme pomocí redukce z HP, tedy ukážeme, že $HP \leq L$. Požadovanou funkci $\sigma : \{0,1,\#\}^* \to \{0,1,\#\}^*$ definujeme takto:

$$\sigma(\langle M\rangle\#\langle w\rangle) = \langle M'\rangle\#\langle w_1\rangle\#\langle w_2\rangle$$

Kódy $\langle w_1\rangle, \langle w_2\rangle$ odpovídají dvěma různým řetězcům $w_1, w_2 \in \Sigma^*$. Pokud $\langle M\rangle\#\langle w\rangle$ není korektní instance HP, vrací σ kód takového TS M', že $L(M') = \{\}$, tj. určitě $w_1 \notin L(M') \wedge w_2 \notin L(M')$. Jinak vrací kód vícepáskového TS M', který pracuje následovně:

- M' spustí simulaci M na řetězci w (ignoruje prozatím svůj vstup w'). Kódy $\langle M\rangle, \langle w\rangle$ jsou uloženy přímo v M'. Pokud simulace cyklí, cyklí i M'.
- Pokud simulace M skončí, pak M' vloží na druhou pásku sekvenci $\Delta w_1 \Delta\Delta^\omega$, hlavu umístí na počátek sekvence w_1 a hlavu na první pásce umístí na počátek slova w'. Do tohoto okamžiku nezáleželo na vstupu w', stroj M' tedy prozatím udělal stejný počet kroků pro každé w'. Tento počet označme c. Stroj pokračuje jednotlivými kroky doprava na obou páskách a porovnává přitom w' s w_1. Jakmile se na některé pozici symboly liší, M' akceptuje. Jakmile se M' na druhé pásce ocitne mimo slovo w_1, akceptuje, pokud w' pokračuje dalšími symboly. Pokud současně skončila i sekvence w', M' učiní na obou páskách současně jeden krok doleva a teprve poté akceptuje. Pokud tedy není w_1 předponou w', stroj M' na vstupu w' učiní méně než $c + |w_1|$ kroků, nebo právě tolik kroků. Pokud je w_1 vlastním prefixem w', pak stroj M' na vstupu w' učiní $c + |w_1| + 1$ kroků. Pokud $w' = w_1$, pak stroj M' na w' učiní $c + |w_1| + 2$ kroků. Jiné možnosti nemohou nastat, je tedy jasné, že pokud M na w zastaví, pak M' učiní na w_1 více kroků než na všech ostatních myslitelných vstupech.

Pro M' tedy platí:

$$\langle M\rangle\#\langle w\rangle \in HP \Rightarrow L(M') = \Sigma^* \wedge steps(M', w_1) > steps(M', w_2) \Rightarrow \langle M'\rangle\#\langle w_1\rangle\#\langle w_2\rangle \in L$$

$$\langle M\rangle\#\langle w\rangle \notin HP \Rightarrow L(M') = \{\} \Rightarrow \langle M'\rangle\#\langle w_1\rangle\#\langle w_2\rangle \notin L$$

$$\langle M\rangle\#\langle w\rangle \in HP \iff \sigma(\langle M\rangle\#\langle w\rangle) \in L$$

Tato redukce je správná, neboť σ je totální, rekurzivně vyčíslitelná funkce zachovávající příslušnost do jazyka.

f)

Jazyk L není rozhodnutelný, ale je částečně rozhodnutelný. Nerozhodnutelnost ukážeme pomocí redukce z HP, tedy ukážeme, že $HP \leq L$. Požadovanou funkci $\sigma : \{0,1,\#\}^* \to \{0,1,\#\}^*$ definujeme takto:

$$\sigma(\langle M\rangle\#\langle w\rangle) = \langle M_1\rangle\#\langle M_2\rangle$$

Pokud $\langle M\rangle\#\langle w\rangle$ není korektní instance HP, vrací σ kód takových dvou TS M_1, M_2, že $L(M_1) = L(M_2) = \{\}$. Jinak vrací kód TS M_1 tak, že $L(M_1) = \{\}$ a kód TS M_2, který pracuje následovně:

- M_2 spustí simulaci M na řetězci w (ignoruje svůj vstup w'). Poznamenejme, že kódy $\langle M\rangle$ a $\langle w\rangle$ jsou uloženy ve stavovém řízení stroje M_2. Pokud simulace cyklí, cyklí i M_2.
- Pokud simulace M_2 skončí, pak M_2 akceptuje každý svůj vstup w'.

Pro M_1, M_2 tedy platí:

$$\langle M\rangle\#\langle w\rangle \in HP \Rightarrow L(M_1) \cup L(M_2) = \Sigma^* \Rightarrow \langle M_1\rangle\#\langle M_2\rangle \in L$$

$$\langle M\rangle\#\langle w\rangle \notin HP \Rightarrow L(M_1) \cup L(M_2) = \{\} \Rightarrow \langle M_1\rangle\#\langle M_2\rangle \notin L$$

$$\langle M\rangle\#\langle w\rangle \in HP \iff \sigma(\langle M\rangle\#\langle w\rangle) \in L$$

Tato redukce je správná, neboť σ je totální, rekurzivně vyčíslitelná funkce zachovávající příslušnost do jazyka.

g*)

Jazyk L není ani částečně rozhodnutelný. Nerozhodnutelnost ukážeme pomocí redukce z $co - HP$, tedy ukážeme, že $co - HP \leq L$. Požadovanou funkci $\sigma : \{0,1,\#\}^* \to \{0,1\}^*$ definujeme takto:

$$\sigma(\langle M\rangle\#\langle w\rangle) = \langle M'\rangle$$

Pokud $\langle M\rangle\#\langle w\rangle$ není korektní instance $co - HP$, vrací σ kód takového stroje M', že $L(M') = \{\}$. Jinak vrací kód TS M', který pracuje následovně:

- Stroj M' si zapamatuje délku svého vstupu w', tedy hodnotu $|w'|$.
- Stroj M' provede nejvýše $|w'|$ kroků simulace stroje M na řetězci w. Poznamenejme, že kódy $\langle M\rangle$ a $\langle w\rangle$ jsou zakódovány ve stavovém řízení stroje M'.
- Pokud simulace M na w skončí v méně než $|w'|$ krocích, případně právě v tomto počtu kroků, stroj M' odmítne svůj vstup.
- Pokud simulace neskončí ani do $|w'|$ kroků, stroj M' akceptuje svůj vstup.

Je zřejmé, že pokud M přijme nebo odmítne w (tj. necyklí na tomto řetězci), musí tak učinit v konečném počtu kroků. Označme tento počet c. Potom ale stroj M' akceptuje pouze řetězce o délce c nebo nižší, což neodpovídá jazyku Σ^*. Nad všemi řetězci s vyšší délkou je totiž povoleno provést více než c kroků simulace M nad w, takže se nám po c krocích podaří detekovat, že simulace již skončila, a můžeme odmítnout vstup. Pokud však M na w cyklí, neexistuje žádný konečný počet kroků c, v němž by simulace M na w skončila, pročež M' přijme každý vstupní řetězec w', neboť nikdy nedojde k tomu, že bychom do $|w'|$ kroků odhalili konec simulace M nad w, tedy M' přijme celý jazyk Σ^*.

Pro M' tedy platí:

$$\langle M \rangle \# \langle w \rangle \in co - HP \Rightarrow L(M') = \Sigma^* \Rightarrow \langle M' \rangle \in L$$

$$\langle M \rangle \# \langle w \rangle \notin co - HP \Rightarrow L(M') \subset \Sigma^* \Rightarrow \langle M' \rangle \notin L$$

$$\langle M \rangle \# \langle w \rangle \in co - HP \iff \sigma(\langle M \rangle \# \langle w \rangle) \in L$$

Tato redukce je správná, neboť σ je totální, rekurzivně vyčíslitelná funkce zachovávající příslušnost.

Úloha 2

Rozhodněte a dokažte, zda platí následující tvrzení

(b) $\forall L_1, L_2 \in 2^{\Sigma^*} : L_1 \leq L_2$
(c) $\forall L_1, L_2 \in 2^{\Sigma^*} \setminus \{\{\}, \Sigma^*\} : L_1 \leq L_2$

b)

Jistě neplatí. V rámci protipříkladu můžeme zvolit $L_1 = HP, L_2 = \{\epsilon\}$. Pomocí diagonalizace lze snadno ukázat nerozhodnutelnost HP. Konstrukcí konečného automatu naopak lze ukázat rozhodnutelnost jazyka $\{\epsilon\}$.
Pro spor předpokládejme, že existuje funkce σ, která je redukcí jazyka HP na jazyk $\{\epsilon\}$. Sestrojme nyní Turingův stroj M, který pracuje nad abecedou $\{0, 1, \#\}$. Tento stroj nejprve na svém vstupu w' provede simulaci stroje M_σ, který realizuje výše popsanou redukční funkci σ. Jelikož je tato funkce totální a rekurzivně vyčíslitelná, tento proces skončí v konečném počtu kroků a jeho výstupem bude řetězec $\sigma(w')$. Stroj M následně rozhodne, zda $\sigma(w') \in \{\epsilon\}$. Toto lze jistě provést v konečném čase, neboť jazyk $\{\epsilon\}$ je rekurzivní. Pokud $\sigma(w') \in \{\epsilon\}$, pak stroj M akceptuje svůj vstup, jinak odmítne. Lze snadno nahlédnout, že stroj M je úplný, tedy zastaví pro každý svůj vstup. Jazyk $L(M)$ je tedy rekurzivní. Z definice redukční funkce σ rovněž plyne, že $L(M) = HP$, tedy Turingův stroj M rozhoduje problém zastavení. To je ovšem spor s faktem, že HP je nerozhodnutelný. Tento spor vzniká z předpokladu, že existuje redukční funkce σ, která je redukcí HP na $\{\epsilon\}$. Z toho plyne, že taková redukční funkce neexistuje, a tedy není pravda, že libovolné dva jazyky nad abecedou Σ jsou na sebe vzájemně redukovatelné.

c)

Lze použít stejný protipříklad a stejnou argumentaci jako v **b)**.

Úloha 3

Diagonalizací ukažte, že

(b) existuje úplný TS, který se liší od všech kontextových gramatik

b)

Důkaz. Pro každé $w \in \{0,1\}^*$ uvažujme o G_w jako o kontextové gramatice s kódem w. Pokud w není legitimní kód kontextové gramatiky, pak G_w přisoudíme kód gramatiky $G_{empty} = \{\{S\}, \{a\}, \{\}, S\}$.
Uvažujme nyní o nekonečné matici

$$\begin{array}{cccccc} \circ & \epsilon & 0 & 1 & 00 & \dots \\ G_\epsilon & a_{\epsilon,\epsilon} & a_{\epsilon,0} & a_{\epsilon,1} & a_{\epsilon,00} & \dots \\ G_0 & a_{0,\epsilon} & a_{0,0} & a_{0,1} & a_{0,00} & \dots \\ G_1 & a_{1,\epsilon} & a_{1,0} & a_{1,1} & a_{1,00} & \dots \\ G_{00} & a_{00,\epsilon} & a_{00,0} & a_{00,1} & a_{00,00} & \dots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots \end{array}$$

kde každý řádek odpovídá kontextové gramatice G_w v lexikografickém pořadí, jak bylo vysvětleno výše. Každý sloupec odpovídá řetězci nad abecedou $\{0,1\}$. Sloupce jsou seřazeny vzestupně dle lexikografického pořadí těchto řetězců. Každá buňka matice $a_{w,x}$ nabývá hodnoty 0 v případě, že $x \notin L(G_w)$, nebo hodnoty 1, pokud $x \in L(G_w)$.
Uvažujme nyní o takové gramatice G, kde $L(G) = \{w_i \mid i \in \mathbb{N} \wedge a_{i,i} = 0\}$. Všimněme si, že provádíme komplement diagonály výše uvedené matice. Z toho důvodu se G liší od všech kontextových gramatik, neboť jsme v matici uvedli všechny kontextové gramatiky a gramatika G nemůže být v matici obsažena.
Nyní ukažme, že $L(G)$ lze přijmout úplným Turingovým strojem. Uvažujme o třípáskovém Turingově stroji T, který pracuje následovně. Na začátku běhu má na první pásce svůj vstup $\Delta w \Delta\Delta^\omega$. Druhá a třetí páska mají tvar $\Delta\Delta^\omega$. Stroj T zkontroluje, zda w je legitimní kód kontextové gramatiky. Pokud ne, stroj T akceptuje (neboť jsme G_w přisoudili gramatiku G_{empty}, která nepřijímá žádný řetězec).
Na druhou pásku překopírujeme kód $\langle S \rangle$, kde S je startující neterminál gramatiky G_w. Druhá páska má nyní tvar $\Delta\langle S \rangle \Delta\Delta^\omega$. Na třetí pásce budeme uchovávat historii posloupností derivací ze startujícího netreminálu. Nyní budeme provádět postupné derivace větné formy na druhé pásce pomocí gramatiky G_w. Na třetí pásce vždy zaznamenáme novou větnou formu, kterou od předchozí oddělíme vhodným oddělovačem. Pro derivaci vždy volíme pravidlo, které může v konečném počtu derivačních kroků vést na větnou formu, kterou jsme dosud nevygenerovali. Toto lze kontrolovat pomocí historie posloupností derivací na třetí pásce.
Jakmile na druhé pásce obdržíme kód odpovídající posloupnosti terminálů, zkontrolujeme, zda se tento kód shoduje s řetězcem w. Pokud ano, odmítneme, jinak pokračujeme překopírováním řetězce $\Delta\langle S \rangle \Delta\Delta^\omega$ na druhou pásku a odstartujeme znovu celý proces tak, že pomocí historie předchozích běhů uložené na třetí pásce zajistíme, abychom použili dosud nevygenerovanou posloupnost derivací.
Pokud jsme na druhé pásce obdrželi kód odpovídající posloupnosti terminálů a neterminálů, jehož délka je vyšší než w, překopírujeme na druhou pásku $\Delta\langle S \rangle \Delta\Delta^\omega$ a pokračujeme znovu jinou posloupností derivací (to opět zajistíme kontrolou historie derivací na třetí pásce). Toto provádíme proto, že kontextová gramatika nemůže obsahovat zkracující pravidla, nebylo by tedy možné vygenerovat řetězec w.
Pokud historie posloupností derivací obsahuje veškeré derivační stromy o výšce w a kratší, akceptujeme.
Tento TS tedy přijme jazyk $L(G) = \{w_i \mid i \in \mathbb{N} \wedge a_{i,i} = 0\}$, o němž jsme ukázali, že nemůže být přijat žádnou kontextovou gramatikou. Současně lze nahlédnout, že tento TS je úplný, neboť buď v konečném počtu kroků odmítne (pokud G_w přijímá w), nebo v konečném počtu kroků akceptuje (pokud historie posloupností derivací obsahuje všechny derivační stromy o délce w a kratší, kterých je konečný počet).
Ukázali jsme tedy, že existuje úplný Turingův stroj, který se liší od všech kontextových gramatik. □

Sekce 19: Asymptotická složitost

Úloha 2

Dokažte, jaký platí vztah mezi následujícími třídami složitosti

(b) $\mathcal{O}(n \cdot \log n)$ a $\mathcal{O}(n^2)$
(c) $\mathcal{O}(n \cdot \sin n)$ a $\mathcal{O}(n^2)$
(d) $\mathcal{O}(2^n)$ a $\mathcal{O}(2^{n^2})$
(e) ${}^*\mathcal{O}(2^n)$ a $\mathcal{O}(n!)$
(f) $\mathcal{O}(n^2)$ a $\mathcal{O}(2^{\ln n})$
(g) $\mathcal{O}(2^{2^n})$ a $\mathcal{O}(2^{n^2})$
(h) $\mathcal{O}(\ln n)$ a $\mathcal{O}(n \cdot \ln \ln n)$
(i) $\mathcal{O}(\ln n)$ a $\mathcal{O}(\log n)$

Asymptotické horní omezení:

Zopakujme definici $\mathcal{O}(.)$. Ať $\mathcal{F}$ je množina všech funkcí $f : \mathbb{N} \to \mathbb{N}$ a $f \in \mathcal{F}$. Pak

$$\mathcal{O}(f(n)) = \{g(n) \in \mathcal{F} \mid \exists n_0 \in \mathbb{N}\ \exists c \in \mathbb{R}^+\ \forall n \in \mathbb{N} : n \geq n_0 \Rightarrow 0 \leq g(n) \leq c \cdot f(n)\}$$

Stolzova-Cesàrova věta:

Zopakujme rovněž definici Stolzovy-Cesàrovy věty pro výpočet limity podílu dvou posloupností. Ať $(a_n), (b_n)$ jsou dvě reálné posloupnosti (tj. zobrazení $\mathbb{N} \to \mathbb{R}$). Dále ať platí

$$\lim_{n\to\infty} b_n = \infty$$

a zároveň ať je (b_n) rostoucí a nenulová (čemuž budeme říkat první podmínka Stolzovy-Cesàrovy věty) a dále ať existuje $L \in \mathbb{R} \cup \{\infty, -\infty\}$, kde

$$L = \lim_{n\to\infty} \frac{a_{n+1} - a_n}{b_{n+1} - b_n}$$

(čemuž budeme říkat druhá podmínka Stolzovy-Cesàrovy věty). Jsou-li tyto dvě podmínky splněny, pak

$$\lim_{n\to\infty} \frac{a_n}{b_n} = L.$$

L'Hospitalovo pravidlo pro limitu posloupností:

Zopakujme dále, jak lze použít L'Hospitalovo pravidlo pro nalezení limity podílu dvou posloupností. Jelikož posloupnosti jsou diskrétní funkce, které nemají v žádném bodě derivaci, nelze limitu podílu posloupností počítat pomocí limity derivací posloupností, ale pomocí limity derivací funkcí, které tyto posloupnosti vhodně aproximují. Mějme dvě reálná zobrazení $f : \mathbb{R} \to \mathbb{R}$ a $g : \mathbb{R} \to \mathbb{R}$. Mějme dále dvě reálné posloupnosti $(a_n), (b_n)$ (tj. zobrazení $\mathbb{N} \to \mathbb{R}$). Ať $\forall n \in \mathbb{N} : f(n) = a_n \wedge g(n) = b_n$. Pokud existuje takové $L \in \mathbb{R} \cup \{\infty, -\infty\}$, že

$$L = \lim_{x\to\infty} \frac{f(x)}{g(x)},$$

pak rovněž

$$L = \lim_{n\to\infty} \frac{a_n}{b_n}.$$

Z toho vyvozujeme, že pokud jsou splněny všechny výše uvedené podmínky a současně

$$\lim_{x\to\infty} f(x) = \infty \wedge \lim_{x\to\infty} g(x) = \infty,$$

pak můžeme limitu podílu dvou posloupností spočíst jako

$$\lim_{n\to\infty} \frac{a_n}{b_n} = \lim_{x\to\infty} \frac{f'(x)}{g'(x)}$$

Za splnění určitých podmínek tedy lze následující příklady řešit pomocí Stolzovy-Cesàrovy věty nebo pomocí L'Hospitalova pravidla. Poznamenejme, v některých případech nelze tyto postupy smysluplně použít.

b)

Důkaz. Jistě $\mathcal{O}(n \cdot \log n) \subset \mathcal{O}(n^2)$. Dokažme to ve dvou krocích, tedy postupně ukažme $\mathcal{O}(n \cdot \log n) \subseteq \mathcal{O}(n^2)$ a $\mathcal{O}(n \cdot \log n) \neq \mathcal{O}(n^2)$.

$\mathcal{O}(n \cdot \log n) \subseteq \mathcal{O}(n^2)$:

Důkaz. Uvažujme o libovolné funkci $f_0 \in \mathcal{O}(n \cdot \log n)$ a vyjděme z definice $\mathcal{O}$. Z ní plyne, že

$$\exists n_0 \in \mathbb{N}\, \exists c \in \mathbb{R}^+\, \forall n \geq n_0 : f_0(n) \leq c \cdot n \cdot \log n.$$

Stačí tedy ukázat, že existuje taková kladná konstanta c, aby rovněž

$$\lim_{n\to\infty} \frac{n^2}{c \cdot n \cdot \log n} = \lim_{n\to\infty} \frac{n}{c \cdot \log n} > 1.$$

Ověřme, zda můžeme využít L'Hospitalovo pravidlo pro výpočet limity posloupnosti. Zavěďme reálné funkce $f(x) = x$ a $g(x) = c \cdot \log(x)$. Zřejmě $\forall n \in \mathbb{N} : f(n) = n \wedge g(n) = c \cdot \log(n)$. Dále je zřejmé, že

$$\lim_{x\to\infty} f(x) = \infty \wedge \lim_{x\to\infty} g(x) = \infty,$$

tedy

$$\lim_{n\to\infty} \frac{n}{c \cdot \log n} = \lim_{x\to\infty} \frac{f'(x)}{g'(x)} = \frac{1}{c} \cdot \lim_{x\to\infty} \frac{1}{\frac{1}{x \cdot \ln(10)}} = \frac{\ln(10)}{c} \cdot \lim_{x\to\infty} x = \infty.$$

pro kladná c. Rozdíl mezi n^2 a $c \cdot n \cdot \log n$ je tedy pro $n \to \infty$ kladný a stále se zvětšuje, takže

$$\exists c \in \mathbb{R}^+\, \exists n_0 \in \mathbb{N}\, \forall n \geq n_0 : f_0(n) \leq c \cdot n \cdot \log n \leq n^2,$$

tedy $f_0(n) \in \mathcal{O}(n^2)$. □

$\mathcal{O}(n \cdot \log n) \neq \mathcal{O}(n^2)$:

Důkaz. Zde postačí ukázat, že existuje funkce $f_0 \in \mathcal{O}(n^2)$, pro kterou platí, že $f_0 \notin \mathcal{O}(n \cdot \log(n))$. Zvolme $f_0(n) = n^2$. V rámci důkazu sporem předpokládejme, že $f_0 \in \mathcal{O}(n \cdot \log n)$. Pak určitě platí, že

$$\exists n_0 \in \mathbb{N}\, \exists c \in \mathbb{R}^+\, \forall n \geq n_0 : n^2 \leq c \cdot n \cdot \log n.$$

Vyšetřeme nyní limitní chování podílu těchto dvou posloupností v nekonečnu, tedy zjistěme, čemu se rovná limita

$$\lim_{n\to\infty} \frac{c \cdot n \cdot \log(n)}{n^2} = \lim_{n\to\infty} \frac{c \cdot \log(n)}{n}.$$

Zvolme dvě reálné funkce $f(x) = c \cdot \log(x), g(x) = x$. Vše potřebné jsme již ověřili v předchozím kroku, proto můžeme pokračovat:

$$\lim_{n\to\infty} \frac{c \cdot \log n}{n} = \lim_{x\to\infty} \frac{f'(x)}{g'(x)} = c \cdot \lim_{x\to\infty} \frac{1}{x \cdot \ln(10)} = \frac{c}{\ln(10)} \cdot \lim_{x\to\infty} \frac{1}{x} = 0.$$

kde c je kladné. Z předpokladu víme, že $\exists c \in \mathbb{R}^+\, \exists n_0 \in \mathbb{N}\, \forall n \geq n_0 : n^2 \leq n \cdot c \cdot \log n$, neboli

$$1 \leq \frac{c \cdot n \cdot \log n}{n^2}.$$

Právě jsme ale ukázali, že

$$\lim_{n\to\infty} \frac{c \cdot n \cdot \log n}{n^2} = 0,$$

což je spor. Proto $n^2 \notin \mathcal{O}(n \cdot \log n)$.

□

Jelikož jsme ukázali, že $\mathcal{O}(n \cdot \log n) \neq \mathcal{O}(n^2)$ a současně $\mathcal{O}(n \cdot \log n) \subseteq \mathcal{O}(n^2)$, pak určitě i $\mathcal{O}(n \cdot \log n) \subset \mathcal{O}(n^2)$.

□

c)

Důkaz. Jistě $\mathcal{O}(n \cdot \sin n) \subset \mathcal{O}(n^2)$. Dokažme to ve dvou krocích, tedy postupně ukažme $\mathcal{O}(n \cdot \sin n) \subseteq \mathcal{O}(n^2)$ a $\mathcal{O}(n \cdot \sin n) \neq \mathcal{O}(n^2)$.

$\mathcal{O}(n \cdot \sin n) \subseteq \mathcal{O}(n^2)$:

Důkaz. Uvažujme o libovolné funkci $f_0 \in \mathcal{O}(n \cdot \sin n)$ a vyjděme z definice $\mathcal{O}$. Z ní plyne, že

$$\exists n_0 \in \mathbb{N}\, \exists c \in \mathbb{R}^+\, \forall n \in \mathbb{N} : n \geq n_0 \Rightarrow f_0(n) \leq c \cdot n \cdot \sin n.$$

Pokud se nám povede ukázat, že

$$\exists n_0 \in \mathbb{N}\, \exists c \in \mathbb{R}^+\, \exists c' \in \mathbb{R}^+\, \forall n \in \mathbb{N} : n \geq n_0 \Rightarrow f_0(n) \leq c \cdot n \cdot \sin n \leq c' \cdot n^2,$$

pak bude zřejmé, že $\mathcal{O}(n \cdot \sin n) \subseteq \mathcal{O}(n^2)$. Zvolme $c' = c$ a upravujme:

$$c \cdot n \cdot \sin n \leq c \cdot n^2.$$

Omezme se na $n \in \mathbb{N} \setminus \{0\}$, pak můžeme upravovat:

$$\sin n \leq n,$$

což zřejmě platí pro každé $n \geq 1$. Ukázali jsme, že $f_0 \in \mathcal{O}(n^2)$, tedy $\mathcal{O}(n \cdot \sin n) \subseteq \mathcal{O}(n^2)$.

□

$\mathcal{O}(n \cdot \sin n) \neq \mathcal{O}(n^2)$:

Důkaz. Zde postačí ukázat, že existuje funkce $f_0 \in \mathcal{O}(n^2)$, pro kterou platí, že $f_0 \notin \mathcal{O}(n \cdot \sin n)$. Zvolme $f_0(n) = n^2$. V rámci důkazu sporem předpokládejme, že $f_0 \in \mathcal{O}(n \cdot \sin n)$. Pak určitě platí, že

$$\exists n_0 \in \mathbb{N}\, \exists c \in \mathbb{R}^+\, \forall n \in \mathbb{N} : n \geq n_0 \Rightarrow n^2 \leq c \cdot n \cdot \sin n.$$

Vyšetřeme nyní limitní chování podílu těchto dvou posloupností v nekonečnu, tedy zjistěme, čemu se rovná limita

$$\lim_{n\to\infty} \frac{c \cdot n \cdot \sin n}{n^2} = c \cdot \lim_{n\to\infty} \frac{\sin n}{n}$$

Lze snadno nahlédnout, že tuto limitu nelze řešit pomocí L'Hospitalova pravidla, ani pomocí Stolz-Cesàrovy věty, pročež toto ukážeme pomocí věty o třech limitách (známé také jako Squeeze theorem), která pro posloupnosti zní následovně:

Ať $(a_n), (b_n), (c_n)$ jsou tři posloupnosti, pro které platí

$$\exists n_0 \in \mathbb{N}\, \forall n \in \mathbb{N} : n \geq n_0 \Rightarrow a_n \leq b_n \leq c_n$$

a ať dále existuje $L \in \mathbb{R}$ takové, že platí

$$\lim_{n\to\infty} a_n = L \wedge \lim_{n\to\infty} c_n = L.$$

Pokud jsou tyto předpoklady splněny, pak i

$$\lim_{n\to\infty} b_n = L.$$

Zvolme tedy $(a_n) = \frac{-1}{n}, (c_n) = \frac{1}{n}$. Po kladná n zřejmě platí

$$\frac{-1}{n} \leq \frac{\sin n}{n} \leq \frac{1}{n},$$

neboť za předpokladu $n > 0$ můžeme vynásobit celou nerovnici n a dostaneme

$$-1 \leq \sin n \leq 1,$$

což z definice $\sin n$ zřejmě platí. Vyšetřeme nyní limitní chování (a_n) a (c_n) v nekonečnu.

$$\lim_{n\to\infty} \frac{-1}{n} = 0,$$

$$\lim_{n\to\infty} \frac{1}{n} = 0.$$

Jelikož se tyto limity rovnají, můžeme dle věty o třech limitách prohlásit, že i

$$c \cdot \lim_{n\to\infty} \frac{\sin n}{n} = 0,$$

kde c je kladné. Z předpokladu víme, že $\exists c \in \mathbb{R}^+\, \exists n_0 \in \mathbb{N}\, \forall n \geq n_0 : n^2 \leq c \cdot n \cdot \sin n$, neboli

$$1 \leq \frac{c \cdot n \cdot \sin n}{n^2}.$$

Právě jsme ale ukázali, že

$$\lim_{n\to\infty} \frac{c \cdot n \cdot \sin n}{n^2} = 0,$$

což je spor. Proto $n^2 \notin \mathcal{O}(n \cdot \sin n)$.

□

Ukázali jsme, že $\mathcal{O}(n \cdot \sin n) \neq \mathcal{O}(n^2)$ a rovněž že $\mathcal{O}(n \cdot \sin n) \subseteq \mathcal{O}(n^2)$. Z toho plyne, že $\mathcal{O}(n \cdot \sin n) \subset \mathcal{O}(n^2)$.

□

d)

Důkaz. Jistě $\mathcal{O}(2^n) \subset \mathcal{O}(2^{n^2})$. Dokažme to ve dvou krocích, tedy postupně ukažme $\mathcal{O}(2^n) \subseteq \mathcal{O}(2^{n^2})$ a $\mathcal{O}(2^n) \neq \mathcal{O}(2^{n^2})$.

$\mathcal{O}(2^n) \subseteq \mathcal{O}(2^{n^2})$:

Důkaz. Uvažujme o libovolné funkci $f_0 \in \mathcal{O}(2^n)$ a vyjděme z definice $\mathcal{O}$. Z ní plyne, že

$$\exists n_0 \in \mathbb{N}\ \exists c \in \mathbb{R}^+\ \forall n \geq n_0 : f_0(n) \leq c \cdot 2^n.$$

Stačí tedy ukázat, že existuje taková kladná konstanta c, aby rovněž

$$\lim_{n\to\infty} \frac{2^{n^2}}{c \cdot 2^n} > 1.$$

Upravujeme:

$$\lim_{n\to\infty} \frac{2^{n^2}}{c \cdot 2^n} = \frac{1}{c} \cdot \lim_{n\to\infty} 2^{(n^2-n)} = \frac{1}{c} \cdot \lim_{n\to\infty} 2^{n\cdot(n-1)}$$

Zřejmě

$$\lim_{n\to\infty} n \cdot (n-1) = \infty,$$

tedy i

$$\frac{1}{c} \cdot \lim_{n\to\infty} 2^{n\cdot(n-1)} = \infty > 1.$$

pro kladná c. Rozdíl mezi 2^{n^2} a $c \cdot 2^n$ je tedy pro $n \to \infty$ kladný a stále se zvětšuje, takže

$$\exists c \in \mathbb{R}^+\ \exists n_0 \in \mathbb{N}\ \forall n \geq n_0 : f_0(n) \leq c \cdot 2^n \leq 2^{n^2},$$

tedy $f_0(n) \in \mathcal{O}(2^{n^2})$.

□

$\mathcal{O}(2^n) \neq \mathcal{O}(2^{n^2})$:

Důkaz. Zde postačí ukázat, že existuje funkce $f_0 \in \mathcal{O}(2^{n^2})$, pro kterou platí, že $f_0 \notin \mathcal{O}(2^n)$. Zvolme $f_0(n) = 2^{n^2}$. V rámci důkazu sporem předpokládejme, že $f_0 \in \mathcal{O}(2^n)$. Pak určitě platí, že

$$\exists n_0 \in \mathbb{N}\, \exists c \in \mathbb{R}^+\, \forall n \geq n_0 : 2^{n^2} \leq c \cdot 2^n.$$

Vyšetřeme nyní limitní chování podílu těchto dvou posloupností v nekonečnu, tedy zjistěme, čemu se rovná limita

$$\lim_{n\to\infty} \frac{c \cdot 2^n}{2^{n^2}}.$$

Upravujeme:

$$c \cdot \lim_{n\to\infty} \frac{2^n}{2^{n^2}} = c \cdot \lim_{n\to\infty} 2^{(n-n^2)} = c \cdot \lim_{n\to\infty} 2^{n\cdot(1-n)}.$$

Zřejmě

$$\lim_{n\to\infty} n \cdot (1-n) = -\infty$$

a

$$\lim_{n\to\infty} 2^{-n} = 0,$$

tedy i

$$c \cdot \lim_{n\to\infty} 2^{n\cdot(1-n)} = 0,$$

kde c je kladné. Z předpokladu víme, že $\exists c \in \mathbb{R}^+\, \exists n_0 \in \mathbb{N}\, \forall n \geq n_0 : 2^{n^2} \leq c \cdot 2^n$, neboli

$$1 \leq \frac{c \cdot 2^n}{2^{n^2}}.$$

Právě jsme ale ukázali, že

$$\lim_{n\to\infty} \frac{c \cdot 2^n}{2^{n^2}} = 0,$$

což je spor. Proto $2^{n^2} \notin \mathcal{O}(2^n)$.

□

Ukázali jsme, že platí $\mathcal{O}(2^n) \neq \mathcal{O}(2^{n^2})$ a současně $\mathcal{O}(2^n) \subseteq \mathcal{O}(2^{n^2})$, pročež i $\mathcal{O}(2^n) \subset \mathcal{O}(2^{n^2})$.

□

e*)

Důkaz. Jistě $\mathcal{O}(2^n) \subset \mathcal{O}(n!)$. Dokažme to ve dvou krocích, tedy postupně ukažme $\mathcal{O}(2^n) \subseteq \mathcal{O}(n!)$ a $\mathcal{O}(2^n) \neq \mathcal{O}(n!)$.

$\mathcal{O}(2^n) \subseteq \mathcal{O}(n!)$:

Důkaz. Uvažujme o libovolné funkci $f_0 \in \mathcal{O}(2^n)$ a vyjděme z definice $\mathcal{O}$. Z ní plyne, že

$$\exists n_0 \in \mathbb{N}\ \exists c \in \mathbb{R}^+\ \forall n \geq n_0 : f_0(n) \leq c \cdot 2^n.$$

Stačí tedy ukázat, že existuje taková kladná konstanta c, aby rovněž

$$\lim_{n\to\infty} \frac{n!}{c \cdot 2^n} > 1.$$

Pro důkaz využijeme Stirlingovu aproximaci funkce Γ pro vysoké hodnoty n. Platí:

$$n! = \Gamma(n+1) \approx \sqrt{2 \cdot \pi \cdot n} \cdot \left(\frac{n}{e}\right)^n$$

Jelikož tato aproximace je stále přesnější pro vyšší a vyšší hodnoty n, lze psát

$$\lim_{n\to\infty} \frac{n!}{\sqrt{2 \cdot \pi \cdot n} \cdot \left(\frac{n}{e}\right)^n} = 1,$$

pročež

$$\lim_{n\to\infty} \frac{n!}{c \cdot 2^n} = \lim_{n\to\infty} \frac{\sqrt{2 \cdot \pi \cdot n} \cdot \left(\frac{n}{e}\right)^n}{c \cdot 2^n}$$

Upravujeme

$$\lim_{n\to\infty} \frac{\sqrt{2 \cdot \pi \cdot n} \cdot \left(\frac{n}{e}\right)^n}{c \cdot 2^n} = \frac{1}{c} \cdot \lim_{n\to\infty} \sqrt{2 \cdot \pi \cdot n} \cdot \left(\frac{n}{2 \cdot e}\right)^n = \frac{\sqrt{2 \cdot \pi}}{c} \cdot \lim_{n\to\infty} \sqrt{n} \cdot \lim_{n\to\infty} \left(\frac{n}{2 \cdot e}\right)^n = \infty > 1,$$

pro kladná c. Rozdíl mezi $n!$ a $c \cdot 2^n$ je tedy pro $n \to \infty$ kladný a stále se zvětšuje, takže

$$\exists c \in \mathbb{R}^+\ \exists n_0 \in \mathbb{N}\ \forall n \geq n_0 : f_0(n) \leq c \cdot 2^n \leq n!,$$

tedy $f_0(n) \in \mathcal{O}(n!)$.

□

$\mathcal{O}(2^n) \neq \mathcal{O}(n!)$:

Důkaz. Zde postačí ukázat, že existuje funkce $f_0 \in \mathcal{O}(n!)$, pro kterou platí, že $f_0 \notin \mathcal{O}(2^n)$. Zvolme $f_0(n) = n!$. V rámci důkazu sporem předpokládejme, že $f_0 \in \mathcal{O}(2^n)$. Pak určitě platí, že

$$\exists n_0 \in \mathbb{N}\ \exists c \in \mathbb{R}^+\ \forall n \geq n_0 : n! \leq c \cdot 2^n.$$

Vyšetřeme nyní limitní chování podílu těchto dvou posloupností v nekonečnu, tedy zjistěme, čemu se rovná limita

$$\lim_{n\to\infty} \frac{c \cdot 2^n}{n!}.$$

Využijeme opět Stirlingovu aproximaci, tedy upravujeme:

$$\lim_{n\to\infty} \frac{c \cdot 2^n}{n!} = \lim_{n\to\infty} \frac{c \cdot 2^n}{\sqrt{2 \cdot \pi \cdot n} \cdot \left(\frac{n}{e}\right)^n} = \lim_{n\to\infty} \frac{c}{\sqrt{2 \cdot \pi \cdot n}} \cdot \left(\frac{2 \cdot e}{n}\right)^n = \frac{c}{\sqrt{2 \cdot \pi}} \cdot \lim_{n\to\infty} \frac{1}{\sqrt{n}} \cdot \lim_{n\to\infty} \left(\frac{2 \cdot e}{n}\right)^n = 0,$$

pro kladné c.Z předpokladu víme, že $\exists c \in \mathbb{R}^+\ \exists n_0 \in \mathbb{N}\ \forall n \geq n_0 : n! \leq c \cdot 2^n$, neboli

$$1 \leq \frac{c \cdot 2^n}{n!}.$$

Právě jsme ale ukázali, že

$$\lim_{n\to\infty} \frac{c \cdot 2^n}{n!} = 0,$$

což je spor. Proto $n! \notin \mathcal{O}(2^n)$.

□

Ukázali jsme tedy, že platí $\mathcal{O}(2^n) \subseteq \mathcal{O}(n!)$ i $\mathcal{O}(2^n) \neq \mathcal{O}(n!)$, pročež i $\mathcal{O}(2^n) \subset \mathcal{O}(n!)$.

□

f)

Důkaz. Jistě $\mathcal{O}(2^{\ln(n)}) \subset \mathcal{O}(n^2)$. Přímočarý důkaz tohoto tvrzení pomocí limit by byl náročný, pročež si raději uvědomme, že ostrá množinová inkluze je relací ostrého uspořádání (tj. ireflexivní částečné uspořádání), tedy pokud se nám povede ukázat, že $\mathcal{O}(2^{\ln(n)}) \subset \mathcal{O}(n) \subset \mathcal{O}(n^2)$, pak z tranzitivity ostrého uspořádání současně plyne $\mathcal{O}(2^{\ln(n)}) \subset \mathcal{O}(n^2)$. (Rovněž si všimněme, že stačí ukázat pouze $\mathcal{O}(2^{\ln(n)}) \subseteq \mathcal{O}(n) \subset \mathcal{O}(n^2)$, tj. u jedné z relací nemusíme ostrost ukazovat). Začněme důkazem $\mathcal{O}(2^{\ln(n)}) \subseteq \mathcal{O}(n)$. To ukážeme v jednom kroku.

$\mathcal{O}(2^{\ln(n)}) \subseteq \mathcal{O}(n)$:

Důkaz. Uvažujme o libovolné funkci $f_0 \in \mathcal{O}(2^{\ln(n)})$ a vyjděme z definice $\mathcal{O}$. Z ní plyne, že

$$\exists n_0 \in \mathbb{N}\ \exists c \in \mathbb{R}^+\ \forall n \geq n_0 : f_0(n) \leq c \cdot 2^{\ln(n)}.$$

Stačí tedy ukázat, že existuje taková kladná konstanta c, aby rovněž

$$\lim_{n\to\infty} \frac{n}{c \cdot 2^{\ln(n)}} > 1.$$

Je vhodné si uvědomit, že pro kladná n platí

$$n = e^{\ln(n)},$$

díky čemuž můžeme provést úpravu

$$\lim_{n\to\infty} \frac{n}{c \cdot 2^{\ln(n)}} = \frac{1}{c} \cdot \lim_{n\to\infty} \frac{e^{\ln(n)}}{2^{\ln(n)}} = \frac{1}{c} \cdot \lim_{n\to\infty} \left(\frac{e}{2}\right)^{\ln(n)}.$$

Jelikož $e > 2$, pak

$$\frac{1}{c} \cdot \lim_{n\to\infty} \left(\frac{e}{2}\right)^{\ln(n)} = \infty > 1$$

pro kladná c. Rozdíl mezi n a $c \cdot 2^{\ln n}$ je tedy pro $n \to \infty$ kladný a stále se zvětšuje, takže

$$\exists c \in \mathbb{R}^+\ \exists n_0 \in \mathbb{N}\ \forall n \geq n_0 : f_0(n) \leq c \cdot 2^{\ln n} \leq n,$$

tedy $f_0(n) \in \mathcal{O}(n)$.

□

Nyní ve dvou krocích ukažme, že $\mathcal{O}(n) \subset \mathcal{O}(n^2)$., tedy postupně ukažme $\mathcal{O}(n) \subseteq \mathcal{O}(n^2)$ a $\mathcal{O}(n) \neq \mathcal{O}(n^2)$

$\mathcal{O}(n) \subseteq \mathcal{O}(n^2)$:

Důkaz. Uvažujme o libovolné funkci $f_0 \in \mathcal{O}(n)$ a vyjděme z definice $\mathcal{O}$. Z ní plyne, že

$$\exists n_0 \in \mathbb{N}\, \exists c \in \mathbb{R}^+\, \forall n \geq n_0 : f_0(n) \leq c \cdot n.$$

Stačí tedy ukázat, že existuje taková kladná konstanta c, aby rovněž

$$\lim_{n\to\infty} \frac{n^2}{c \cdot n} > 1.$$

Upravujeme:

$$\frac{1}{c} \cdot \lim_{n\to\infty} \frac{n^2}{n} = \frac{1}{c} \cdot \lim_{n\to\infty} n = \infty > 1$$

pro kladná c. Rozdíl mezi n^2 a $c \cdot n$ je tedy pro $n \to \infty$ kladný a stále se zvětšuje, takže

$$\exists c \in \mathbb{R}^+\, \exists n_0 \in \mathbb{N}\, \forall n \geq n_0 : f_0(n) \leq c \cdot n \leq n^2,$$

tedy $f_0(n) \in \mathcal{O}(n^2)$.

□

$\mathcal{O}(n) \neq \mathcal{O}(n^2)$:

Důkaz. Zde postačí ukázat, že existuje funkce $f_0 \in \mathcal{O}(n^2)$, pro kterou platí, že $f_0 \notin \mathcal{O}(n)$. Zvolme $f_0(n) = n^2$. V rámci důkazu sporem předpokládejme, že $f_0 \in \mathcal{O}(n)$. Pak určitě platí, že

$$\exists n_0 \in \mathbb{N}\, \exists c \in \mathbb{R}^+\, \forall n \geq n_0 : n^2 \leq c \cdot n.$$

Vyšetřeme nyní limitní chování podílu těchto dvou posloupností v nekonečnu, tedy zjistěme, čemu se rovná limita

$$\lim_{n\to\infty} \frac{c \cdot n}{n^2} = 0.$$

Upravujeme:

$$\lim_{n\to\infty} \frac{c \cdot n}{n^2} = c \cdot \lim_{n\to\infty} \frac{1}{n} = 0$$

pro kladná c. Z předpokladu víme, že $\exists c \in \mathbb{R}^+\, \exists n_0 \in \mathbb{N}\, \forall n \geq n_0 : n^2 \leq c \cdot n$, neboli

$$1 \leq \frac{c \cdot n}{n^2}.$$

Právě jsme ale ukázali, že

$$\lim_{n\to\infty} \frac{c \cdot n}{n^2} = 0,$$

což je spor. Proto $n^2 \notin \mathcal{O}(n)$.

□

Podařilo se nám tedy ukázat, že $\mathcal{O}(n) \neq \mathcal{O}(n^2)$ a současně $\mathcal{O}(n) \subseteq \mathcal{O}(n^2)$, pročež i $\mathcal{O}(n) \subset \mathcal{O}(n^2)$. Dále jsme ukázali, že $\mathcal{O}(2^{\ln(n)}) \subseteq \mathcal{O}(n)$. Z tranzitivity uspořádání v důsledku plyne, že $\mathcal{O}(2^{\ln(n)}) \subset \mathcal{O}(n^2)$.

□

g)

Důkaz. Jistě $\mathcal{O}(2^{n^2}) \subset \mathcal{O}(2^{2^n})$. Dokažme to ve dvou krocích, tedy postupně ukažme $\mathcal{O}(2^{n^2}) \subseteq \mathcal{O}(2^{2^n})$ a $\mathcal{O}(2^{n^2}) \neq \mathcal{O}(2^{2^n})$.

$\mathcal{O}(2^{n^2}) \subseteq \mathcal{O}(2^{2^n})$:

Důkaz. Uvažujme o libovolné funkci $f_0 \in \mathcal{O}(2^{n^2})$ a vyjděme z definice $\mathcal{O}$. Z ní plyne, že

$$\exists n_0 \in \mathbb{N}\, \exists c \in \mathbb{R}^+\, \forall n \geq n_0 : f_0(n) \leq c \cdot 2^{n^2}.$$

Stačí tedy ukázat, že existuje taková kladná konstanta c, aby rovněž

$$\lim_{n\to\infty} \frac{2^{2^n}}{c \cdot 2^{n^2}} > 1.$$

Upravujeme:

$$\lim_{n\to\infty} \frac{2^{2^n}}{c \cdot 2^{n^2}} = \frac{1}{c} \cdot \lim_{n\to\infty} 2^{(2^n - n^2)}.$$

Nyní pomocí Stolzovy-Cesàrovy věty ukažme, že posloupnost $a_n = 2^n$ roste od určité hodnoty rychleji než $b_n = n^2$, tedy že platí

$$\lim_{n\to\infty} = \frac{2^n}{n^2} = \infty.$$

Lze snadno nahlédnout, že

$$\lim_{n\to\infty} n^2 = \infty,$$

tedy první podmínka Stolzovy-Cesàrovy věty je splněna. Ověřme druhou podmínku:

$$\lim_{n\to\infty} \frac{2^{(n+1)} - 2^n}{(n+1)^2 - n^2} = \lim_{n\to\infty} \frac{2^n}{2 \cdot n + 1}.$$

Jelikož

$$\lim_{n\to\infty} 2 \cdot n + 1 = \infty,$$

je opět splněna první podmínka a ověřme druhou podmínku pro opakovanou aplikaci Stolzovy-Cesàrovy věty:

$$\lim_{n\to\infty} \frac{2^{(n+1)} - 2^n}{2 \cdot (n+1) + 1 - (2 \cdot n + 1)} = \lim_{n\to\infty} \frac{2^n}{2} = \infty,$$

tedy i

$$\lim_{n\to\infty} = \frac{2^n}{n^2} = \infty,$$

z čehož dále plyne, že

$$\frac{1}{c} \cdot \lim_{n\to\infty} 2^{(2^n - n^2)} = \infty,$$

pro kladné c. Rozdíl mezi 2^{2^n} a $c \cdot 2^{n^2}$ je tedy pro $n \to \infty$ kladný a stále se zvětšuje, takže

$$\exists c \in \mathbb{R}^+ \, \exists n_0 \in \mathbb{N} \, \forall n \geq n_0 : f_0(n) \leq c \cdot 2^{n^2} \leq 2^{2^n},$$

tedy $f_0(n) \in \mathcal{O}(2^{2^n})$. □

$\mathcal{O}(2^{n^2}) \neq \mathcal{O}(2^{2^n})$:

Důkaz. Zde postačí ukázat, že existuje funkce $f_0 \in \mathcal{O}(2^{2^n})$, pro kterou platí, že $f_0 \notin \mathcal{O}(2^{n^2})$. Zvolme $f_0(n) = 2^{2^n}$. V rámci důkazu sporem předpokládejme, že $f_0 \in \mathcal{O}(2^{n^2})$. Pak určitě platí, že

$$\exists n_0 \in \mathbb{N} \, \exists c \in \mathbb{R}^+ \, \forall n \geq n_0 : 2^{2^n} \leq c \cdot 2^{n^2}.$$

Vyšetřeme nyní limitní chování podílu těchto dvou posloupností v nekonečnu, tedy zjistěme, čemu se rovná limita

$$\lim_{n \to \infty} \frac{c \cdot 2^{n^2}}{2^{2^n}}.$$

Upravujeme:

$$\lim_{n \to \infty} \frac{c \cdot 2^{n^2}}{2^{2^n}} = c \cdot \lim_{n \to \infty} 2^{(n^2 - 2^n)}.$$

Už jsme ukázali, že posloupnost 2^n roste rychleji než n^2, pročež

$$\lim_{n \to \infty} n^2 - 2^n = -\infty$$

a tedy

$$c \cdot \lim_{n \to \infty} 2^{(n^2 - 2^n)} = 0$$

pro kladné c. Z předpokladu víme, že $\exists c \in \mathbb{R}^+ \, \exists n_0 \in \mathbb{N} \, \forall n \geq n_0 : 2^{2^n} \leq c \cdot 2^{n^2}$, neboli

$$1 \leq \frac{c \cdot 2^{n^2}}{2^{2^n}}.$$

Právě jsme ale ukázali, že

$$\lim_{n \to \infty} \frac{c \cdot 2^{n^2}}{2^{2^n}} = 0,$$

což je spor. Proto $2^{2^n} \notin \mathcal{O}(2^{n^2})$.

□

Ukázali jsme, že $\mathcal{O}(2^{n^2}) \subseteq \mathcal{O}(2^{2^n})$ a současně $\mathcal{O}(2^{n^2}) \neq \mathcal{O}(2^{2^n})$, pročež $\mathcal{O}(2^{n^2}) \subset \mathcal{O}(2^{2^n})$

□

h)

Důkaz. Jistě $\mathcal{O}(\ln(n)) \subset \mathcal{O}(n \cdot \ln(\ln(n)))$. Přímočarý důkaz tohoto tvrzení pomocí limit by byl náročný, pročež si raději uvědomme, že ostrá množinová inkluze je relací ostrého uspořádání (tj. ireflexivní částečné uspořádání), tedy pokud se nám povede ukázat, že $\mathcal{O}(\ln(n)) \subset \mathcal{O}(n) \subset \mathcal{O}(n \cdot \ln(\ln(n)))$, pak z tranzitivity ostrého uspořádání současně plyne $\mathcal{O}(\ln(n)) \subset \mathcal{O}(n \cdot \ln(\ln(n)))$. (Rovněž si všimněme, že stačí ukázat pouze $\mathcal{O}(\ln(n)) \subset \mathcal{O}(n) \subseteq \mathcal{O}(n \cdot \ln(\ln(n)))$, tj. u jedné z relací nemusíme ostrost ukazovat). Začněme důkazem $\mathcal{O}(\ln(n)) \subset \mathcal{O}(n)$. To ukážeme ve dvou krocích, tedy postupně ukážeme, že $\mathcal{O}(\ln(n)) \subseteq \mathcal{O}(n)$ a $\mathcal{O}(\ln(n)) \neq \mathcal{O}(n)$.

$\mathcal{O}(\ln(n)) \subseteq \mathcal{O}(n)$:

Důkaz. Uvažujme o libovolné funkci $f_0 \in \mathcal{O}(\ln(n))$ a vyjděme z definice $\mathcal{O}$. Z ní plyne, že

$$\exists n_0 \in \mathbb{N}\, \exists c \in \mathbb{R}^+\, \forall n \geq n_0 : f_0(n) \leq c \cdot \ln(n).$$

Stačí tedy ukázat, že existuje taková kladná konstanta c, aby rovněž

$$\lim_{n\to\infty} \frac{n}{c \cdot \ln(n)} > 1.$$

Ověřme, zda můžeme použít L'Hospitalovo pravidlo pro limitu podílu dvou posloupností. Zavedeme $f(x) = x$ a $g(x) = c \cdot \ln(x)$. Zřejmě $\forall n \in \mathbb{N} : f(n) = n \wedge g(n) = \ln(n)$. Dále je zřejmé, že

$$\lim_{x\to\infty} f(x) = \lim_{x\to\infty} g(x) = \infty.$$

Můžeme tedy psát:

$$\lim_{n\to\infty} \frac{n}{c \cdot \ln(n)} = \lim_{x\to\infty} \frac{f'(x)}{g'(x)} = \frac{1}{c} \cdot \lim_{x\to\infty} \frac{1}{\frac{1}{x}} = \frac{1}{c} \cdot \lim_{x\to\infty} x = \infty > 1$$

pro kladné c. Rozdíl mezi n a $c \cdot \ln n$ je tedy pro $n \to \infty$ kladný a stále se zvětšuje, takže

$$\exists c \in \mathbb{R}^+\, \exists n_0 \in \mathbb{N}\, \forall n \geq n_0 : f_0(n) \leq c \cdot \ln n \leq n,$$

tedy $f_0(n) \in \mathcal{O}(n)$.

□

$\mathcal{O}(\ln(n)) \neq \mathcal{O}(n)$:

Důkaz. Zde postačí ukázat, že existuje funkce $f_0 \in \mathcal{O}(n)$, pro kterou platí, že $f_0 \notin \mathcal{O}(\ln(n))$. Zvolme $f_0(n) = n$. V rámci důkazu sporem předpokládejme, že $f_0 \in \mathcal{O}(\ln(n))$. Pak určitě platí, že

$$\exists n_0 \in \mathbb{N}\, \exists c \in \mathbb{R}^+\, \forall n \geq n_0 : n \leq c \cdot \ln n.$$

Vyšetřeme nyní limitní chování podílu těchto dvou posloupností v nekonečnu, tedy zjistěme, čemu se rovná limita

$$\lim_{n\to\infty} \frac{c \cdot \ln(n)}{n}.$$

Zavedeme dvě spojité funkce $f(x) = c \cdot \ln(x)$ a $g(x) = x$. Všechny podmínky jsme ověřili v předchozím kroku, proto můžeme psát:

$$\lim_{n\to\infty} \frac{c \cdot \ln(n)}{n} = \lim_{x\to\infty} \frac{f'(x)}{g'(x)} = c \cdot \lim_{x\to\infty} \frac{\frac{1}{x}}{1} = c \cdot \lim_{x\to\infty} \frac{1}{x} = 0$$

pro kladná c. Z předpokladu víme, že $\exists c \in \mathbb{R}^+ \, \exists n_0 \in \mathbb{N} \, \forall n \geq n_0 : n \leq c \cdot \ln n$, neboli

$$1 \leq \frac{c \cdot n}{\ln n}.$$

Právě jsme ale ukázali, že

$$\lim_{n\to\infty} \frac{c \cdot n}{\ln n} = 0,$$

což je spor. Proto $n \notin \mathcal{O}(\ln n)$.

□

Ukázali jsme tedy, že $\mathcal{O}(\ln(n)) \neq \mathcal{O}(n)$ a současně $\mathcal{O}(\ln(n)) \subseteq \mathcal{O}(n)$, pročež i $\mathcal{O}(\ln(n)) \subset \mathcal{O}(n)$. Nyní ukážeme druhou inkluzi.

$\mathcal{O}(n) \subseteq \mathcal{O}(n \cdot \ln(\ln(n)))$:

Důkaz. Uvažujme o libovolné funkci $f_0 \in \mathcal{O}(n)$ a vyjděme z definice $\mathcal{O}$. Z ní plyne, že

$$\exists n_0 \in \mathbb{N} \, \exists c \in \mathbb{R}^+ \, \forall n \geq n_0 : f_0(n) \leq c \cdot n.$$

Stačí tedy ukázat, že existuje taková kladná konstanta c, aby rovněž

$$\lim_{n\to\infty} \frac{n \cdot \ln(\ln(n))}{c \cdot n} > 1.$$

Můžeme psát

$$\lim_{n\to\infty} \frac{n \cdot \ln(\ln(n))}{c \cdot n} = \frac{1}{c} \cdot \lim_{n\to\infty} \ln(\ln(n)) = \infty > 1$$

pro kladné c. Rozdíl mezi n a $c \cdot n \cdot \ln \ln n$ je tedy pro $n \to \infty$ kladný a stále se zvětšuje, takže

$$\exists c \in \mathbb{R}^+ \, \exists n_0 \in \mathbb{N} \, \forall n \geq n_0 : f_0(n) \leq c \cdot n \leq n \cdot \ln \ln n,$$

tedy $f_0(n) \in \mathcal{O}(n \cdot \ln \ln n)$.

□

Podařilo se nám ukázat, že $\mathcal{O}(\ln(n)) \subset \mathcal{O}(n) \subseteq \mathcal{O}(n \cdot \ln(\ln(n)))$. Z tranzitivity tedy plyne inkluze tříd $\mathcal{O}(\ln(n)) \subset \mathcal{O}(n \cdot \ln(\ln(n)))$.

□

i)

Důkaz. Jistě $\mathcal{O}(\log(n)) = \mathcal{O}(\ln(n))$. Uvažujme o libovolných funkcích $f_0 \in \mathcal{O}(\log(n))$ a $g_0 \in \mathcal{O}(\ln(n))$. Pak určitě

$$\exists c_f \in \mathbb{R}^+ \, \exists n_0 \in \mathbb{N} \, \forall n \in \mathbb{N} : n \geq n_0 \Rightarrow 0 \leq f_0(n) \leq c_f \cdot \log(n)$$

a rovněž

$$\exists c_g \in \mathbb{R}^+ \, \exists n_0 \in \mathbb{N} \, \forall n \in \mathbb{N} : n \geq n_0 \Rightarrow 0 \leq g_0(n) \leq c_g \cdot \ln(n)$$

Jelikož existuje $n'_0 \in \mathbb{N}, c'_f = \frac{c_f}{\ln(10)}$ tak, že

$$\forall n \geq n'_0 : c_f \cdot \log(n) \leq c'_f \cdot \ln(n),$$

pak $\mathcal{O}(\log(n)) \subseteq \mathcal{O}(\ln(n))$.

Podobně můžeme ukázat, že existuje $n''_0 \in \mathbb{N}, c'_g = \frac{c_g}{\log(e)}$ tak, že

$$\forall n \geq n''_0 : c_g \cdot \ln(n) \leq c'_g \cdot \log(n),$$

z čehož plyne $\mathcal{O}(\ln(n)) \subseteq \mathcal{O}(\log(n))$.

Ukázali jsme, že $\mathcal{O}(\ln(n)) \subseteq \mathcal{O}(\log(n))$ a současně $\mathcal{O}(\log(n)) \subseteq \mathcal{O}(\ln(n))$, tedy $\mathcal{O}(\log(n)) = \mathcal{O}(\ln(n))$. □

Sekce 20: Konstrukce TS

Úloha 1

Snažte se najít co nejmenší k takové, aby platily následující vztahy. Dokažte, že vztah platí, ale nedokazujte, že vaše k je opravdu nejmenší možné

(b) $L' \in DTIME[n^2] \Rightarrow L \in NTIME[n^k]$, kde
$L = \{w \in \Sigma^* \mid$ pro všechny prefixy w' slova w platí, že $w' \in L'\}$
(c) $L' \in DTIME[n^3] \Rightarrow L \in DTIME[n^k]$, kde
$L = \{w \in \Sigma^* \mid w$ obsahuje dvě různá podslova w_1, w_2 taková, že $w_1, w_2 \in L'\}$

b)

Předpokládejme, že zadaný jazyk L' patřící do $DTIME[n^2]$ je přijímán nějakým deterministickým $m-$páskovým Turingovým strojem M'. Stroj M' rozhoduje L' v čase $\mathcal{O}(n^2)$. Sestrojme nyní deterministický $(m+2)$-páskový Turingův stroj M, jenž přijímá jazyk L, následovně:

První páska stroje M je vstupní, tedy na počátku simulace má tvar $\Delta w \Delta\Delta^\omega$. Stroj pro každé i, kde $0 \leq i \leq |w|$, postupně kopíruje prvních i symbolů řetězce w na druhou pásku, která tedy vždy bude mít tvar $\Delta x \Delta\Delta^\omega$, kde $w = xx'$ a $|x| = i$. Tato páska tedy bude sloužit jako počítadlo délek dosud zkontrolovaných prefixů. Po překopírování předpony slova w na druhou pásku dojde k překopírování téže předpony na třetí pásku, ta tedy bude mít rovněž tvar $\Delta x \Delta\Delta^\omega$. Ostatní pásky budou prázdné (tj. budou mít tvar $\Delta\Delta^\omega$).
Stroj M následně odsimuluje na $3.-(m+2).$ pásce stroj M', který je uložen ve stavovém řízení M. Pokud stroj M' odmítne, odmítne i M, protože existuje prefix w, který M' nepřijímá. Pokud stroj M' akceptuje, provede se úklid na $3.-(m+2).$ pásce, aby tyto pásky měly opět tvar $\Delta\Delta^\omega$. Hlavy se u těchto pásek posunou na začátek. Dále se pokračuje překopírováním nového prefixu na druhou pásku. Pokud se ukáže, že již byly vyčerpány všechny předpony a tedy nelze odstartovat další iteraci, M akceptuje, neboť všechny předpony byly přijaty.

Testujeme $\mathcal{O}(n+1)$ různých slov. Pro každé slovo musíme překopírovat nový symbol na druhou pásku, což odpovídá složitosti $\mathcal{O}(1)$ a následně celé toto slovo překopírujeme na třetí pásku, což odpovídá $\mathcal{O}(n)$. Simulace M' nad předponou má dle zadání složitost $\mathcal{O}(n^2)$. Jelikož není jasné, kolik prostoru na páskách 3 až $(m+2)$ zabraly simulace, musíme předpokládat, že i úklid bude mít složitost $\mathcal{O}(n^2)$. Celková složitost tedy odpovídá:

$$\mathcal{O}(n+1)(\mathcal{O}(1) + \mathcal{O}(n) + \mathcal{O}(n^2) + \mathcal{O}(n^2)) = \mathcal{O}(n^3)$$

Řešením tedy je $k = 3$. Je tedy zřejmé, že nedeterminismus nám v tomto případě nijak nepomůže, neboť musíme zkontrolovat veškeré předpony, abychom měli jistotu, že jsou všechny přijímány.

c)

Předpokládejme, že zadaný jazyk L' patřící do $DTIME[n^3]$ je přijímán nějakým deterministickým $m-$páskovým Turingovým strojem M'. Stroj M' rozhoduje L' v čase $\mathcal{O}(n^3)$. Sestrojme nyní deterministický $(m+3)$-páskový Turingův stroj M, jenž přijímá jazyk L, následovně:

První páska stroje M je vstupní, tedy na počátku simulace má tvar $\Delta w \Delta\Delta^\omega$. Stroj postupně generuje veškeré podřetězce w na druhou pásku následujícím způsobem. Druhá páska má nejprve tvar $\Delta\Delta^\omega$. Za první symbol Δ jsou postupně v dalším iteracích přidávány symboly z řetězce w, tedy druhá páska má postupně tvar $\Delta\Delta^\omega, \Delta\sigma_1\Delta^\omega, \Delta\sigma_1\sigma_2\Delta^\omega, \ldots, \Delta\sigma_1\sigma_2\ldots\sigma_n\Delta^\omega$, kde $\sigma_1\sigma_2\ldots\sigma_n = w$. Následně jsou vždy smazány všechny

symboly kromě druhého a opět se pokračuje přidáváním jednotlivých symbolů, páska má tedy postupně tvar $\Delta\Delta\sigma_2\Delta^\omega, \Delta\Delta\sigma_2\sigma_3\Delta^\omega, \ldots, \Delta\Delta\sigma_2\sigma_3\ldots\sigma_n\Delta^\omega, \Delta\Delta\Delta\sigma_3\Delta^\omega, \Delta\Delta\Delta\sigma_3\sigma_4\Delta^\omega, \Delta\Delta\Delta\sigma_3\sigma_4\ldots\sigma_n\Delta^\omega, \ldots,$ dokud nedojde k vyčerpání všech podřetězců w. Po vygenerování nového podřetězce je hlava položena na poslední symbol Δ před daným podřetězcem, pročež vícenásobný výskyt Δ před podřetězci nevadí.
Třetí páska bude sloužit pouze k uchování informace, zda už byl některý podřetězec přijat. Zpočátku tedy má tvar $\Delta\Delta^\omega$.
Na čtvrtou pásku se v každé iteraci překopíruje aktuálně zkoumaný podřetězec z druhé pásky. Ostatní pásky jsou prázdné, mají tedy tvar $\Delta\Delta^\omega$.
Stroj M spustí na $4.-(m+3).$ pásce simulaci stroje M'. Pokud simulace skončí odmítnutím řetězce, vygeneruje se na druhé pásce výše popsaným způsobem nový podřetězec slova w, obsah $4.-(m+3).$ pásky se smaže a odstartuje se další iterace. Pokud simulace skončí odmítnutím a už neexistuje další dosud nepoužitý podřetězec slova w, který by mohl být vygenerován, M odmítne. Pokud simulace skončí přijetím a třetí páska má tvar $\Delta\Delta^\omega$, změní stroj M její obsah na $\Delta\#\Delta^\omega$ a po smazání obsahu $4.-(m+3).$ pásky se odstartuje další iterace (za předpokladu, že existuje další dosud nepoužitý podřetězec w, jinak M odmítne). Pokud simulace skončí přijetím a třetí páska má tvar $\Delta\#\Delta^\omega$, stroj M akceptuje, neboť jsme našli již druhé podslovo w, které M' přijímá.

Je zřejmé, že slovo w delky $|w| = n$ má dohromady až

$$1 + \sum_{i=1}^{n} i = 1 + \frac{n + n^2}{2}$$

různých podslov, což odpovídá $\mathcal{O}(n^2)$. Pro každé toto podslovo je třeba překopírovat nový symbol na druhou pásku, případně tuto pásku kromě jednoho symbolu vyprázdnit, což odpovídá složitosti $\mathcal{O}(n)$. Každé toto slovo dále překopírujeme na čtvrtou pásku, což lze realizovat v čase $\mathcal{O}(n)$. Simulace stroje M' nad podslovem odpovídá dle zadání $\mathcal{O}(n^3)$. Jelikož není jasné, kolik prostoru na páskách 4 až $(m+3)$ zabraly simulace, musíme předpokládat, že i úklid bude mít složitost $\mathcal{O}(n^3)$. Kontrola jednoho symbolu na třetí pásce a případné zapsání $\#$ má složitost $\mathcal{O}(1)$. Celková složitost tedy odpovídá:

$$\mathcal{O}(n^2)(\mathcal{O}(n) + \mathcal{O}(n) + \mathcal{O}(n^3) + \mathcal{O}(n^3) + \mathcal{O}(1)) = \mathcal{O}(n^5)$$

Řešením je tedy $k = 5$. Poznamenejme, že v tomto případě by nedeterminismus byl užitečný, neboť by stačilo zkontrolovat 2 různá nedetrministicky zvolená podslova a složitost by byla $\mathcal{O}(n^3)$.

Úloha 2

Rozhodněte a dokažte, zda třída **P** je uzavřena na následující operace

(b) $\diamond L = \{w \in L \mid w = 2^n \wedge n \geq 0\}$
(c) $\diamond L = \{w$ je kód TS M, pro který platí, že $w \in L(M)\}$

b)

Třída **P** je uzavřena na operaci $\diamond$. Jazyk $\diamond L = L \cap L'$, kde $L' = \{w \in \Sigma^* \mid w = 2^n \wedge n \geq 0\}$. Jelikož třída **P** je uzavřena na průnik, je třeba rozhodnout, zda $L' \in$ **P**.

Musíme popsat chování Turingova stroje DIV_2, který provádí dělení vstupního čísla dvěma. Stroj definujeme jako třípáskový. Abecedu stroje DIV_2 tvoří desetiprvková množina číslic $0-9$. Na první pásce má uložen svůj vstup w, tato páska má tedy tvar $\Delta w\Delta\Delta^\omega$. Ostatní pásky jsou zpočátku prázdné, mají tedy tvar $\Delta\Delta^\omega$. Stroj postupuje od první číslice σ_1 (tj. $w = \sigma_1 w'$). Je-li tato číslice sudá, zapíše stroj na druhou pásku její podíl dvěma, druhá páska má tedy tvar $\Delta\sigma_{1'}\Delta\Delta^\omega$, kde $\sigma_1 = 2 \cdot \sigma_{1'}$. Informaci o správném výsledku podílu

má stroj DIV_2 zakódován ve svém stavovém řízení. Je-li tato první číslice lichá, zapíše stroj na druhou pásku její podíl dvěma zaokrouhlený dolů, druhá páska má tedy tvar $\Delta\sigma_{1'}\Delta\Delta^\omega$, kde $\sigma_1 = 2 \cdot \sigma_{1'} + 1$. Současně se na třetí pásku poznamená příznak přenosu, třetí páska má tedy tvar $\Delta\#\Delta\Delta^\omega$. Hlava na první pásce se posune na další symbol. Pokud není nastaven příznak přenosu, dělíme σ_2 (kde $w = \sigma_1\sigma_2 w''$) dvěma podobně, jak je popsáno výše. Pokud je nastaven příznak přenosu, dělíme $10 + \sigma_2$ dvěma. Pokud je výsledek operace sudé číslo a byl nastaven příznak přenosu, tento příznak se smaže, třetí páska tudíž bude mít tvar $\Delta\Delta^\omega$. Pokud je výsledek operace liché číslo a příznak nebyl nastaven, nastaví se, třetí páska tudíž bude mít tvar $\Delta\#\Delta^\omega$. Ve zbylých případech zůstane třetí páska nezměněna. Mezivýsledky dělení se kumulují na druhé pásce, ta má tedy nyní tvar $\Delta\sigma_{1'}\sigma_{2'}\Delta\Delta^\omega$, kde $\sigma_{2'}$ je opět podíl symbolu σ_2 po dělení dvěma (případně zaokrouhlený dolů).
Tímto způsobem provádíme dělení tak dlouho, dokud nevyčerpáme veškeré vstupní symboly. Pokud je na konci procesu nastaven příznak přenosu, DIV_2 odmítne, jinak akceptuje. Lze snadno nahlédnout, že stroj pracuje se složitostí $\mathcal{O}(n)$.

Nyní popišme chování Turingova stroje POW_2, který kontroluje, zda je vstupní řetězec $n.$ mocninou čísla 2 (pro $n \in \mathbb{N}$). Stroj definujeme jako třípáskový. Na první pásce má POW_2 uložen svůj vstup w, páska má tedy tvar $\Delta w\Delta\Delta^\omega$. Zbylé pásky jsou zpočátku prázdné, mají tedy tvar $\Delta\Delta^\omega$. Stroj nejprve zkontroluje, zda je tvar první pásky roven $\Delta 0\Delta\Delta^\omega$. Pokud ano, odmítne, jinak pokračuje. Dále zkontroluje, zda je tvar první pásky roven $\Delta 1\Delta\Delta^\omega$. Pokud ano, přijme, jinak pokračuje. Dále zkontroluje, zda se vstup w skládá pouze z číslic. Pokud ne, odmítne, jinak pokračuje. Poté simuluje na všech svých páskách stroj DIV_2, který je uložen v jeho stavovém řízení. Pokud DIV_2 odmítne, odmítne i POW_2. Pokud DIV_2 akceptuje, stroj POW_2 zkontroluje, zda je obsah jeho druhé pásky roven $\Delta 1\Delta\Delta^\omega$. (Připoměňme, že na této pásce bude uložen výsledek operace dělení dvěma.) Pokud je na této pásce pouze číslo 1, stroj POW_2 akceptuje, jinak pokračuje. Stroj překopíruje obsah druhé pásky na první pásku a smaže obsah druhé a třetí pásky. Následně začne znovu se simulací stroje DIV_2. Tento postup opakuje tak dlouho, dokud DIV_2 neodmítne, nebo dokud na druhé pásce nebude pouze symbol 1. Poznamenejme, že jeden z těchto případů musí nastat, neboť opakovaným dělením dvěma lze cyklit pouze na čísle 0, což jsme ošetřili výše.

Počáteční kontrola vstupu odpovídá složitosti $\mathcal{O}(n) + \mathcal{O}(1) + \mathcal{O}(1)$. Každé dělení vyžaduje spuštění stroje DIV_2, přičemž jedno spuštění odpovídá složitosti $\mathcal{O}(n)$. Následné překopírování výstupu na první pásku a úklid zbylých dvou pásek odpovídá složitosti $\mathcal{O}(n) + \mathcal{O}(n) + \mathcal{O}(1)$. Podstatné dále je, kolikrát bude pro vstup délky n spuštěn proces dělení dvěma. Pro kladné hodnoty o délce n zřejmě platí, že nejvyšší číslo zapsatelné pomocí n cifer je $10^{n+1} - 1$ (tedy např. 9999 pro $n = 4$). Dále je vhodné si uvědomit, že pro nějaké kladné číslo c provedeme vždy nejvýše $\log_2(c)$ dělení dvěma. Z toho nám vyplývá, že pro vstup o délce n provedeme nejvýše

$$\log_2(10^{n+1} - 1)$$

dělení dvěma. Zřejmě

$$\log_2(10^{n+1} - 1) \leq \log_2(10^{n+1})$$

a

$$\log_2(10^{n+1}) = (n+1) \cdot \log_2(10) = \frac{(n+1) \cdot \log 10}{\log 2} = \frac{n+1}{\log 2},$$

tedy počet dělení dvěma je nejhůře lineární vzhledem k délce vstupu.

Celková složitost tedy odpovídá:

$$\mathcal{O}(n) + \mathcal{O}(1) + \mathcal{O}(1) + \mathcal{O}(n)(\mathcal{O}(n) + \mathcal{O}(n) + \mathcal{O}(n) + \mathcal{O}(1)) = \mathcal{O}(n^2).$$

Z toho plyne, že POW_2 pracuje v polynomiálním čase. Jelikož jsme ukázali, že $\diamond L = L \cap L'$ a právě jsme dokázali, že L' lze přijmout v polynomiálním čase, pak je třída **P** uzavřena na operaci $\diamond$.

c)

Třída **P** není uzavřena na operaci $\diamond$. V rámci protipříkladu uvažujme o jazyku $L = \Sigma^*$. Tento jazyk jistě leží ve třídě **P**, neboť lze rozhodnout Turingovým strojem, který pro každý svůj vstup okamžitě akceptuje. Jeho složitost je tedy v $\mathcal{O}(1)$. Jelikož $\diamond L = SELFREF$, kde $SELFREF = \{w \in \Sigma^* \mid w$ je kód TS M, kde $w \in L(M)\}$, je třeba ukázat, že $SELFREF \notin$ **P**. Z toho poté vyplyne neuzavřenost třídy **P** na operaci $\diamond$.

Pro spor nyní předpokládejme, že existuje deterministický Turingův stroj $M_{SELFREF}$, který přijímá $SELFREF$ v čase $\mathcal{O}(p)$, kde p je nějaký polynom, tedy předpokládejme, že $SELFREF \in$ **P**. Potom musí existovat deterministický Turingův stroj $M_{co-SELFREF}$ přijímající $co - SELFREF$ rovněž v polynomiálním čase. To ukážeme popisem chování $M_{co-SELFREF}$.
Stroj $M_{co-SELFREF}$ nejprve pro svůj vstup w' vyhodnotí $p(|w'|)$, tedy maximální počet kroků, který může $M_{SELFREF}$ díky příslušnosti své časové složitosti do $\mathcal{O}(p)$ provést. Stroj $M_{co-SELFREF}$ následně provede nejvýše $p(|w'|)$ kroků simulace $M_{SELFREF}$ nad w'. Pokud simulace skončila do $p(|w'|)$ kroků, stroj $M_{co-SELFREF}$ invertuje její výsledek a ten vrátí. Pokud simulace neskončila do $p(|w'|)$ kroků, stroj $M_{co-SELFREF}$ akceptuje, neboť je jasné, že $M_{SELFREF}$ tento řetězec přijmout nemůže, neboť jsme počtem provedených kroků přesáhli maximální hranici danou příslušností do $\mathcal{O}(p)$. Je tedy jasné, že $L(M_{co-SELFREF}) = co - SELFREF$ a současně $co - SELFREF \in$ **P**.

Nyní ukažme, že jazyk $SELFREF$ je částečně rozhodnutelný, ale je nerozhodnutelný. Částečná rozhodnutelnost je zřejmá, neboť pokud je w korektní kód TS, můžeme spustit simulaci M_w na w a pokud M_w akceptuje, akceptuje i $SELFREF$, jinak odmítne, nebo cyklí. Nerozhodnutelnost ukážeme pomocí redukce z HP, tedy ukážeme, že $HP \leq SELFREF$. Požadovanou redukční funkci $\sigma : \{0, 1, \#\}^* \to \{0, 1\}^*$ definujeme takto:

$$\sigma(\langle M \rangle \# \langle w \rangle) = \langle M' \rangle.$$

Pokud $\langle M \rangle \# \langle w \rangle$ není korektní instance HP, vrací σ kód takového TS M', že $L(M') = \{\}$, tj. určitě $\langle M' \rangle \notin SELFREF$, neboť takový stroj nepřijímá žádný řetězec, tedy nemůže přijímat ani svůj kód. Jinak vrací redukční funkce σ kód TS M', který pracuje následovně:

- Stroj M' ignoruje svůj vstup w' a spustí simulaci M na w. Poznamenejme, že kódy M a w jsou zakódovány ve stavovém řízení M'.
- Pokud simulace M na w cyklí, cyklí i M'.
- Pokud simulace M na w skončí, M' akceptuje.

Všimněme si, že pokud M na w cyklí, tedy $\langle M \rangle \# \langle w \rangle \notin HP$, pak M' nepřijímá žádný řetězec, tedy nemůže přijímat ani svůj vlastní kód, pročež $\langle M' \rangle \notin SELFREF$. Pokud však M na w zastaví, tedy $\langle M \rangle \# \langle w \rangle \in HP$, pak M' přijímá všechny řetězce ze Σ^*, tedy určitě přijímá i svůj vlastní kód, pročež $\langle M' \rangle \in SELFREF$.

Pro M' tedy platí:

$$\langle M \rangle \# \langle w \rangle \in HP \Rightarrow L(M') = \Sigma^* \Rightarrow \langle M' \rangle \in SELFREF$$

$$\langle M \rangle \# \langle w \rangle \notin HP \Rightarrow L(M') = \{\} \Rightarrow \langle M' \rangle \notin SELFREF$$

$$\langle M \rangle \# \langle w \rangle \in HP \iff \sigma(\langle M \rangle \# \langle w \rangle) \in SELFREF$$

Tato redukce je správná, neboť σ zachovává příslušnost a současně je to totální, rekurzivně vyčíslitelná funkce.

Právě jsme pomocí redukce ukázali, že $SELFREF$ je rekurzivně vyčíslitelný, nerekurzivní jazyk. Z toho ale plyne, že $co - SELFREF$ není ani částečně rozhodnutelný problém, tedy neexistuje žádný Turingův stroj, který přijímá $co - SELFREF$. Předtím jsme však ukázali, že existuje deterministický Turingův stroj $M_{co-SELFREF}$, který $co - SELFREF$ rozhoduje v polynomiálním čase. Došli jsme tedy ke sporu, který vychází z nesprávného předpokladu existence deterministického Turingova stroje přijímajícího $SELFREF$ v polynomiálním čase p. Z toho tedy plyne, že $SELFREF \notin \mathbf{P}$.

Ukázali jsme, že $SELFREF \notin \mathbf{P}$. Dříve jsme zavedli $L = \Sigma^*$ a ukázali jsme, že $L \in \mathbf{P}$. Jelikož $\diamond L = SELFREF$ a $SELFREF \notin \mathbf{P}$, pak třída $\mathbf{P}$ není uzavřena na operaci $\diamond$.

Sekce 21: Třídy složitosti a NP-úplnost

Úloha 1

Pro následující jazyky/problémy dokažte, že

- patří do **NP**
- patří do **EXP** (v důkaze nevyužívejte fakt, že **NP** $\subseteq$ **EXP**)
- jsou **NP**-těžké

(b) *Nezávislá množina vrcholů* grafu $G = (V, E)$ je množina vrcholů $U \subseteq V$ taková, že žádné dva vrcholy z U nejsou spojeny hranou, tj. $\forall\{v_1, v_2\} \in E : \{v_1, v_2\} \not\subseteq U$. *Problém nezávislé množiny* (INDEPENDENT-SET):

Má graf G nezávislou množinu velikosti alespoň m?

Pro důkaz **NP**-těžkosti použijte redukci z 3-SAT.

(c) Mějme konečnou množinu R, váhovou funkci $w : R \to \mathbb{Z}$ a cenovou funkci $v : R \to \mathbb{Z}$. Dále mějme maximální váhu $W \in \mathbb{Z}$ a minimální cenu $V \in \mathbb{Z}$. *Problém batohu* (KNAPSACK) je definovaný následovně:

$$\text{Existuje } R' \subseteq R \text{ taková, že } \sum_{r \in R'} w(r) \leq W \text{ a zároveň } \sum_{r \in R'} v(r) \geq V?$$

Pro důkaz **NP**-těžkosti použijte redukci z problému SUBSET-SUM.

(d) * Graf $G = (V, E)$ má *chromatické číslo c*, pokud existuje funkce *obarvení* $f : V \to \{1, \ldots, c\}$ taková, že pro každé dva sousední uzly platí, že *nejsou* obarveny stejnou barvou, tj $\forall\{v_1, v_2\} \in E : f(v_1) \neq f(v_2)$. *Problém chromatického čísla grafu* (CHROMATIC-NUMBER) je definovaný následovně:

Má graf G chromatické číslo c?

Pro důkaz **NP**-těžkosti použijte redukci z problému 3-SAT.

(e) *Klikové pokrytí grafu* $G = (V, E)$ je množina klik K taková, že G je sjednocení klik z K. *Problém klikového pokrytí* (CLIQUE-COVER) je definovaný následovně:

Existuje množina klik K o velikosti maximálně m taková, že K je klikové pokrytí G?

Pro důkaz **NP**-těžkosti použijte redukci z problému CHROMATIC-NUMBER.

b)

Příslušnost do **NP**:

Uvažujme o vícepáskovém nedeterministickém Turingově stroji M rozhodujícím $INDEPENDENT-SET$ v polynomiálním čase, který pracuje následovně. Na své první pásce má stroj svůj vstup $\Delta\langle G\rangle\#\langle m\rangle\Delta\Delta^\omega$, kde $\langle G\rangle$ je kód grafu $G = (V, E)$ a $\langle m\rangle$ je kód přirozeného čísla m. Stroj nejprve zkontroluje validitu vstupu, což se mu podaří v kubickém čase $\mathcal{O}(n^3)$. Pokud je vstup invalidní, odmítne, jinak pokračuje. Stroj M následně na svou druhou pásku nedeterministicky vygeneruje množinu $U \subseteq V$, tedy množinu některých vrcholů grafu. To lze provést v čase $\mathcal{O}(n)$. Stroj M dále zkontroluje, zda má tato množina alespoň velikost m. Pokud ne, odmítne, jinak pokračuje. Tuto kontrolu lze rovněž provést v

lineárním čase $\mathcal{O}(n)$. Stroj následně pro každou dvojici vrcholů z U projde množinu hran E a zkontroluje, zda jsou tyto dva vrcholy spojeny hranou, nebo ne. Pokud je taková hrana detekována, M odmítne. Pokud se ukáže, že žádné dva vrcholy z U nejsou spojeny hranou, M akceptuje. Toto lze zřejmě provést v čase $\mathcal{O}(n^4)$, neboť všech dvojic vrcholů v U je $\mathcal{O}(n^2)$, kontrola každé dvojice vyžaduje průchod množiny hran o složitosti $\mathcal{O}(n)$ a porovnání o složitosti $\mathcal{O}(n)$. Nedeterministický stroj M tedy pracuje se složitostí $\mathcal{O}(n^4)$, pročež $INDEPENDENT - SET \in$ **NP**.

Příslušnost do **EXP**:

Uvažujme o vícepáskovém deterministickém Turingově stroji M rozhodujícím $INDEPENDENT-SET$ v exponenciálním čase, který pracuje následovně. Na své první pásce má stroj svůj vstup $\Delta\langle G\rangle\#\langle m\rangle\Delta\Delta^\omega$, kde $\langle G\rangle$ je kód grafu $G = (V, E)$ a $\langle m\rangle$ je kód přirozeného čísla m. Stroj nejprve zkontroluje validitu vstupu, což se mu podaří v kubickém čase $\mathcal{O}(n^3)$. Dále na druhou pásku postupně generuje všechny podmnožiny $U \subseteq V$ o mohutnosti m a vyšší, kterých je v nejhorším případě $2^{|V|}$. Toto provádí od nejméně mohutných množin po nejmohutnější, přičemž jednotlivé množiny stejné mohutnosti generuje v lexikografickém pořadí prvků těchto množin. Stroj následně pro každou dvojici vrcholů z U projde množinu hran E a zkontroluje, zda jsou tyto dva vrcholy spojeny hranou, nebo ne. Pokud je taková hrana detekována, M přejde k vygenerování další množiny. Pokud se ukáže, že žádné dva vrcholy z U nejsou spojeny hranou, M akceptuje. Pokud stroj vyčerpá veškeré takto generované množiny U aniž by přijal, odmítne. Toto lze zřejmě provést v čase $\mathcal{O}(n^4)$, neboť všech dvojic vrcholů v U je $\mathcal{O}(n^2)$, kontrola každé dvojice vyžaduje průchod množiny hran o složitosti $\mathcal{O}(n)$ a porovnání o složitosti $\mathcal{O}(n)$. Je tedy zřejmé, že deterministický stroj M pracuje v čase $\mathcal{O}(n^4 \cdot 2^n) \subseteq \mathcal{O}(2^{n^2})$, a tudíž $INDEPENDENT - SET \in$ **EXP**.

NP-těžkost:

NP-těžkost dokážeme pomocí polynomiální redukce z problému SAT_3 (problém, zda je množina klauzulí o třech literálech splnitelná). Ukážeme tedy $SAT_3 \leq_P^m INDEPENDENT - SET$. Toto nám bude stačit jako důkaz, neboť SAT_3 je **NP**-úplný. Redukční funkci f definujeme takto:

$$f(\langle\Phi\rangle) = \langle G\rangle\#\langle n\rangle$$

kde

$$\Phi = \bigwedge_{i=1}^{n} \varphi_i$$

je množina klauzulí a

$$\forall i \in \mathbb{N} : 1 \leq i \leq n \Rightarrow \varphi_i = l_{i,1} \vee l_{i,2} \vee l_{i,3},$$

tedy φ_i je klauzule o třech literálech, kde

$$\forall j \in \{1, 2, 3\} : l_{i,j} = x_l \vee l_{i,j} = \overline{x_l},$$

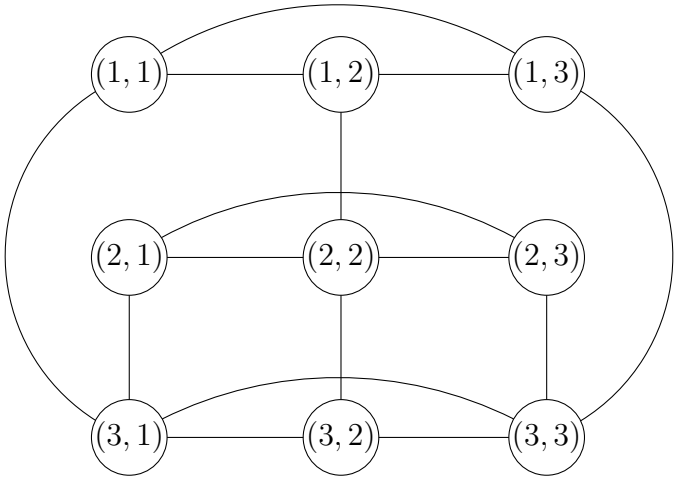
kde x_l je nějaká proměnná z množiny $X = \{x_1, x_2, \ldots x_k\}$ a $\langle n\rangle$ je kód přirozeného čísla n, kde n je počet klauzulí Φ. Poznamenejme, že pokud existuje množina $Val \subseteq X$, kde všem proměnným z množiny Val přiřadíme pravdivostní hodnotu $\top$ a ostatním pravdivostní hodnotu $\bot$, a s takovým ohodnocením je formule Φ vyhodnocena jako $\top$, pak je Φ splnitelná.
Pokud formule Φ nemá výše uvedený formát, pak redukční funkce vrací kód prázdného grafu $G = (\{\}, \{\})$ a kód $\langle 1\rangle$ přirozeného čísla 1. Tento graf jistě nemá nezávislou množinu o jednom nebo více prvcích. Jinak vrací kód následujícího grafu $G = (V, E)$:

- $V = \{(i,j) \mid i \in \mathbb{N} \wedge 1 \leq i \leq n \wedge j \in \{1,2,3\}\}$
- $E = \{\{(i,j),(i',j')\} \mid (i = i' \vee l_{i,j} = \overline{l_{i',j'}}) \wedge (i,j) \neq (i',j')\}$

Neformálně řečeno reprezentuje každý uzel daného grafu jeden literál formule Φ. Dva uzly jsou spojené hranou právě tehdy, pokud jsou dané literály součástí jedné klauzule, nebo pokud by jejich konjunkce tvořila kontradikci (tj. např. $x \wedge \overline{x}$). Současně není žádný uzel spojen hranou sám se sebou. Pro názornou ukázku uvažujme o následující formuli:

$$\Phi_0 = (x_1 \vee \overline{x_2} \vee x_3) \wedge (x_1 \vee x_2 \vee x_3) \wedge (\overline{x_1} \vee \overline{x_2} \vee \overline{x_3}),$$

kterou redukční funkce f zobrazí na graf:



Všimněme si, že kupříkladu ulzy $(1,1)$ a $(1,2)$ jsou spojeny hranou, neboť odpovídají literálům ze stejné klauzule. Dále uzly $(1,2)$ a $(2,2)$ jsou spojeny hranou proto, že uzel $(1,2)$ odpovídá litrálu $\overline{x_2}$ a uzel $(2,2)$ odpovídá literálu x_2. Přitom platí $\overline{x_2} \wedge x_2 \iff \bot$.

Lze snadno nahlédnout, že pro formuli Φ_0 existuje ohodnocení $Val = \{x_2, x_3\}$, pro nějž je formule splněna. Tomu odpovídá fakt, že množina vrcholů $\{(3,1),(2,2),(1,3)\}$ neobsahuje žádné dva vrcholy spojené uzlem.

Nyní je třeba ukázat, že tuto redukci lze implementovat úplným Turingovým strojem, který pracuje v polynomiálním čase. Kontrolu správnosti formátu formule lze provést v čase $\mathcal{O}(n)$. K vytvoření množiny vrcholů je třeba jediný průchod formulí, neboť je třeba spočítat počet klauzulí. To odpovídá složitosti $\mathcal{O}(n)$. Následně je třeba pro každý literál projít všechny klauzule a zjistit, zda mají být se zkoumaným literálem spojeny hranou. To odpovídá složitosti $\mathcal{O}(n^3)$. TS implementující redukční funkci tedy pracuje v čase $\mathcal{O}(n^3)$.

Ukázali jsme, že $SAT_3 \leq INDEPENDENT - SET$, pročež $INDEPENDENT - SET$ je **NP**-těžký problém.

c)

Příslušnost do **NP**:

Uvažujme o vícepáskovém nedeterministickém Turingově stroji M rozhodujícím $KNAPSACK$ v polynomiálním čase, který pracuje následovně. Na své první pásce má stroj zapsaný svůj vstup $\Delta\langle R\rangle\#\langle w\rangle\#\langle v\rangle\#\langle W\rangle\#\langle V\rangle\Delta\Delta^\omega$, kde $\langle R\rangle$ je kód multimnožiny R, $\langle w\rangle$ je kód váhové funkce $w : R \to \mathbb{Z}$, $\langle v\rangle$ je kód cenové funkce $v : R \to \mathbb{Z}$ a $\langle W\rangle, \langle V\rangle$ jsou kódy celých čísel W a V. Stroj nejprve zkontroluje

validitu vstupu, což lze provést v kvadratickém čase $\mathcal{O}(n^2)$. Pokud je vstup nevalidní, M odmítne, jinak pokračuje. Stroj M následně na svou druhou pásku nedeterministicky vygeneruje množinu $R' \subseteq R$. To lze provést v lineárním čase $\mathcal{O}(n)$. Stroj M dále na svou třetí pásku přepíše množinu R' z druhé pásky, přičemž však každý prvek z R' nahradí za odpovídající váhu dle funkce w. To lze provést v kvadratickém čase $\mathcal{O}(n^2)$, neboť je třeba pro každý prvek množiny R' projít dvojice $(r, z) \in w$. Stroj dále na čtvrou pásku přepíše množinu R' z druhé pásky, přičemž však každý prvek z R' nahradí za odpovídající cenu dle funkce v. To lze opět provést v kvadratickém čase $\mathcal{O}(n^2)$. Stroj dále sečte všechny váhy na třetí pásce a výsledek zapíše na pátou pásku. Obdobně sečte veškeré ceny na čtvrté pásce a výsledek zapíše na šestou pásku. To lze provést v lineárním čase $\mathcal{O}(n)$. Stroj poté porovná číslo na páté pásce s hodnotu W. Pokud je toto číslo vyšší, M odmítne, jinak pokračuje. Porovná číslo na šesté pásce s hodnotou V. Pokud je toto číslo nižší, odmítne, jinak akceptuje. Je tedy zřejmé, že tento nedeterministický stroj pracuje v kvadratickém čase $\mathcal{O}(n^2)$, pročež $KNAPSACK \in$ **NP**.

Příslušnost do **EXP**:
Uvažujme o vícepáskovém deterministickém Turingově stroji M rozhodujícím $KNAPSACK$ v exponenciálním čase, který pracuje následovně. Na své první pásce má stroj zapsaný svůj vstup $\Delta\langle R\rangle\#\langle w\rangle\#\langle v\rangle\#\langle W\rangle\#\langle V\rangle\Delta\Delta^\omega$, kde $\langle R\rangle$ je kód multimnožiny R, $\langle w\rangle$ je kód váhové funkce $w : R \to \mathbb{Z}$, $\langle v\rangle$ je kód cenové funkce $v : R \to \mathbb{Z}$ a $\langle W\rangle, \langle V\rangle$ jsou kódy celých čísel W a V. Stroj nejprve zkontroluje validitu vstupu, což lze provést v kvadratickém čase $\mathcal{O}(n^2)$. Pokud je vstup nevalidní, M odmítne, jinak pokračuje.
Stroj M následně na svou druhou pásku generuje jednotlivé podmnožiny $R' \subseteq R$, kterých je 2^R. Generuje je postupně od nejméně mohutné po nejmohutnější množinu, přičemž u stejně mohutných množin se řídí lexikografickým pořadím prvků množiny. Pro každou z těchto množin provede následující operace: Stroj M na svou třetí pásku přepíše množinu R' z druhé pásky, přičemž však každý prvek z R' nahradí za odpovídající váhu dle funkce w. To lze provést v kvadratickém čase $\mathcal{O}(n^2)$, neboť je třeba pro každý prvek množiny R' projít dvojice $(r, z) \in w$. Stroj dále na čtvrou pásku přepíše množinu R' z druhé pásky, přičemž však každý prvek z R' nahradí za odpovídající cenu dle funkce v. To lze opět provést v kvadratickém čase $\mathcal{O}(n^2)$. Stroj dále sečte všechny váhy na třetí pásce a výsledek zapíše na pátou pásku. Obdobně sečte veškeré ceny na čtvrté pásce a výsledek zapíše na šestou pásku. To lze provést v lineárním čase $\mathcal{O}(n)$. Stroj poté porovná číslo na páté pásce s hodnotu W. Pokud je toto číslo vyšší, smaže obsah třetí až šesté pásky a pokračuje další iterací (vygenerováním nové podmnožiny), jinak pokračuje. Porovná číslo na šesté pásce s hodnotou V. Pokud je toto číslo nižší, smaže obsah třetí až šesté pásky a pokračuje další iterací (vygenerováním nové podmnožiny). Poznamenejme, že úklid těchto pásek lze realizovat v lineárním čase $\mathcal{O}(n)$. Pokud je však hodnota na šesté pásce stejně velká nebo vyšší než V, stroj M akceptuje. Pokud stroj neakceptoval a již byly vyčerpány všechny podmnožiny R, odmítne. Je tedy zřejmé, že tento deterministický stroj pracuje v exponenciálním čase $\mathcal{O}(n^2 \cdot 2^n) \subseteq \mathcal{O}(2^{n^2})$, pročež $KNAPSACK \in$ **EXP**.

NP-těžkost:

NP-těžkost dokážeme pomocí polynomiální redukce z problému $SUBSET - SUM$. Ukážeme tedy $SUBSET - SUM \leq_P^m KNAPSACK$. Toto nám bude stačit jako důkaz, neboť $SUBSET - SUM$ je **NP**-úplný. Připomeňme, v čem spočívá problém $SUBSET - SUM$. Instancí tohoto problému je dvojice (S, T), kde S je multimnožina celých čísel (tj. množina povolující duplicity) a T je kýžený součet. Otázkou je, zda existuje $S' \subseteq S$, aby

$$\sum_{s \in S'} s = T.$$

Redukční funkci f definujeme takto:

$$f(\langle S\rangle\#\langle T\rangle) = \langle R\rangle\#\langle w\rangle\#\langle v\rangle\#\langle W\rangle\#\langle V\rangle$$

Pokud $\langle S\rangle\#\langle T\rangle$ není validní instance $SUBSET - SUM$, pak redukční funkce f vrací takový řetězec $\langle R\rangle\#\langle w\rangle\#\langle v\rangle\#\langle W\rangle\#\langle V\rangle$, že $R = \{\}, v = \{\}, w = \{\}, W = 0$ a $V = 1$, tedy určitě neexistuje vhodná podmnožina R, aby součet cen jejích prvků byl roven alespoň 1. Jinak vrací takové kódy, že

- $R = S$
- $w = \{(s, s) \mid s \in S\}$
- $v = \{(s, s) \mid s \in S\}$
- $W = T$
- $V = T$

Funkce v a w jsou totožné a zobrazují celé číslo samo na sebe. Hodnoty W a V jsou taktéž totožné a odpovídají kýženému součtu v instanci problému $SUBSET - SUM$. Z toho plyne, že tato instance $KNAPSACK$ bude přijata pouze tehdy, pokud se cena i váha rovnají T. Vzhledem k definici R, w a v to může nastat pouze tehdy, pokud $\langle S\rangle\#\langle T\rangle \in SUBSET - SUM$. Lze snadno nahlédnout, že tato redukce je tedy správná.

Nyní ukážeme, že redukční funkci lze implementovat pomocí úplného Turingova stroje, který pracuje v polynomiálním čase. Kontrolu správnosti formátu instance $SUBSET - SUM$ lze provést v čase $\mathcal{O}(n)$. K vytvoření R, w a v stačí lineární průchod vstupem, lze to tedy provést v čase $\mathcal{O}(n)$. Volba W a V odpovídá složitosti $\mathcal{O}(n)$. Stroj tedy pracuje v polynomiálním čase $\mathcal{O}(n)$.

Ukázali jsme, že $SUBSET - SUM \leq KNAPSACK$, pročež $KNAPSACK$ je **NP**-těžký problém.

d*)

Příslušnost do **NP**:

Uvažujme o vícepáskovém nedeterministickém Turingově stroji M rozhodujícím $CHROMATIC - NUMBER$ v polynomiálním čase, který pracuje následovně. Na své první pásce má stroj svůj vstup $\Delta\langle G\rangle\#\langle c\rangle\Delta\Delta^\omega$, kde $\langle G\rangle$ je kód grafu $G = (V, E)$ a $\langle c\rangle$ je kód přirozeného čísla c (kde $c \geq 1$), o němž budeme zjišťovat, zda je chromatickým číslem G. Stroj nejprve zkontroluje validitu vstupu, což lze provést v kubickém čase $\mathcal{O}(n^3)$. Pokud je vstup nevalidní, stroj M odmítne, jinak pokračuje. Stroj M následně na svou druhou pásku nedeterministicky vygeneruje funkci $f : V \to \{i \in \mathbb{N} \mid 1 \leq i \leq c\}$, tedy vygeneruje množinu dvojic (v, i), kde $v \in V \wedge i \in \mathbb{N} \wedge 1 \leq i \leq c$ a hodnoty v napříč dvojcemi budou unikátní. Toto lze provést v polynomiálním čase $\mathcal{O}(n^3)$. Stroj následně překopíruje obsah této druhé pásky na třetí v čase $\mathcal{O}(n)$. Stroj M bude následně procházet od začátku druhou pásku. Pro každý prvek (v, i) projde množinu E a zjistí, zda existuje nějaký vrchol v', pro který platí, že $\{v, v'\} \in E$. Pokud takový vrchol najde, projde celou třetí pásku a zjistí, na jaké číslo se zobrazuje vrchol v'. Pokud se ukáže, že $f(v) = f(v')$, stroj M odmítne, jinak pokračuje. Takto projde celou druhou pásku v čase $\mathcal{O}(n^4)$. Pokud stroj M během tohoto procesu neodmítl a vyčerpal všechny dvojice z druhé pásky, akceptuje. Je tedy zřejmé, že tento nedeterministický Turingův stroj pracuje v bikvadratickém čase $\mathcal{O}(n^4)$, pročež $CHROMATIC - NUMBER \in$ **NP**.

Příslušnost do **EXP**:

Uvažujme o vícepáskovém deterministickém Turingově stroji M rozhodujícím $CHROMATIC - NUMBER$ v exponenciálním čase, který pracuje následovně. Na své první pásce má stroj svůj vstup $\Delta\langle G\rangle\#\langle c\rangle\Delta\Delta^\omega$, kde $\langle G\rangle$ je kód grafu $G = (V, E)$ a $\langle c\rangle$ je kód přirozeného čísla c (kde $c \geq 1$), o němž budeme zjišťovat, zda je chromatickým číslem G. Stroj nejprve zkontroluje validitu vstupu, což lze provést

v kubickém čase $\mathcal{O}(n^3)$. Pokud je vstup nevalidní, stroj M odmítne, jinak pokračuje. Stroj poté bude na svou druhou pásku postupně generovat všechny funkce $f : V \to \{i \in \mathbb{N} \mid 1 \leq i \leq c\}$, tedy v každé iteraci vygeneruje množinu dvojic (v, i), kde $v \in V \wedge i \in \mathbb{N} \wedge 1 \leq i \leq c$ a hodnoty v napříč dvojcemi budou unikátní. Generovat bude v lexikografickém pořadí oboru hodnot f vzhledem k definičnímu oboru f, tj. začne funkcí $f_1 = \{(v_1, 1), (v_2, 1), \ldots, (v_{|V|}, 1)\}$, bude pokračovat funkcí $f_2 = \{(v_1, 1), (v_2, 1), \ldots, (v_{|V|}, 2)\}$ a bude pokračovat až k funkci $f_{c^{|V|}} = \{(v_1, c), (v_2, c), \ldots, (v_{|V|}, c)\}$. Lze snadno nahlédnout, že takových funkcí je $c^{|V|}$, tedy exponenciální počet vzhledem k délce vstupu. Každou takovou funkci si stroj M překopíruje na svou třetí pásku v čase $\mathcal{O}(n)$. Stroj M bude následně od začátku procházet druhou pásku. Pro každý prvek (v, i) projde množinu E a zjistí, zda existuje nějaký vrchol v', pro který platí, že $\{v, v'\} \in E$. Pokud se ukáže, že $f(v) = f(v')$, stroj pokračuje další iterací (vygeneruje nové zobrazení), jinak pokračuje. Takto projde celou druhou pásku v čase $\mathcal{O}(n^4)$. Pokud stroj M během tohoto procesu nepřeskočil na další iteraci a vyčerpal všechny dvojice z druhé pásky, akceptuje. Pokud stroj M vyčerpal veškeré funkce f, aniž by přijal, odmítne. Je tedy zřejmé, že tento deterministický Turingův stroj pracuje v exponenciálním čase $\mathcal{O}(n^4 \cdot n^n) \subseteq \mathcal{O}(2^{n^2})$, pročež $CHROMATIC - NUMBER \in$ **EXP**.

NP-těžkost:

Naprosto netuším.

e)

Příslušnost do **NP**:

Uvažujme o vícepáskovém nedeterministickém Turingově stroji M rozhodujícím $CLIQUE - COVER$ v polynomiálním čase, který pracuje následovně. Na první pásce má stroj svůj vstup $\Delta\langle G\rangle\#\langle m\rangle\Delta\Delta^\omega$, kde $\langle G\rangle$ je kód grafu $G = (V, E)$ a $\langle m\rangle$ je kód přirozeného čísla m. Stroj nejprve na svou druhou pásku nedeterministicky vygeneruje systém množin S, pro který platí

$$\bigcup_{A \in S} A = V.$$

Tento systém je také rozkladem množiny V, tedy kromě výše uvedené vlastnosti pro něj platí

$$\forall A, B \in S : A \cap B \neq \{\} \Rightarrow A = B,$$

tedy každá dvojice množin v systému S je buď disjunktní, nebo obsahuje tytéž prvky. Lze snadno nahlédnout, že pokud nějaký rozklad množiny V, který vznikne sjednocením některých tříd systému S, tvoří klikové pokrytí G, pak i S tvoří klikové pokrytí G. Jelikož hledáme klikové pokrytí velikosti m nebo nižší, postačí nám omezit se na rozklady množiny V o velikosti právě m. Nedeterminitické vygenerování totoho systému lze realizovat v čase $\mathcal{O}(n)$.

Stroj M nejprve v čase $\mathcal{O}(n)$ ověří, zda je jednotlivých tříd rozkladu m nebo méně. Pokud ne, odmítne, jinak pokračuje. Stroj následně prochází jednotlivé třídy rozkladu V. V každé této množině ověří, zda je každá dvojice různých vrcholů spojena hranou. Nalezení všech dvojic v třídě rozkladu má složitost $\mathcal{O}(n^2)$. Ověření, že dvojice má společnou hranu, lze realizovat v čase $\mathcal{O}(n)$. Porovnání probíhá v čase $\mathcal{O}(n)$. Pokud se ukáže, že některá dvojice vrcholů ve stejné třídě rozkladu není spojena hranou, stroj M odmítne. Pokud stroj projde všechny třídy rozkladu a zjistí, že veškeré dvojice různých vrcholů ve všech třídách jsou spojeny hranou, akceptuje. Nedeterministický Turingův stroj M pracuje v polynomiálním čase $\mathcal{O}(n^4)$, pročež $CLIQUE - COVER \in$ **NP**.

Příslušnost do **EXP**:

Abychom mohli ukázat, že $CLICQUE - COVER \in$ **EXP**, je vhodné si uvědomit, co platí pro klikové pokrytí grafu. Mějme graf $G = (V, E)$. Je zřejmé, že tento graf má klikové pokrytí o velikosti $|V|$, neboť každý vrchol sám o sobě tvoří 1-úplný podgraf G, tj. kliku o velikosti 1. Pokud se nám podaří najít v grafu klikové pokrytí o velikosti $k < |V|$, pak to určitě znamená, že G má i kliková pokrytí o všech velikostech z množiny $\{i \in \mathbb{N} \mid k \leq i \leq |V|\}$, neboť každou kliku o velikosti $l > 1$ můžeme rozdělit na dvě kliky o velikostech $l - 1$ a 1.

Z toho plyne, že pokud rozhodujeme o existenci klikového pokrytí o maximální velikosti m, stačí nám zaměřit se na zkoumání klikového pokrytí o velikosti právě m, neboť existence klikového pokrytí o menší velikosti by existenci klikového pokrytí o velikosti m automaticky implikovala (pokud $m \leq |V|$).

Uvažujme tedy o vícepáskovém deterministickém Turingově stroji M rozhodujícím $CLIQUE - COVER$ v exponenciálním čase, který pracuje následovně. Na první pásce má stroj svůj vstup $\Delta\langle G\rangle\#\langle m\rangle\Delta\Delta^{\omega}$, kde $\langle G\rangle$ je kód grafu $G = (V, E)$ a $\langle m\rangle$ je kód přirozeného čísla m. Pokud $m > |V|$, stroj ihned odmítne. Stroj na svou druhou pásku poté postupně generuje jednotlivé rozklady množiny V o mohutnosti m. Postupuje tak, že dle lexikografického pořadí názvů uzlů nejprve dává nejmenší uzly do nejmohutnějších tříd rozkladu. Různých rozkladů množiny V do m tříd je

$$S\left\{ {|V| \atop m} \right\}$$

což je Stirlingovo číslo druhého druhu pro množinu o mohutnosti $|V|$ a m tříd rozkladu. Je známo, že Stirlingova čísla druhého druhu pro $2 \leq |V|$ a $1 \leq m \leq |V| - 1$ lze shora ohraničit následujícím způsobem:

$$S\left\{ {|V| \atop m} \right\} \leq \frac{1}{2}\binom{|V|}{m} \cdot m^{|V|-m}$$

tedy pro vstup o délce n jistě není více než $\mathcal{O}(2^{n^2})$ rozkladů vstupní množiny.

Pro každý vygenerovaný rozklad V spustí stroj M následující proces. Stroj postupně prochází jednotlivé třídy rozkladu. V každé z nich postupně projde všechny dvojice různých vrcholů grafu a průchodem množiny E ověří, zda jsou tyto vrcholy spojeny hranou. Pokud ne, stroj M se přesune k další iteraci (vygeneruje nový rozklad V). Pokud stroj zjistí, že každá třída rozkladu tvoří kliku, akceptuje. Je zřejmé, že tento proces má pro jeden rozklad složitost $\mathcal{O}(n^4)$ a generování rozkladů V má exponenciální složitost. Pokud stroj M vyčerpá veškeré rozklady V o mohutnosti m, aniž by přijal, odmítne. Deterministický Turingův stroj M tedy rozhoduje $CLIQUE - COVER$ v exponenciálním čase, pročež $CLIQUE - COVER \in$ **EXP**.

NP-těžkost:

NP-těžkost dokážeme pomocí polynomiální redukce z problému $CHROMATIC - NUMBER$. Ukážeme tedy $CHROMATIC - NUMBER \leq_P^m CLICQUE - COVER$. Toto nám bude stačit jako důkaz, neboť $CHROMATIC - NUMBER$ je **NP**-úplný. Redukční funkci definujeme takto:

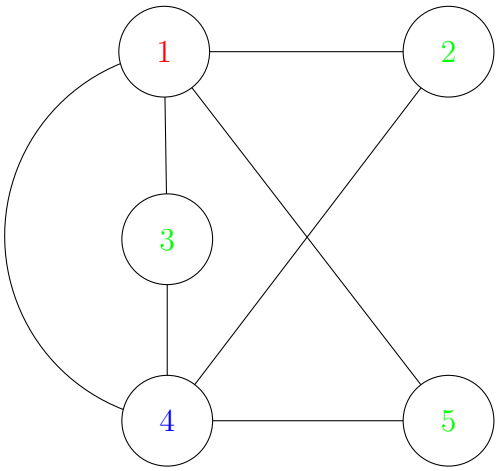
$$f(\langle G\rangle\#\langle c\rangle) = \langle G'\rangle\#\langle m\rangle$$

kde $\langle G'\rangle$ je kód grafu $G' = (V', E')$ a $\langle m\rangle$ je kód přirozeného čísla m. Pokud $\langle G\rangle\#\langle c\rangle$ není korektní instance $CHROMATIC - NUMBER$, vrátí redukční funkce f takové kódy $\langle G'\rangle$ a $\langle m\rangle$, že $G' = (\{\}, \{\})$ a $m = 1$, neboť u prázdného grafu jistě neexistuje klikové pokrytí velikosti 1. Jinak vrací kód grafu $G' = (V', E')$ a kód přirozeného čísla m, pro něž platí:

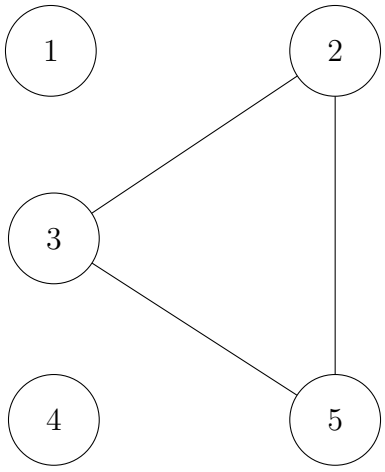
- $V' = V$
- $E' = \{\{i, j\} \in V \times V \mid \{i, j\} \notin E \wedge i \neq j\}$

- $m = \min\{c, |V|\}$

Pokud $c \geq |V|$, pak je jasné, že graf G jistě má chromatické číslo c, neboť vrcholy obarvujeme vyšším počtem barev, než je počet samotných vrcholů. Pokud ale $m \geq |V|$, pak graf G' určitě nemá klikové pokrytí o velikosti m tvořené po dvojicích disjunktními třídami rozkladu, protože byl překročen počet uzlů. Proto volíme $m = \min\{c, |V|\}$, aby se v případě překročení počtu uzlů napevno vybralo klikové pokrytí o velikosti $|V|$, které vždy existuje, neboť každý úzel z V je 1-úplným podgrafem G'.
Nerofmálně řečeno bude mít graf G' invertované všechny hrany oproti grafu G, přitom však vyloučíme reflexivní hrany. Tento postup jsme zvolili proto, že tak jistě ze všech stejně obarvených uzlů vzniknou úplné podgrafy, neboť tyto vrcholy nemohly být v grafu G spojeny hranami. Pokud tedy graf G má chromatické číslo $c \leq |V|$, pak graf G' obsahuje klikové pokrytí velikosti c. Pokud graf G chromatické číslo $c \leq |V|$ nemá, pak graf G' neobsahuje klikové pokrytí velikosti c. Demonstrujme si to na příkladu. Mějme následující graf:



Je zřejmé, že tento graf má chromatické číslo 3, neboť jednotlivé uzly lze obarvit tak, jak je demonstrováno výše. Tento graf však již nemá chromatické číslo 2, neboť se dvěma barvami si nevystačíme. Redukční funkce by vrátila následující graf:



Je zřejmé, že tento graf má klikové pokrytí o velikosti 3, neboť lze vytvořit rozklad $\{\{1\}, \{2, 3, 5\}, \{4\}\}$, přičemž každá třída rozkladu tvoří úplný podgraf. Klikové pokrytí o velikosti 2 však již nejsme schopni vytvořit.
Nyní ukážeme, že redukční funkci jsme schopni implementovat pomocí úplného Turingova stroje, který pracuje v polynomiálním čase. Kontrolu správnosti instance $CHROMATIC - NUMBER$ lze provést v

čase $\mathcal{O}(n^3)$. K vytvoření množiny V' stačí lineární průchod vstupem, tedy lze provést v čase $\mathcal{O}(n)$. Množinu E' zkonstruujeme v bikvadratickém čase $\mathcal{O}(n^4)$, neboť pro každou dvojici čísel i a j, kde $i \neq j$, je třeba provést lineární průchod množinou E a zjistit, zda se tam tato dvojice nachází, či nikoliv. Hodnotu m jsme schopni určit v polynomiálním čase $\mathcal{O}(n)$, neboť je třeba pouze zjistit mohutnost množiny V. Stroj tedy pracuje v bikvadratickém čase $\mathcal{O}(n^4)$.

Ukázali jsme, že $CHROMATIC-NUMBER \leq_P^m CLIQUE-COVER$, pročež $CLIQUE-COVER$ je **NP**-těžký problém.