Teoretická informatika
2022/2023

# Domácí úloha 2

Pavel Šesták (xsesta07)

Brno, 13. listopadu 2022

# Obsah



# Seznam obrázků



# Seznam tabulek

**Teoretická informatika (TIN) – 2022/2023**
**Úkol 2**
(max. zisk 5 bodů – 10 bodů níže odpovídá 1 bodu v hodnocení předmětu)

1. Uvažujte abecedu $\Sigma = \{a, b, A, 0, B\}$. Sestrojte deterministický zásobníkový automat přijímající jazyk $L \subseteq \Sigma^*$ definovaný jako

$$\begin{aligned} L = &\{w.0 \mid w \in \{a,b\}^* \wedge \#_a(w) = \#_b(w)\} \cup \\ &\{w.A \mid w \in \{a,b\}^* \wedge \#_a(w) > \#_b(w)\} \cup \\ &\{w.B \mid w \in \{a,b\}^* \wedge \#_a(w) < \#_b(w)\}, \end{aligned}$$

kde $\#_x(w)$ pro $x \in \{a, b\}$ značí počet výskytů symbolu $x$ v řetězci $w$. Použijte abecedu zásobníkových symbolů $\Gamma = \{\times, \bullet\}$ a symbol $\times$ jako startovací symbol zásobníku. Automat zapište v grafické formě. Demonstrujte přijetí slova $abaabA$.

12 bodů

2. Uvažujte abecedu $\Sigma$ takovou, že $\Sigma \cap \{0, 1\} = \emptyset$ a jazyk $L = L(G)$ nad abecedou $\Sigma$ zadaný pomocí bezkontextové gramatiky $G$. Dále uvažujte jazyk $L'$ definovaný jako

$$L' = \{w.x \mid w \in \Sigma^* \wedge (w \in L \iff x = 1) \wedge (w \notin L \iff x = 0)\}.$$

Inspirujte se v kapitole o uzávěrových vlastnostech bezkontextových jazyků a proveďte jedno z následujících:

(a) Navrhněte algoritmus, který bude mít na vstupu gramatiku $G$ a sestaví bezkontextovou gramatiku $G'$ takovou, že $L(G') = L'$.

(b) Dokažte, že takový algoritmus neexistuje.

12 bodů

3. Navrhněte algoritmus, který pro bezkontextovou gramatiku $G = (N, \Sigma, P, S)$ spočítá množinu

$$N_{aa} = \{A \in N \mid \exists w \in \Sigma^* \colon A \Rightarrow^+_G w \wedge aa \text{ je podřetězec } w\}.$$

V algoritmu můžete využít množinu $N_\epsilon = \{A \in N \mid A \Rightarrow^+_G \epsilon\}$. Doporučujeme nadefinovat si další vhodné množiny neterminálů a algoritmicky popsat jejich výpočet (u $N_\epsilon$ popis výpočtu není potřeba).

Ilustrujte použití algoritmu na příkladě gramatiky s pravidly

$$S \to XY \mid YX \mid Z \qquad X \to Xba \mid \epsilon \qquad Y \to Yab \mid \epsilon \qquad Z \to XbaaU$$

14 bodů

4. Uvažujte následující operátor $\blacktriangle \colon 2^{\Sigma^*} \times 2^{\Sigma^*} \to 2^{\Sigma^*}$ na jazycích nad abecedou $\Sigma$:

$$L_1 \blacktriangle L_2 = \{w \in \Sigma^* \mid \forall w_1, w_2 \in \Sigma^* \colon w_1 w_2 = w \implies (w_1 \in L_1 \vee w_2 \in L_2)\}$$

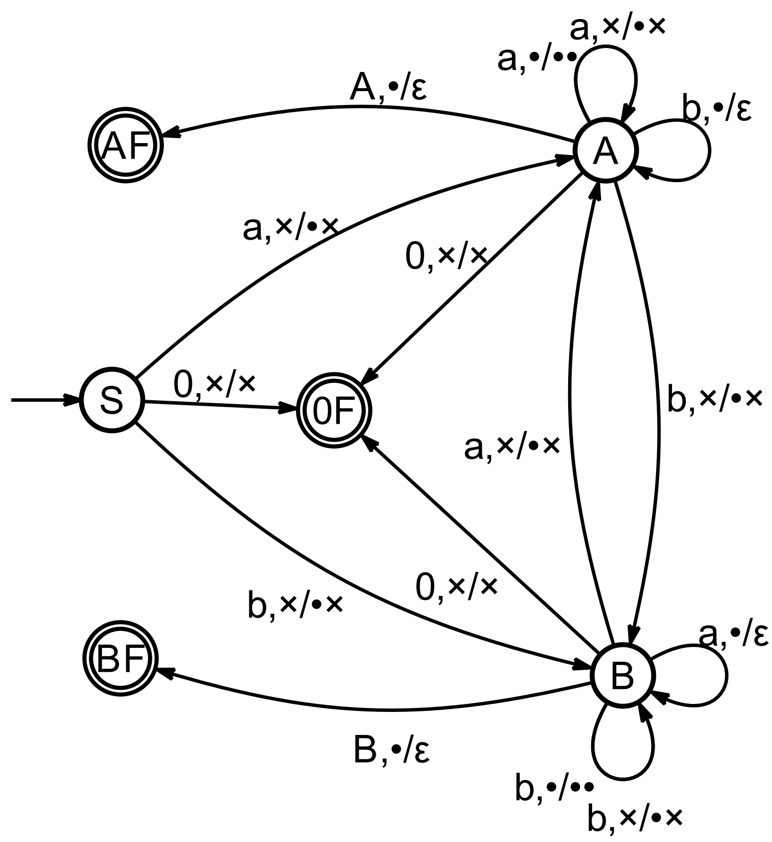
Dokažte, že množina rekurzivně vyčíslitelných jazyků je uzavřena na $\blacktriangle$.

12 bodů

# 1 Tvorba zásobníkového automatu

V této části zkonstruujeme deterministický konečný automat, který zapíšeme pomocí grafické notace. Poznámka k výslednému automatu, vstupní přechod do stavu S na své hraně obsahuje vložení dna zásobníku, v rámci SW pro tvorbu automatů se mi nepodařilo přidat popisek vstupní hraně.

Obrázek 1: Konstrukce deterministického zásobníkové automatu pro jazyk L




## 1.1 Demonstrace přijetí slova abaabA výše uvedeným automatem

(S, abaabA, $\times$) $\vdash (A, baabA, \bullet\times) \vdash (A, aabA, \times) \vdash (A, abA, \bullet\times) \vdash (A, bA, \bullet\bullet\times) \vdash (A, A, \bullet\times) \vdash$ $(AF, \epsilon,\ \times)$

# 2 Algoritmus na sestavení bezkontextové gramatiky G'

Algoritmus nelze sestavit, jelikož výstupní gramatika G' má být bezkontextová. V definici jazyka se pracuje s bezkontextovým jazykem L a jeho doplňkem. Kdyby bylo specifikované, že jazyk L jsme schopni přijímat DZA pak by daný problém šel vyřešit nicméně obecně neplatí, že doplněk bezkontextového jazyka je bezkontextový což plyne z uzávěrových vlastností pro $\mathcal{L}_2$. Jakmile by doplněk nebyl bezkontextový, pak by tento jazyk nešlo přijímat bezkontextovou gramatikou. Příklad jazyka $L = co - \{a^n b^n c^n | n \geq 0\} \in \mathcal{L}_2$, ale $co - L = \{a^n b^n c^n | n \geq 0\} \notin \mathcal{L}_2$

## 3 Návrh algoritmu pro výpočet množiny $N_{aa}$

**Algorithm 1:** Algoritmus pro nalezení množiny neterminálů, které se umí přepsat na řetězec terminálních symbolů.

**Input** : BKG = (N, $\Sigma, P, S$)
**Output:** $N_T = \{A \in N \mid \exists w \in \Sigma^*$: A $\Rightarrow_G^+$ w $\}$

Vypočti algoritmicky $N_\epsilon$
$N_T^0 = \emptyset$
i = 0
**repeat**
  $N_T^{i+1} = N_T^i \cup \{A \in N | \exists A \rightarrow \alpha \in P \wedge \alpha \in (\Sigma \cup N_T^i)^*\}$
  i = i + 1
**until** $N_T^i \neq N_T^{i-1}$;
**return** $N_T^i$

**Algorithm 2:** Algoritmus pro nalezení množiny neterminálů, které se umí přepsat na řetězec terminálních symbolů a začínají symbolem a.

**Input** : BKG = (N, $\Sigma, P, S$)
**Output:** $N_{start\_a} = \{A \in N \mid \exists w \in \Sigma^*$: A $\Rightarrow_G^+$ aw $\}$

Vypočti algoritmicky $N_\epsilon$
Vypočti algoritmicky $N_T$
$N_{start\_a}^0 = \emptyset$
i = 0
**repeat**
  $N_{start\_a}^{i+1} = N_{start\_a}^i \cup \{A \in N | \exists A \rightarrow \alpha \in P \wedge \alpha \in (N_\epsilon)^*(\{a\} \cup N_{start\_a}^i)(N_T \cup \Sigma)^*\}$
  i = i + 1
**until** $N_{start\_a}^i \neq N_{start\_a}^{i-1}$;
**return** $N_{start\_a}^i$

**Algorithm 3:** Algoritmus pro nalezení množiny neterminálů, které se umí přepsat na řetězec terminálních symbolů a končící symbolem a.

**Input** : BKG = (N, $\Sigma, P, S$)
**Output:** $N_{end\_a} = \{A \in N \mid \exists w \in \Sigma^*$: A $\Rightarrow_G^+$ wa $\}$

Vypočti algoritmicky $N_\epsilon$
Vypočti algoritmicky $N_T$
$N_{end\_a}^0 = \emptyset$
i = 0
**repeat**
  $N_{end\_a}^{i+1} = N_{end\_a}^i \cup \{A \in N | \exists A \rightarrow \alpha \in P \wedge \alpha \in (N_T \cup \Sigma)^*(\{a\} \cup N_{end\_a}^i)(N_\epsilon)^*\}$
  i = i + 1
**until** $N_{end\_a}^i \neq N_{end\_a}^{i-1}$;
**return** $N_{end\_a}^i$

**Algorithm 4:** Algoritmus pro nalezení množiny neterminálů, které se umí přepsat na řetězec terminálních symbolů s podřetězcem aa.

**Input** : BKG = (N, $\Sigma$, $P$, $S$)
**Output:** $N_{aa} = \{A \in N \mid \exists w \in \Sigma^*$: A $\Rightarrow_G^+$ w $\wedge$ aa je podřetězec w$\}$
Vypočti algoritmicky $N_T$
Vypočti algoritmicky $N_{start\_a}$
Vypočti algoritmicky $N_\epsilon$
Vypočti algoritmicky $N_{end\_a}$
$N_{aa}^0 = \emptyset$
i = 0
**repeat**
  $N_{aa}^{i+1} = N_{aa}^i \cup \{A \in N | \exists A \rightarrow \alpha \in P \wedge \alpha \in (N_T \cup \Sigma)^*(N_{aa}^i \cup (\{a\} \cup N_{end\_a})N_\epsilon^*(N_{start\_a} \cup \{a\}))(N_T \cup \Sigma)^*\}$
  i = i + 1
**until** $N_{aa}^i \neq N_{aa}^{i-1}$;
**return** $N_{aa}^i$

## 3.1 Demonstrace použití algoritmu na gramatice G

G = ({S, X, Y, Z, U}, {a,b},
{S $\rightarrow XY|YX|Z$
$X \rightarrow Xba|\epsilon$
Y $\rightarrow Yab|\epsilon$
Z $\rightarrow XbaaU\}$,
$S$)

### 3.1.1 Výpočet množiny $N_\epsilon$

$N_\epsilon^0 = \{\}$
$N_\epsilon^1 = \{X, Y\}$
$N_\epsilon^2 = \{X, Y, S\}$
$N_\epsilon^3 = \{X, Y, S\}$

### 3.1.2 Výpočet množiny $N_T$

$N_T^0 = \{\}$
$N_T^1 = \{X, Y\}$
$N_T^2 = \{X, Y, S\}$
$N_T^3 = \{X, Y, S\}$

### 3.1.3 Výpočet množiny $N_{start\_a}$

$N_{start\_a}^0 = \{\}$
$N_{start\_a}^1 = \{Y\}$
$N_{start\_a}^2 = \{Y, S\}$
$N_{start\_a}^3 = \{Y, S\}$

### 3.1.4 Výpočet množiny $N_{end\_a}$

$N_{end\_a}^0 = \{\}$
$N_{end\_a}^1 = \{X\}$
$N_{end\_a}^2 = \{X, S\}$
$N_{end\_a}^3 = \{X, S\}$

### 3.1.5 Výpočet množiny $N_{aa}$

$N_{aa}^0 = \{\}$
$N_{aa}^1 = \{S\}$
$N_{aa}^2 = \{S\}$

# 4 Důkaz uzavřenosti rekurzivně vyčíslitelných jazyků nad operací ▲

Operátor ▲ :$2^{\Sigma^*} \times 2^{\Sigma^*} \rightarrow 2^{\Sigma^*}$ pro jazyky nad abecedou $\Sigma$:

$$L_1 \blacktriangle L_2 = \{w \in \Sigma^* | \forall w_1, w_2 \in \Sigma^* : w_1 w_2 = w \implies (w_1 \in L_1 \lor w_2 \in L_2)\}$$

Důkaz uzavřenosti operace ▲ pro rekurzivně vyčíslitelnými jazyky:

$$L_1 \in \mathcal{L}_{RE} \implies \exists\, TS\; M_1 : L(M_1) = L_1$$
$$L_2 \in \mathcal{L}_{RE} \implies \exists\, TS\; M_2 : L(M_2) = L_2$$

Jako důkaz uzavřenosti sestrojíme TS M', který bude rozhodovat operaci▲. TS M' je pěti páskový TS, přičemž obsah pásek je popsán zde:
1. páska obsahuje zakodovane Turingovy stroje a slovo
2. páska obsahuje slovo w
3. páska obsahuje počítadlo pomocí sekvence symbolů a
4. páska obsahuje slovo $w_1$
5. páska obsahuje slovo $w_2$

---

**Algorithm 5:** Algoritmus popisující chování TS M', který ověřuje operaci ▲

---

**Input :** $\triangle < M1 > \# < M2 > \# w \triangle$, kde
<M1> je zakódovaný TS $M_1 : L(M_1) = L_1$,
<M2> je zakódovaný TS $M_2 : L(M_2) = L_2$,
w $\in \Sigma^*$,
\# jsou separátory mezi jednotlivými částmi,
$\triangle$ reprezentuje blank symbol

1 M' zkontroluje, že <M1> a <M2> je kód TS
2 M' překopíruje slovo w z první pásky na druhou.
3 M' Přesune hlavu druhé pásky na začátek.
4 M' synchronizovaně čte druhou a třetí pásku zleva doprava, přičemž kopíruje čtený obsah druhé pásky na čtvrtou pásku, dokud hlava na třetí pásce nepřečte symbol $\triangle$.
5 M' překopíruje nepřečtený suffix w na pátou pásku.
6 M' Přesune hlavu čtvrté a páté pásky na začátek.
7 M' krokuje současně simulaci TS $M_1$ nad čtvrtou páskou a simulaci TS $M_2$ nad pátou páskou.
8 **if** $M_1$ *akceptuje* $\lor M_2$ *akceptuje* **then**
9     **if** *pokud délka řetězce na třetí pásce je shodná s délkou řetězce na druhé pásce* **then**
10         M'akceptuje.
11     **else**
12         M' vyčistí čtvrtou a pátou pásku
13         M' Zapíše symbol "a"na konec třetí pásky.
14         M' Přesune hlavu první, třetí, čtvrté a páté pásky na začátek a pokračuje bodem 2.
15     **end**
16 **end**
17 **if** $M_1$ *zamítne* $\land M_2$ *zamítne* **then**
18     M' zamítne.
19 **end**
20 **if** $M_1$ *nepřijme* $\lor M_2$ *nepřijme* **then**
21     M' cyklí.
22 **end**

---