BRNO UNIVERSITY OF TECHNOLOGY FACULTY OF INFORMATION TECHNOLOGY

Teoretická informatika
2022/2023

Domácí úloha 1

Pavel Šesták (xsesta07)

Brno, 16. října 2022

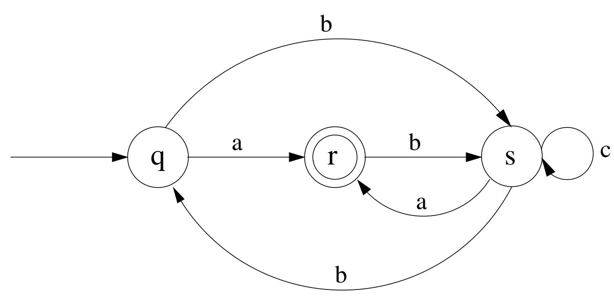
Obsah

Seznam obrázků

Seznam tabulek

Teoretická informatika (TIN) – 2022/2023
Úkol 1
(max. zisk 5 bodů – 10 bodů níže odpovídá 1 bodu v hodnocení předmětu)

1. Uvažte NKA M_3 nad abecedou $\Sigma = \{a, b, c\}$ z obrázku 1:



Obrázek 1: NKA M_3

   (a) Řešením rovnic nad regulárními výrazy sestavte k tomuto automatu ekvivalentní regulární výraz.

   (b) Sestrojte relaci pravé kongruence $\sim$ s konečným indexem takovou, že $L(M_3)$ je sjednocením některých tříd rozkladu $\{a, b, c\}^*/_\sim$.

15 bodů

2. Mějme jazyk L_1 nad abecedou $\{a, b, c\}$ definovaný následovně:

$$L_1 = \{w \mid w \in \{a, b, c\}^* \wedge \#_a(w) > \#_b(w)\}$$

   Sestrojte jazyk L_2 takový, že $L_1 \cap L_2 \in \mathcal{L}_3$ a zároveň $L_1 \cup L_2 \in \mathcal{L}_3$. Dále rozhodněte a dokažte, zda L_2 je, či není, regulární. Pro důkaz regularity sestrojte příslušný konečný automat, nebo gramatiku. Pro důkaz neregularity použijte pumping lemma.

   Dokažte, že jazyk L_1 není regulární.

15 bodů

3. Uvažujme jazyk L_s, jehož slova jsou n-tice binárních čísel[1] oddělené znakem $\#$. Konkrétněji, jazyk L_s obsahuje slova tvaru $w_1\#w_2\#\ldots\#w_n\#$, kde $w_1, \ldots, w_n \in \{0, 1\}^+$ jsou binární čísla. Tato slova odpovídají regulárnímu výrazu $R = ((\mathtt{0} + \mathtt{1})(\mathtt{0} + \mathtt{1})^*\#)^*$. Uvažujme dále omezení, že alespoň jedno číslo ve slově $w_1\#w_2\#\ldots\#w_n\#$ je sudé—tedy jeho poslední znak je 0. Formálně zapsáno:

$$L_s = \{w \in \{0, 1, \#\}^* \mid w \in \mathcal{L}(R) \wedge w \in \mathcal{L}((\mathtt{0} + \mathtt{1} + \#)^*\mathtt{0}\#(\mathtt{0} + \mathtt{1} + \#)^*)\}$$

   (a) Sestrojte nedeterministický konečný automat M_1 přijímající jazyk L_s (není nutné použít algoritmický postup).

   (b) Automat M_1 převeďte algoritmicky na deterministický konečný automat M_2.

20 bodů

[1] Jako binární číslo budeme chápat libovolný řetězec nad abecedou $\{0, 1\}$. Číslo tak může obsahovat počáteční nuly.

1 Úloha 1

1.1 Řešením soustavy rovnic nad regulárnímy výrazy sestavte k tomuto automatu ekvivalentní regulární výraz

Podle odchozích hran sestavíme soustavu rovnic, pro každý stav jednu rovnici. Pro lepší přehlednost jsem v následujícím postupu stavy značil velkými písmeny a symboly abecedy malými písmeny.

$$\begin{aligned} Q &= bS + aR \\ R &= bS + \epsilon \\ S &= cS + aR + bQ \end{aligned}$$

Dosadíme R do Q a S.

$$\begin{aligned} Q &= bS + abS + a \\ R &= bS + \epsilon \\ S &= cS + a(bS + \epsilon) + bQ \end{aligned}$$

Dosadíme Q do S a upravíme výsledné rovnice.

$$\begin{aligned} Q &= (b + ab)(c + ab + bb + bab)^*(a + ba) + a \\ R &= b((c + ab + bb + bab)^*(a + ba)) + \epsilon \\ S &= (c + ab + bb + bab)^*(a + ba) \end{aligned}$$

Výsledný regulární výraz je výraz, který se rovná Q, jelikož se jedná o startovací symbol a odpovídá tedy jazyku přijímanému daným automatem.

$(\mathbf{b} + \mathbf{ab})(\mathbf{c} + \mathbf{ab} + \mathbf{bb} + \mathbf{bab})^*(\mathbf{a} + \mathbf{ba}) + \mathbf{a}$ je výsledný regulární výraz.

1.2 Sestrojte relaci pravé kongruence s konečným indexem takovou, že L(M_3) je sjednocením některých tříd rozkladu a, b, c/

V tomto příkladě již využijeme značení zadaného automatu.

$$L^{-1}(q) = \epsilon \cup (\{ab, b\}(\{ab\}^* \{c\}^* \{bb\}^* \{bab\}^*)^* b)^*$$

$$L^{-1}(r) = \{a\} \cup (\{ab, b\}(\{ab\}^* \{c\}^* \{bb\}^* \{bab\}^*)^* \{a, ba\})^+$$

$$L^{-1}(s) = \{ab, b\}(\{ab\}^* \{c\}^* \{bb\}^* \{bab\}^*)^*$$

$$L(M_3) = L^{-1}(r)$$

2 Úloha 2

2.1 Sestrojte jazyk L_2 takový, že $L_1 \cap L_2 \in \mathcal{L}_3 \wedge L_1 \cup L_2 \in \mathcal{L}_3$

Uvažujme doplněk jazyka L_1 vůči universu $\{a, b, c\}^*$, potom:

$$L_2 = \{w | w \in \{a, b, c\}^* \wedge \#_a(w) \leq \#_b(w)\}$$

$$L_1 \cap L_2 = \emptyset$$

$$L_1 \cup L_2 = \{a, b, c\}^*$$

Prázdný jazyk je podle definice regulární jazyk. Výsledek sjednocení je také regulární jazyk, jelikož jsme schopni pro něj sestrojit konečný automat.

2.1.1 Důkaz neregularity pomocí pumping lemma pro L_2

Jazyk L_2 není regulární a pro důkaz zvolíme pumping lemma.
Důkaz sporem:
Předpokládejme, že jazyk $L_2 \in \mathcal{L}_3$. Pak tedy musí platit pumping lemma[1]:

$$L \in \mathcal{L}_3 \implies \exists p > 0 :$$

$$\forall w \in \Sigma^* : w \in L \wedge |w| \geq p \implies$$
$$(\exists x, y, z \in \Sigma^* : w = xyz \wedge y \neq \epsilon \wedge |xy| \leq p \wedge \forall i \geq 0 : xy^i z \in L)$$

Uvažme tedy libovolné p $>$ 0 splňující výše uvedenou podmínku. Zvolme řetězec $w = b^p a^p$, pro který platí:

$$w \in L_2 \wedge |w| = 2p \geq p$$

Dle pumping lemma víme, že pro daný řetězec tedy musí platit:

$$\exists x, y, z \in \{a, b, c\}^* : b^p a^p = xyz \wedge y \neq \epsilon \wedge |xy| \leq p \wedge \forall i \geq 0 : xy^i z \in L_2$$

Uvažme libovolné x,y,z splňující výše uvedené.
Zvolme i $=$ 0.

$$y = b^l, 0 < l \leq p$$
$$xy^0 z = b^{p-l} a^p$$
$$xy^0 z \in L_2 \wedge l > 0 \wedge p - l \geq p \, (SPOR)$$

Jestliže jsme došli ke sporu, tak musí platit původní tvrzení, tedy $L_1 \notin \mathcal{L}_3$

2.2 Důkaz neregularity pomocí pumping lemma pro L_1

Jazyk L_1 není regulární a pro důkaz zvolíme pumping lemma.
Důkaz sporem:
Předpokládejme, že jazyk $L_1 \in \mathcal{L}_3$. Pak tedy musí platit pumping lemma[1]:

$$L \in \mathcal{L}_3 \implies \exists p > 0 :$$

$$\forall w \in \Sigma^* : w \in L \wedge |w| \geq p \implies$$

$$(\exists x, y, z \in \Sigma^* : w = xyz \wedge y \neq \epsilon \wedge |xy| \leq p \wedge \forall i \geq 0 : xy^i z \in L)$$

Uvažme tedy libovolné p $>$ 0 splňující výše uvedenou podmínku. Zvolme řetězec $w = a^p a b^p$, pro který platí:

$$w \in L_1 \wedge |w| = 2p + 1 \geq p$$

Dle pumping lemma víme, že pro daný řetězec tedy musí platit:

$$\exists x, y, z \in \{a, b, c\}^* : a^p a b^p = xyz \wedge y \neq \epsilon \wedge |xy| \leq p \wedge \forall i \geq 0 : xy^i z \in L_1$$

Uvažme libovolné x,y,z splňující výše uvedené.
Zvolme i $=$ 0.

$$y = a^l, 0 < l \leq p$$

$$xy^0 z = a^{p-l+1} b^p$$

$$xy^0 z \in L_2 \wedge l > 0 \wedge p - l + 1 > p \; (SPOR)$$

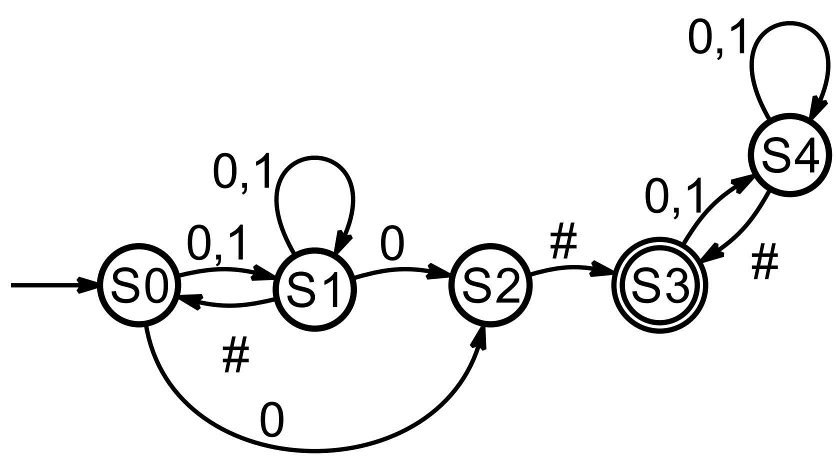
Jestliže jsme došli ke sporu, tak musí platit původní tvrzení, tedy $L_1 \notin \mathcal{L}_3$. Náš řetězec w $\in L_1$, jelikož $\#_a(w) = p + 1 \wedge \#_b(w) = p$. Ovšem podřetězec xy mohl obsahovat pouze znaky a. My jsme tento řetězec zkrátili, jelikož víme, že y je neprázdné. Pak víme, že jsme z řetězce vypustili minimálně jeden výskyt znaku a, tedy v novém řetězci je buďto stejný počet znaků a jako znaků b nebo je znaků b dokonce více. Každá z těchto situací porušuje podmínku jazyka L$_2$. Jestliže došlo ke sporu u negovaného výroku, tak původní výrok musí být pravdivý, jazyk L$_2$ není regulární.

3 Úloha 3

3.1 Nedeterministický konečný automat M_1 přijímající jazyk L_S

Nedeterministický automat je navržen tak, aby přijímal binární čísla oddělená mřížkou. Někdy se může rozhodnout, že přečte symbol *0*, který je následovaný symbolem #, čímž automat vyhodnotí, že přečetl sudé číslo. Nyní může buď skončit v koncovém stavu nebo číst další číslo. Determinismus je porušen ve stavu *S1*, kdy nevíme po přečtení symbolu *0* jestli máme zůstat ve stavu *S1* nebo přejít do stavu *S2*.[1]

Obrázek 1: Výsledný nedeterminický konečný automat



[1]Automat namodelován v programu AutomataEditor, který je dostupný na adrese: https://sourceforge.net/projects/automataeditor šířený pod licencí GPLv2

3.2 Algoritmický převod automatu $\mathbf{M_1}$ na deterministický konečný automat $\mathbf{M_2}$

Algoritmus pro převod NKA na DKA[1]:
Vstup NKA $M = (Q, \sigma, \delta, q_0, F)$
Výstup DKA $M' = (Q', \sigma, \delta', q_0', F')$
Postup:
1. Polož $Q' = 2^Q$.
2. Polož $q_0' = \{q_0\}$.
3. Polož $F' = \{S \mid S \in 2^Q \wedge S \cap F \neq \emptyset\}$.
4. Pro všechna $S \in 2^Q$ a pro všechna $a \in \Sigma$ polož:

$$\delta'(S, a) = \bigcup_{q \in S} \delta(q, a)$$

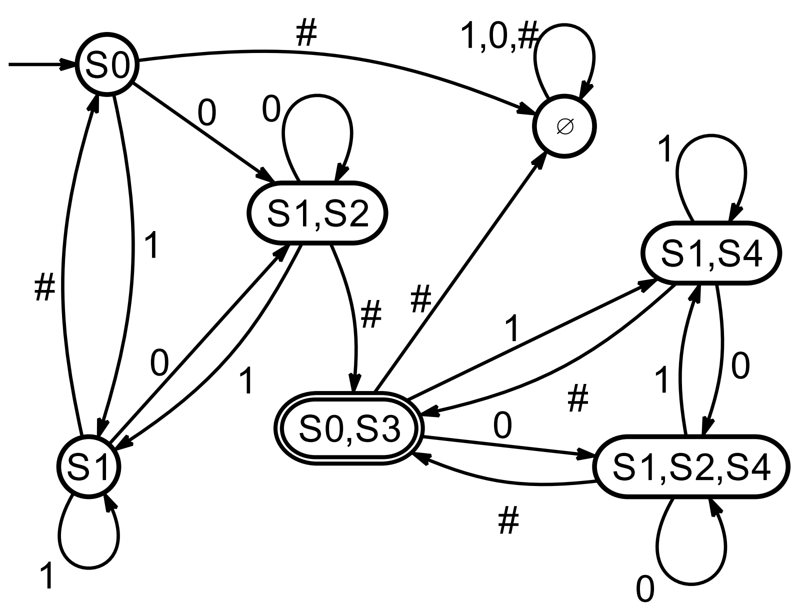
Tento algoritmus sice bude fungovat, nicméně pro ruční počítání nastává problém hned v bodě jedna, kdy nám exponenciálně vzroste počet stavů. Jelikož ve výsledném DKA nemusíme nutně potřebovat všechny možné stavy, tak můžeme začít od stávájícího automatu a přidávat pouze stavy, které budeme postupně potřebovat.

Tabulka 1: Výsledná tabulka stavů a přechodů po determinizaci

	0	**1**	#
->**{S0}**	{S1,S2}	{S1}	$\emptyset$
{S1}	{S1,S2}	{S1}	{S0}
{S1,S2}	{S1,S2}	{S1}	{S0,S3}
<-**{S0,S3}**	{S1,S2,S4}	{S1,S4}	$\emptyset$
{S1,S4}	{S1,S2,S4}	{S1,S4}	{S0,S3}
{S1,S2,S4}	{S1,S2,S4}	{S1,S4}	{S0,S3}
$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$

Jak je vidět, bylo nutné přidat pouze čtyři nové stavy, nicméně některé původní stavy se nám staly nedostupnými.[2]

Obrázek 2: Konečný automat převedený algoritmicky na deterministický



[2]Automat namodelován v programu AutomataEditor, který je dostupný na adrese: https://sourceforge.net/projects/automataeditor šířený pod licencí GPLv2

Reference

[1] doc. RNDr. Milan Češka, P.: Přednášky předmětu Teoretická informatika. 2016.