

## Týden 4

### Kanonické (deterministické) konečné automaty

Připomněli jsme si:

**Věta.** Jazyk  $L \subseteq \Sigma^*$  je regulární (tzn. přijímaný konečným automatem) právě tehdy, když je množina kvocientů  $\{w \setminus L \mid w \in \Sigma^*\}$  konečná.

Načrtli jsme si důkaz:

“ $\Rightarrow$ ”: Nechtě  $L \subseteq \Sigma^*$  je regulární, tedy  $L = L(A)$  pro jistý KA  $A = (Q, \Sigma, \delta, q_0, F)$ . Snadno ověříme, že  $q_0 \xrightarrow{w} q$  implikuje  $w \setminus L = L_q$ , když definujeme  $L_q = \{u \in \Sigma^* \mid q \xrightarrow{u} F\}$ . Množina  $\{w \setminus L \mid w \in \Sigma^*\}$  má tedy nejvýše tolik prvků, kolik má  $A$  stavů.

“ $\Leftarrow$ ”: Nechtě  $\{w \setminus L \mid w \in \Sigma^*\} = \{L_0, L_1, \dots, L_k\}$  (pro  $k \in \mathbb{N}$ ); nechtě přitom  $L_0 = \varepsilon \setminus L = L$ . Uvažujme KA  $A = (Q, \Sigma, \delta, L_0, F)$ , kde  $Q = \{L_0, L_1, \dots, L_k\}$ ,  $F = \{L_i \mid \varepsilon \in L_i\}$  a  $\delta(L_i, a) = a \setminus L_i$ . (Jelikož  $a \setminus (w \setminus L) = wa \setminus L$ , je  $\delta : Q \times \Sigma \rightarrow Q$  dobře definována.) Indukcí podle délky slova snadno ověříme, že v automatu  $A$  máme  $L_0 \xrightarrow{w} w \setminus L$  a také  $L(A) = \{w \mid L_0 \xrightarrow{w} F\} = \{w \mid w \in L\} = L$  (neboť  $\varepsilon \in w \setminus L \Leftrightarrow w \in L$ ).

Automat popsáný v části “ $\Leftarrow$ ” je *kanonickým automatem* pro jazyk  $L$ ; je vidět, že je to nejmenší konečný automat, který  $L$  přijímá. Navíc je takový nejmenší automat jednoznačně určen až na izomorfismus, tedy až na pojmenování stavů. Z úvah uvedených níže plyne algoritmická konstrukce kanonických automatů.

### Redukce konečného automatu

Podívali jsme se na následující automat (který by vznikl po dotažení modulární konstrukce z první přednášky: má  $3 \cdot 4 = 12$  stavů, jež vznikly přejmenováním původních dvojic).

	0	1
$\leftrightarrow s_1$	$s_1$	$s_2$
$s_2$	$s_3$	$s_4$
$s_3$	$s_5$	$s_6$
$\leftarrow s_4$	$s_7$	$s_2$
$s_5$	$s_8$	$s_4$
$s_6$	$s_9$	$s_6$
$\leftarrow s_7$	$s_1$	$s_9$
$s_8$	$s_5$	$s_{10}$
$s_9$	$s_6$	$s_{11}$
$s_{10}$	$s_{12}$	$s_{10}$
$s_{11}$	$s_{11}$	$s_9$
$s_{12}$	$s_8$	$s_{11}$

Rychle jsme zjistili, že všechny stavy jsou dosažitelné a promýšleli jsme, jak zjistit, zda existují dva různé stavy  $q, q' \in Q = \{s_1, s_2, \dots, s_{12}\}$  splňující  $L_q = L_{q'}$ . Jde tedy o zjištění,

zda pro  $q, q'$  existuje (či neexistuje) *rozlišující slovo*, tj. slovo  $w$ , pro něž platí  $q \xrightarrow{w} F$  a  $\neg(q \xrightarrow{w} F)$  či naopak  $\neg(q \xrightarrow{w} F)$  a  $q \xrightarrow{w} F$ .

Uvědomili jsme si, že se nabízí induktivní postup, při němž konstruujeme ekvivalence  $\sim_0, \sim_1, \sim_2, \dots$  na množině  $Q$  definované takto:

$$q \sim_i q' \Leftrightarrow \text{pro stavy } q, q' \text{ neexistuje rozlišující slovo délky } \leq i.$$

Každé ekvivalenci  $\sim_i$  odpovídá příslušný rozklad  $R_i$  na množině  $Q$ .

Výchozí  $R_0$  jsme hravě sestrojili; má dvě třídy, přijímající a nepřijímající stavy.

$$R_0: \quad \text{I} = \{s_1, s_4, s_7\}, \quad \text{II} = \{s_2, s_3, s_5, s_6, s_8, s_9, s_{10}, s_{11}, s_{12}\}$$

Pak jsme začali sestrojovat  $R_1$  (postupem popsáním ve studijním textu).

$$R_1: \quad \text{I} = \{s_1, s_4, s_7\}, \quad \text{II} = \{s_2, s_5\}, \quad \text{III} = \{s_3, s_6, s_8, s_9, s_{10}, s_{11}, s_{12}\}.$$

Uvědomili jsme si, že postup se opírá o toto pozorování:

pokud pro  $q, q'$  existuje rozlišující slovo délky  $i + 1$ , tak nutně existuje  $b \in \Sigma$  (v našem případě  $\Sigma = \{0, 1\}$ ) tak, že  $q \xrightarrow{b} q'', q' \xrightarrow{b} q'''$  a pro  $q'', q'''$  existuje rozlišující slovo délky  $i$ .

(Pozn.: použili jsme symbol  $b$  pro proměnnou, ať si uvědomíme, že nemusíme vždy stereotypně používat  $a$ .)

Kdybychom pokračovali, zjistili bychom, že

$$R_2: \quad \text{I} = \{s_1, s_4\}, \quad \text{II} = \{s_7\}, \quad \text{III} = \{s_2, s_5\}, \quad \text{IV} = \{s_3, s_8\}, \quad \text{V} = \{s_6, s_9, s_{10}, s_{11}, s_{12}\},$$

atd., až bychom dospěli k pevnému bodu, tedy k rozkladu, který se již nezjemní. Ten má 3-prvkovou třídu  $\{s_6, s_9, s_{11}\}$  a pak už jen jednoprvkové třídy.

Redukcí (původně 12-stavového) automatu tedy vznikne ekvivalentní automat s 10 stavy (jeho stavy odpovídají třídám závěrečného rozkladu).

(Pozn. Na přednášce jsme postup demonstrovali na příkladu uvedeném na slidech.)

## Ekvivalence konečných automatů; minimální automaty

Přemýšleli jsme, zda bychom uměli navrhnout (a naprogramovat) algoritmus řešící následující problém.

NÁZEV: *Ekvivalence konečných automatů*

VSTUP: dva konečné automaty  $A_1, A_2$

VÝSTUP: ANO – jestliže  $L(A_1) = L(A_2)$ ,

NE – jestliže  $L(A_1) \neq L(A_2)$ .

K jednomu možnému algoritmu nás přímo přivedla myšlenka, že u negativního případu, tedy  $L(A_1) \neq L(A_2)$ , by bylo dobré také ukázat protipříklad, tedy (co nejkratší) slovo  $w$ , které patří jen do jednoho z jazyků  $L(A_1), L(A_2)$ . Uvědomili jsme si, že takové slovo patří do jazyka

$$L = (L(A_1) - L(A_2)) \cup (L(A_2) - L(A_1)) = (L(A_1) \cap \overline{L(A_2)}) \cup (L(A_2) \cap \overline{L(A_1)})$$

a že díky dříve probraným algoritmům snadno sestrojíme  $A$  tak, že  $L(A) = L$ . Je nám tedy jasný algoritmus, který k zadaným  $A_1, A_2$  sestrojí  $A$  tak, že

$$L(A_1) = L(A_2) \Leftrightarrow L(A) = \emptyset.$$

No a napsat proceduru (algoritmus), která o zadaném konečném automatu  $A$  zjistí, zda  $L(A) = \emptyset$ , hravě zvládneme (když si vzpomeneme na algoritmus pro zjišťování dosažitelných stavů; stačí prostě zjistit, zda nějaký přijímající stav je dosažitelný [z počátečního]). Ekvivalence konečných automatů se ovšem dá algoritmicky (rychle) zjišťovat i jinak. Nejprve jsme si uvedli následující přirozenou definici.

Konečný automat  $A$  je *minimální*, jestliže neexistuje automat  $A'$ , který je ekvivalentní s  $A$  (pro nějž je tedy  $L(A) = L(A')$ ) a který má méně stavů než  $A$ .

Odvodili jsme si, že platí:

*Věta.* Následující podmínky pro konečný automat  $A$  jsou ekvivalentní:

1.  $A$  je minimální.
2.  $A$  je kanonický (pro jazyk  $L(A)$ ).
3.  $A$  je redukovaný, tj. nemá nedosažitelné stavy a nemá různé stavy  $q, q'$  splňující  $L_q = L_{q'}$ .

Uvědomili jsme si, že algoritmus rozhodující ekvivalenci konečných automatů může být tedy založen na redukci (odstranění nedosažitelných stavů a ztotožnění stavů se stejným  $L_q$ ) a na převodu do *normovaného tvaru*. (Dva automaty jsou ekvivalentní právě tehdy, když po redukci mají stejný normovaný tvar.)

## Bezkontextové gramatiky

Připomeňme, že

$$(((a \cdot a) + b)^*)$$

je příklad (úplně uzávorkovaného) regulárního výrazu, který reprezentuje jazyk v abecedě  $\{a, b\}$ . Obecný regulární výraz nad abecedou  $\{a, b\}$  je prostě řetězcem symbolů abecedy

$$\Sigma = \{a, b, +, \cdot, *, (, )\}.$$

(Pro úplnost bychom měli přidat znaky  $\emptyset, \epsilon$ , ale zde se bez nich obejdeme.)

Ne každý řetězec v  $\Sigma^*$  je ovšem regulárním výrazem; např. řetězec “ $a$ ” $++$ “ regulárním výrazem není.

Snadno jsme vyvodili, že množina těchto regulárních výrazů, označovaná  $RV(\{a, b\})$ , není regulárním jazykem (tedy není rozpoznatelná konečným automatem). Dá se ale generovat bezkontextovou gramatikou, např.

$$R \longrightarrow a \mid b \mid (R + R) \mid (R \cdot R) \mid (R^*).$$

Demonstrovali jsme si základní pojmy teorie bezkontextových gramatik. Ukázali jsme si (*levou derivaci* slova  $((a \cdot a) + b)^*$ , příslušný *derivační strom*, apod.

Připomněli jsme definici *bezkontextové gramatiky* jako struktury

$$G = (\Pi, \Sigma, S, P)$$

a *jazyka generovaného gramatikou*

$$L(G) = \{w \in \Sigma^* \mid S \Rightarrow^* w\}.$$

Pak jsme se vrátili k jazyku  $RV(\{a, b\})$  a upravili jej tak, že v příslušných řetězcích (regulárních výrazech) můžeme vynechávat tečku pro zřetězení a některé závorky podle naší výše uvedené dohody. Např. výraz  $((a \cdot a) + b)^*$  můžeme zjednodušit na  $(aa + b)^*$ . Takto upravený jazyk regulárních výrazů generuje např. gramatika

$$R \longrightarrow a \mid b \mid R + R \mid R \cdot R \mid RR \mid R^* \mid (R).$$

Všimli jsme si ovšem, že např. slovo  $aa + b$  má v této gramatice dva různé derivační stromy; tedy tato gramatika *není jednoznačná*. Příčinou je fakt, že naše dohodnutá priorita operátorů není v uvedené gramatice reflektována.

Po jistém zamyšlení se nám podařilo navrhnout ekvivalentní gramatiku (tedy generující tentýž jazyk), která jednoznačná je; konkrétně šlo o následující gramatiku

$$\begin{aligned} R &\longrightarrow T + R \mid T \\ T &\longrightarrow FT \mid F \\ F &\longrightarrow F^* \mid (R) \mid C \\ C &\longrightarrow a \mid b \end{aligned}$$

Uvedli jsme pojem (*vnitřně*) *jednoznačný bezkontextový jazyk* a několik souvisejících poznámek (viz slidy).

## Partie textu k prostudování

V návaznosti na část 3.3. (modulární návrh) se jedná zejména o část 3.4. (dosažitelné stavy, normovaný tvar), část 3.6. (minimalizace konečných automatů) a část 3.7. (ekvivalence konečných automatů, minimální automaty). Dále se jedná o části 4.1., 4.2., 4.3. (bezkontextové gramatiky, jednoznačné gramatiky).

(Máte si udělat přinejmenším dobrou první představu a zamyslet se nad příklady, speciálně těmi plánovanými na cvičení, ať se můžete na cvičení aktivně účastnit a případné problémy si tam objasnit.)

## Cvičení

První dva příklady doplňují předchozí týden (a na některých cvičeních již byly probrány).

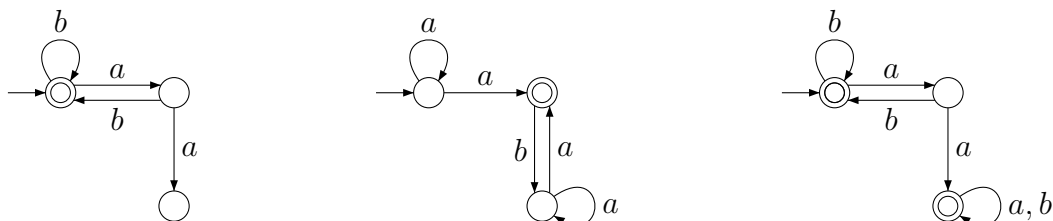
### Příklad 4.1

Následující jazyky zadejte regulárními výrazy.

- $L_1 = \{01, 011, 0101, 0111\}$
- $L_2 = \{w \in \{a, b, c\}^* \mid w \text{ obsahuje podslovo } abb\}$
- $L_3 = \{w \in \{a, b, c\}^* \mid w \text{ začíná prefixem } abc \text{ nebo končí sufixem } bcac\}$
- $L_4 = \{w \in \{0, 1\}^* \mid |w|_0 \bmod 2 = 0\}$ .
- $L_5 = \{w \in \{0, 1\}^* \mid |w|_0 \bmod 3 = 1\}$ .
- $L_6 = \{w \in \{a, b\}^* \mid w \text{ obsahuje podslovo } bab \text{ nebo } |w|_b \leq 3\}$
- $L_7 = \{w \in \{a, b\}^* \mid w \text{ obsahuje podslovo } bab \text{ a } |w|_b \leq 3\}$
- $L_8 = \{w \in \{a, b\}^* \mid w \text{ neobsahuje podslovo } ab\}$

### Příklad 4.2

Jazyky přijímané následujícími konečnými automaty zadejte regulárními výrazy.



**Příklad 4.3**

V příkladu 2.1. jste měli zkonstruovat automat pro jazyk

$L = \{w \in \{a, b\}^* \mid w \text{ začíná } a \text{ a končí } b \text{ nebo začíná } b \text{ a končí } a\}$ .

Připomeňte si tento (5-stavový?) automat. Měli jste konstruovat jen dosažitelné stavy. Zkuste teď nalézt pro každou dvojici stavů  $q \neq q'$  slovo  $w$  tak, že  $q \xrightarrow{w} F$  a  $q' \not\xrightarrow{w} F$  (tedy  $q' \xrightarrow{w} (Q-F)$ ) či naopak  $q \not\xrightarrow{w} F$  a  $q' \xrightarrow{w} F$ ; takovému slovu  $w$  budeme říkat

*rozlišující slovo pro stavy  $q, q'$ .*

Povedlo se? Jestli ano, co z toho o vašem automatu vyvodíte?

**Příklad 4.4**

Proveďte kompletně algoritmus redukce pro automat z přednášky

	0	1
$\leftrightarrow s_1$	$s_1$	$s_2$
$s_2$	$s_3$	$s_4$
$s_3$	$s_5$	$s_6$
$\leftarrow s_4$	$s_7$	$s_2$
$s_5$	$s_8$	$s_4$
$s_6$	$s_9$	$s_6$
$\leftarrow s_7$	$s_1$	$s_9$
$s_8$	$s_5$	$s_{10}$
$s_9$	$s_6$	$s_{11}$
$s_{10}$	$s_{12}$	$s_{10}$
$s_{11}$	$s_{11}$	$s_9$
$s_{12}$	$s_8$	$s_{11}$

**Příklad 4.5**

Nalezněte všechny dvojice stavů  $q, q'$ , pro něž platí  $q \sim q'$ , tedy  $L_q = L_{q'}$ .

	a	b
$\rightarrow 0$	0	1
$\leftarrow 1$	1	2
$\leftarrow 2$	3	1
3	2	4
4	2	3

	a	b
$\rightarrow 5$	5	6
6	7	5
$\leftarrow 7$	7	9
8	9	8
$\leftarrow 9$	8	7

**Příklad 4.6**

Připomeňme problém

NÁZEV: *Ekvivalence konečných automatů*

VSTUP: dva konečné automaty  $A_1, A_2$

VÝSTUP: ANO – jestliže  $L(A_1) = L(A_2)$ ,  
NE – jestliže  $L(A_1) \neq L(A_2)$ .

a uvažujme následující algoritmus pro jeho řešení:

generuj postupně (všechna) slova  $w_0, w_1, w_2, \dots$  v abecedě daných automatů;  
pro každé  $w_i$  přitom zjisti, zda je oběma automaty přijímáno či oběma nepřijímáno – když je jedním přijímáno a druhým nepřijímáno, skonči s výsledkem  $L(A_1) \neq L(A_2)$ .

Je zřejmé, že běh tohoto algoritmu neskončí v případě  $L(A_1) = L(A_2)$ . Lze tento nedostatek opravit tím, že necháme algoritmus probírat jen slova do délky  $n$ , kde  $n$  je součtem počtu stavů  $A_1$  a  $A_2$ , a nenajde-li se rozlišující slovo do té doby, pak algoritmus zahlásí  $L(A_1) = L(A_2)$ ? Pokud je tomu tak (tedy odpověď algoritmu je vždy správná), v čem je tento algoritmus zřejmě horší než algoritmy probrané na přednášce?

**Příklad 4.7**

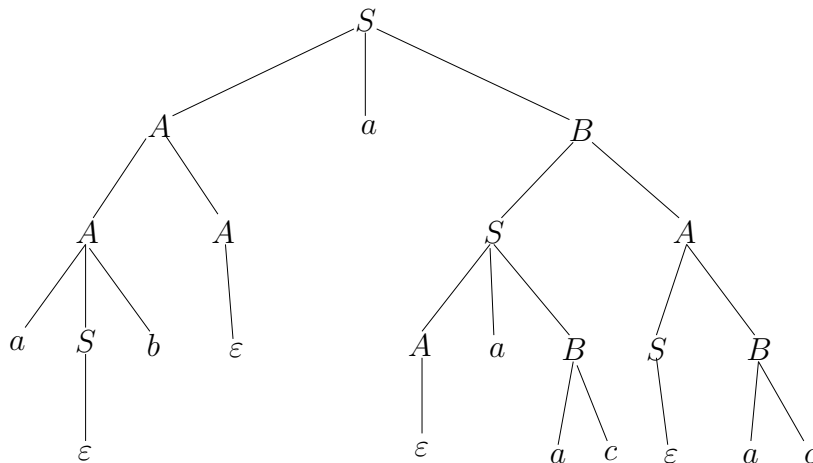
Při konstrukci ekvivalence  $\sim \subseteq Q \times Q$  definované vztahem  $q \sim q' \Leftrightarrow L_q = L_{q'}$  (pro automat  $A = (Q, \Sigma, \delta, q_0, F)$ , přičemž  $L_q = \{w \in \Sigma^* \mid q \xrightarrow{w} F\}$ ), jsme de facto postupovali *koinduktivně*: vzali jsme (největší) relaci  $R_0 = Q \times Q$ , na ni jsme aplikovali určitý monotónní funkcionál  $\mathcal{F}$ , dostali jsme tedy (menší) relaci  $R_1 = \mathcal{F}(R_0)$ , pak jsme dostali relaci  $R_2 = \mathcal{F}(R_1)$ , atd., až jsme dospěli k (největšímu) pevnému bodu  $R$  (pro nějž  $R = \mathcal{F}(R)$ ). (Monotónností funkcionálu  $\mathcal{F} : 2^{Q \times Q} \rightarrow 2^{Q \times Q}$  zde rozumíme vlastnost, že pro  $T_1 \subseteq T_2$  máme  $\mathcal{F}(T_1) \subseteq \mathcal{F}(T_2)$ .) Snažte se funkcionál  $\mathcal{F}$  přesně popsat.

**Příklad 4.8**

Odhadněte časovou složitost algoritmu redukce konečného (deterministického) automatu. Přinejmenším argumentujte, proč je algoritmus polynomiální.

**Příklad 4.9**

Na obrázku je derivační strom pro slovo  $w = abaaacac$  odpovídající jisté bezkontextové gramatice  $G$ .



- Vypište všechna pravidla  $G$ , jejichž existenci můžete vyvodit z daného derivačního stromu.
- Napište levé odvození (levou derivaci) slova  $w$  podle gramatiky  $G$ .
- Najděte *menší* derivační strom pro slovo  $abaaacac$  a zakreslete jej tak, že všechny listy budou na stejné úrovni (tedy odvozené slovo bude celé na „jednom řádku“).
- Najděte nejlevější větev (v onom menším stromě), která obsahuje dva výskyty neterminálu  $B$ . Využijte to k důkazu, že gramatika generuje také slovo  $abaac$ . Pak ukažte, že gramatika také generuje slova  $aba(a)ac(ac)$ ,  $aba(a)^2ac(ac)^2$ ,  $aba(a)^3ac(ac)^3$ ,  $\dots$
- Lze z dostupné informace zjistit něco ohledně jednoznačnosti gramatiky  $G$  ?

#### Příklad 4.10

Jaký jazyk generuje následující gramatika ? Porovnejte s gramatikou na přednášce. Je gramatika jednoznačná ?

$$\begin{aligned}
 R &\longrightarrow L \mid (RBR) \mid (RU) \\
 L &\longrightarrow a \mid b \\
 B &\longrightarrow + \mid \cdot \\
 U &\longrightarrow *
 \end{aligned}$$

#### Příklad 4.11

Připomeňte si následující gramatiku z přednášky.

$$\begin{aligned}
 R &\longrightarrow T + R \mid T \\
 T &\longrightarrow FT \mid F \\
 F &\longrightarrow F^* \mid (R) \mid C \\
 C &\longrightarrow a \mid b
 \end{aligned}$$



Zvolte si nějaký (malý) regulární výraz nad abecedou  $\{a, b\}$ . Nakreslete nejdříve jeho syntaktický strom a poté derivační strom pro zvolený výraz podle uvedené gramatiky. (Vidíte nějaký vztah mezi oběma stromy?)

Pak alespoň neformálně argumentujte, proč je gramatika jednoznačná. (Nápověda.  $T$  (Term) reprezentuje ty regulární výrazy, které nejsou ve tvaru  $R_1 + R_2$  (pro dva regulární výrazy  $R_1, R_2$ ), a  $F$  (Factor) reprezentuje ty výrazy, které nejsou ve tvaru  $R_1 + R_2$  ani  $R_1R_2$ .)