Formální jazyky a automaty

Uzávěrové vlastnosti a (ne)rozhodnutelné problémy pro CFL, deterministické CFL, Turingovy stroje

Jan Křetínský

Fakulta informatiky, MU Brno

Jaro 2024

Vlastnosti bezkontextových jazyků

Věta 3.58. (a 3.61.)

Třída bezkontextových jazyků ($\mathcal{L}_2$) **je** uzavřena vzhledem k operacím:

...

Věta 3.60.

Třída bezkontextových jazyků ($\mathcal{L}_2$) **není** uzavřena vzhledem k operacím:

...

Sjednocení

L_1 je generován CFG $\mathcal{G}_1 = (N_1, \Sigma_1, P_1, S_1)$ a
L_2 je generován CFG $\mathcal{G}_2 = (N_2, \Sigma_2, P_2, S_2)$.
Bez újmy na obecnosti můžeme předpokládat $N_1 \cap N_2 = \emptyset$.

Sjednocení

L_1 je generován CFG $\mathcal{G}_1 = (N_1, \Sigma_1, P_1, S_1)$ a
L_2 je generován CFG $\mathcal{G}_2 = (N_2, \Sigma_2, P_2, S_2)$.
Bez újmy na obecnosti můžeme předpokládat $N_1 \cap N_2 = \emptyset$.

Definujeme $\mathcal{G} = (N_1 \cup N_2 \cup \{S\}, \Sigma_1 \cup \Sigma_2, P, S)$, kde S je nový symbol a

$$P = P_1 \cup P_2 \cup \{S \to S_1, S \to S_2\}.$$

Každá derivace v $\mathcal{G}$ začne použitím buď $S \to S_1$ nebo $S \to S_2$.
Podmínka $N_1 \cap N_2 = \emptyset$ zaručí, že při použití $S \to S_1$ (resp. $S \to S_2$) lze v dalším derivování používat jen pravidla z P_1 (resp. P_2).
Jazyk $L = L_1 \cup L_2$ je generován gramatikou $\mathcal{G}$.

Zřetězení

L_1 je generován CFG $\mathcal{G}_1 = (N_1, \Sigma_1, P_1, S_1)$ a
L_2 je generován CFG $\mathcal{G}_2 = (N_2, \Sigma_2, P_2, S_2)$.
Bez újmy na obecnosti můžeme předpokládat $N_1 \cap N_2 = \emptyset$.

Zřetězení

L_1 je generován CFG $\mathcal{G}_1 = (N_1, \Sigma_1, P_1, S_1)$ a
L_2 je generován CFG $\mathcal{G}_2 = (N_2, \Sigma_2, P_2, S_2)$.
Bez újmy na obecnosti můžeme předpokládat $N_1 \cap N_2 = \emptyset$.

Definujeme $\mathcal{G} = (N_1 \cup N_2 \cup \{S\}, \Sigma_1 \cup \Sigma_2, P, S)$, kde S je nový symbol a

$$P = P_1 \cup P_2 \cup \{S \to S_1 S_2\}.$$

Jazyk $L = L_1.L_2$ je generován gramatikou $\mathcal{G}$.

Iterace a pozitivní iterace

L_1 je generován CFG $\mathcal{G}_1 = (N_1, \Sigma_1, P_1, S_1)$.

Iterace a pozitivní iterace

L_1 je generován CFG $\mathcal{G}_1 = (N_1, \Sigma_1, P_1, S_1)$.

Definujeme $\mathcal{G} = (N_1 \cup \{S\}, \Sigma_1, P, S)$, kde S je nový symbol a

$$P = P_1 \cup \{S \to SS_1 \mid \varepsilon\}.$$

Jazyk $L = L_1^*$ je generován gramatikou $\mathcal{G}$.

Definujeme $\mathcal{G} = (N_1 \cup \{S\}, \Sigma_1, P, S)$, kde S je nový symbol a

$$P = P_1 \cup \{S \to SS_1 \mid S_1\}.$$

Jazyk $L = L_1^+$ je generován gramatikou $\mathcal{G}$.

Průnik

$L_1 = \{a^n b^n c^m \mid m, n \geq 1\}$ $L_2 = \{a^n b^m c^n \mid m, n \geq 1\}$

Oba tyto jazyky jsou CFL.

Kbyby $\mathcal{L}_2$ byla uzavřena vzhledem k operaci průniku, pak by i $L_1 \cap L_2 = \{a^n b^n c^n \mid n \geq 1\}$ musel být bezkontextový, což však není.

Doplněk

Neuzavřenost $\mathcal{L}_2$ vůči doplňku plyne z její uzavřenosti na sjednocení, neuzavřenosti na průnik a z De Morganových pravidel:

$$L_1 \cap L_2 = co{-}(co{-}L_1 \cup co{-}L_2),$$

tj., kdyby $\mathcal{L}_2$ byla uzavřena na doplněk, musela by být uzavřena i na průnik, což však není.

Doplněk

Neuzavřenost $\mathcal{L}_2$ vůči doplňku plyne z její uzavřenosti na sjednocení, neuzavřenosti na průnik a z De Morganových pravidel:

$$L_1 \cap L_2 = co\text{-}(co\text{-}L_1 \cup co\text{-}L_2),$$

tj., kdyby $\mathcal{L}_2$ byla uzavřena na doplněk, musela by být uzavřena i na průnik, což však není.

(Další) protipříklad k uzavřenosti na doplněk:

$L = \{ww \mid w \in \{a, b\}^*\}$ není CFL.
$co\text{-}L$ je CFL.

Průnik s regulárním jazykem

$L = L(\mathcal{P})$, kde $\mathcal{P}$ je PDA $\mathcal{P} = (Q_1, \Sigma, \Gamma, \delta_1, q_1, Z_0, F_1)$
$R = L(\mathcal{A})$, kde $\mathcal{A}$ je deterministický FA $\mathcal{A} = (Q_2, \Sigma, \delta_2, q_2, F_2)$
Sestrojíme PDA $\mathcal{P}'$ takový, že $L(\mathcal{P}') = L \cap R$.
$\mathcal{P}' = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$, kde

- $Q = Q_1 \times Q_2$
- $q_0 = \langle q_1, q_2 \rangle$
- $F = F_1 \times F_2$
- δ : pro každé $p \in Q_1$, $q \in Q_2$, $a \in \Sigma \cup \{\varepsilon\}$, $Z \in \Gamma$ platí:

$$\delta(\langle p, q \rangle, a, Z) = \{(\langle p', q' \rangle, \gamma) \mid (p', \gamma) \in \delta_1(p, a, Z) \text{ a } \widehat{\delta_2}(q, a) = q'\}$$

Zřejmě platí $w \in L(\mathcal{P}') \iff w \in L(\mathcal{P}) \cap L(\mathcal{A})$.

Vlastnosti bezkontextových jazyků

Věta 3.58. (a 3.61.)

Třída bezkontextových jazyků ($\mathcal{L}_2$) **je** uzavřena vzhledem k operacím:

1. sjednocení
2. zřetězení
3. iterace
4. pozitivní iterace
5. průnik s regulárním jazykem

Věta 3.60.

Třída bezkontextových jazyků ($\mathcal{L}_2$) **není** uzavřena vzhledem k operacím:

1. průnik
2. doplněk

Rozhodnutelné problémy pro bezkontextové jazyky

Problém příslušnosti
Existuje algoritmus, který pro libovolnou danou CFG $\mathcal{G}$ a slovo w rozhoduje, zda $w \in L(\mathcal{G})$ či nikoliv.

Problém prázdnosti
Existuje algoritmus, který pro libovolnou danou CFG $\mathcal{G}$ rozhoduje, zda $L(\mathcal{G}) = \emptyset$ či nikoliv.

Problém konečnosti
Existuje algoritmus, který pro libovolnou danou CFG $\mathcal{G}$ rozhoduje, zda $L(\mathcal{G})$ je konečný či nikoliv.

Rozhodnutelné problémy pro bezkontextové jazyky

Problém příslušnosti
Existuje algoritmus, který pro libovolnou danou CFG $\mathcal{G}$ a slovo w rozhoduje, zda $w \in L(\mathcal{G})$ či nikoliv.

Problém prázdnosti
Existuje algoritmus, který pro libovolnou danou CFG $\mathcal{G}$ rozhoduje, zda $L(\mathcal{G}) = \emptyset$ či nikoliv.

Problém konečnosti
Existuje algoritmus, který pro libovolnou danou CFG $\mathcal{G}$ rozhoduje, zda $L(\mathcal{G})$ je konečný či nikoliv.

Věta 3.68.

Ke každé CFG $\mathcal{G}$ lze sestrojit čísla m, n taková, že $L(\mathcal{G})$ je nekonečný právě když existuje slovo $z \in L(\mathcal{G})$ takové, že $m < |z| \leq n$.

Důkaz: pomocí Lemmatu o vkládání, analogicky regulárnímu případu.

Vlastnost sebevložení

Definice 3.70.

Nechť $\mathcal{G} = (N, \Sigma, P, S)$ je CFG. Řekneme, že $\mathcal{G}$ má vlastnost sebevložení, jestliže existují $A \in N$ a $u, v \in \Sigma^+$ taková, že $A \Rightarrow^+ uAv$. CFL L má vlastnost sebevložení, jestliže každá bezkontextová gramatika, která jej generuje, má vlastnost sebevložení.

Věta 3.71.

CFL L má vlastnost sebevložení, právě když L není regulární.

Důkaz. Viz skripta.

Nerozhodnutelné problémy pro bezkontextové jazyky

Problém regularity
Neexistuje algoritmus, který pro libovolnou danou CFG $\mathcal{G}$ rozhoduje, zda $L(\mathcal{G})$ je regulární či nikoliv.
(Tedy není rozhodnutelné, zda $L(\mathcal{G})$ má vlastnost sebevložení či nikoliv.)

Problém univerzality
Neexistuje algoritmus, který pro libovolnou danou CFG $\mathcal{G}$ rozhoduje, zda $L(\mathcal{G}) = \Sigma^*$ či nikoliv.

Problémy ekvivalence a inkluze také nejsou rozhodnutelné (plyne z nerozhodnutelnosti problému univerzality).

Deterministické zásobníkové automaty

Definice 3.72.

Řekneme, že PDA $\mathcal{M} = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$ je deterministický (DPDA), jestliže jsou splněny tyto podmínky:

1. pro žádné $q \in Q$, $Z \in \Gamma$ a $a \in \Sigma \cup \{\varepsilon\}$ neobsahuje $\delta(q, a, Z)$ více než jeden prvek
2. pro všechna $q \in Q$ a $Z \in \Gamma$ platí:
   kdykoliv $\delta(q, \varepsilon, Z) \neq \emptyset$, pak $\delta(q, a, Z) = \emptyset$ pro všechna $a \in \Sigma$

Řekneme, že L je deterministický bezkontextový jazyk (DCFL), právě když existuje DPDA $\mathcal{M}$ takový, že $L = L(\mathcal{M})$.

Vlastnosti deterministických bezkontextových jazyků

Věta 3.82.

Třída DCFL **je** uzavřena na doplňek.

Intuice:

- DPDA má nad každým slovem právě jeden výpočet.
- Pro doplňek stačí zaměnit koncové a nekoncové stavy.

Vlastnosti deterministických bezkontextových jazyků

Věta 3.82.

Třída DCFL **je** uzavřena na doplňek.

Intuice:

- DPDA má nad každým slovem právě jeden výpočet.
- Pro doplňek stačí zaměnit koncové a nekoncové stavy.
- Musí dočíst vstup až do konce!

DPDA by nemusel dočíst vstupní slovo do konce, protože

- se vyprázdní zásobník nebo přechod není definován

Vlastnosti deterministických bezkontextových jazyků

Věta 3.82.

Třída DCFL **je** uzavřena na doplňek.

Intuice:

- DPDA má nad každým slovem právě jeden výpočet.
- Pro doplňek stačí zaměnit koncové a nekoncové stavy.
- Musí dočíst vstup až do konce!

DPDA by nemusel dočíst vstupní slovo do konce, protože

- se vyprázdní zásobník nebo přechod není definován
- přestane číst vstup a jen provádí ε-kroky pod kterými zásobník neomezeně roste

Vlastnosti deterministických bezkontextových jazyků

Věta 3.82.

Třída DCFL **je** uzavřena na doplňek.

Intuice:

- DPDA má nad každým slovem právě jeden výpočet.
- Pro doplňek stačí zaměnit koncové a nekoncové stavy.
- Musí dočíst vstup až do konce!

DPDA by nemusel dočíst vstupní slovo do konce, protože

- se vyprázdní zásobník nebo přechod není definován
- přestane číst vstup a jen provádí ε-kroky pod kterými zásobník neomezeně roste

Vlastnosti deterministických bezkontextových jazyků

Věta 3.82.

Třída DCFL **je** uzavřena na doplňek.

Intuice:

- DPDA má nad každým slovem právě jeden výpočet.
- Pro doplňek stačí zaměnit koncové a nekoncové stavy.
- Musí dočíst vstup až do konce!

DPDA by nemusel dočíst vstupní slovo do konce, protože

- se vyprázdní zásobník nebo přechod není definován
- přestane číst vstup a jen provádí ε-kroky pod kterými zásobník neomezeně roste $\iff$ vzroste o $|Q| \cdot |\Gamma| \cdot$ max.délka pravidla

Vlastnosti deterministických bezkontextových jazyků

Věta 3.82.

Třída DCFL **je** uzavřena na doplňek.

Intuice:

- DPDA má nad každým slovem právě jeden výpočet.
- Pro doplňek stačí zaměnit koncové a nekoncové stavy.
- Musí dočíst vstup až do konce!

DPDA by nemusel dočíst vstupní slovo do konce, protože

- se vyprázdní zásobník nebo přechod není definován
- přestane číst vstup a jen provádí ε-kroky pod kterými zásobník neomezeně roste $\iff$ vzroste o $|Q| \cdot |\Gamma| \cdot$ max.délka pravidla
- cyklí

Vlastnosti deterministických bezkontextových jazyků

Věta 3.82.

Třída DCFL **je** uzavřena na doplňek.

Intuice:

- DPDA má nad každým slovem právě jeden výpočet.
- Pro doplňek stačí zaměnit koncové a nekoncové stavy.
- Musí dočíst vstup až do konce!

DPDA by nemusel dočíst vstupní slovo do konce, protože

- se vyprázdní zásobník nebo přechod není definován
- přestane číst vstup a jen provádí ε-kroky pod kterými zásobník neomezeně roste $\iff$ vzroste o $|Q| \cdot |\Gamma| \cdot$ max.délka pravidla
- cyklí
- dočte slovo, ale pak pod ε-kroky prochází koncové i nekoncové stavy (tj. některá slova jsou akceptována původním DPDA i DPDA se zaměněnými koncovými stavy).

Průnik a sjednocení

Věta.

Třída DCFL **není** uzavřena na průnik.

Věta 3.84.

Třída DCFL **není** uzavřena na sjednocení.

Důkaz. Plyne z uzavřenosti na doplňek, neuzavřenosti na průnik a z De Morganových pravidel:

$$L_1 \cap L_2 = co\text{-}(co\text{-}L_1 \cup co\text{-}L_2)$$

(Z uzavřenosti na sjednocení by plynula uzavřenost na průnik.) □

Věta 3.86.

Třída DCFL tvoří vlastní podtřídu třídy bezkontextových jazyků.

Příklad. Jazyk $co\text{-}\{ww \mid w \in \{a, b\}^*\}$ je CFL, ale není DCFL.

Aplikace (deterministických) bezkontextových jazyků

- syntaxe programovacích jazyků je definována pomocí CFG (dobře uzávorkované výrazy, *if–then–else* konstrukty)
- DTD (Document Type definition) umožňuje definovat bezkontexové jazyky – využití ve značkovacích jazycích (HTML, XML,...)
- nástroje pro tvorbu parserů/překladačů využívají různé algoritmy pro lineární deterministickou syntaktickou analýzu:
  LALR(1) - Yacc, Bison, javacup
  LL(k) - JavaCC, ANTLR

Turingův stroj

Turingův stroj

Definice (Syntax)

(Deterministický) Turingův stroj (Turing Machine, TM) je devítice $\mathcal{M} = (Q, \Sigma, \Gamma, \triangleright, \sqcup, \delta, q_0, q_{\text{acc}}, q_{\text{rej}})$, kde

- Q je konečná množina, jejíž prvky nazýváme stavy,
- Σ je konečná množina, tzv. vstupní abeceda,
- Γ je konečná množina, tzv. pracovní abeceda, $\Sigma \subseteq \Gamma$,
- $\triangleright \in \Gamma \setminus \Sigma$ je levá koncová značka,
- $\sqcup \in \Gamma \setminus \Sigma$ je symbol označující prázdné políčko,
- $\delta : (Q \setminus \{q_{\text{acc}}, q_{\text{rej}}\}) \times \Gamma \to Q \times \Gamma \times \{L, R\}$ je totální přechodová funkce,
- $q_0 \in Q$ je počáteční stav,
- $q_{\text{acc}} \in Q$ je akceptující stav,
- $q_{\text{rej}} \in Q$ je zamítající stav, $q_{\text{acc}} \neq q_{\text{rej}}$

a pro každé $\delta(q, \triangleright) = (p, x, X)$ je $x = \triangleright, X = R$ (tj. $\triangleright$ nelze přepsat ani posunout hlavu za okraj pásky).

Konfigurace Turingova stroje

Definice.

Konfigurace TM je trojice $(q, z, n) \in Q \times \{y\sqcup^\omega \mid y \in \Gamma^*\} \times \mathbb{N}_0$, kde

- q je stav,
- $y\sqcup^\omega$ je obsah pásky, (kde $\sqcup^\omega = \sqcup\sqcup\sqcup\sqcup\sqcup\sqcup\ldots$)
- n značí pozici hlavy na pásce.

Počáteční konfigurace pro vstup $w \in \Sigma^*$ je trojice $(q_0, \triangleright w\sqcup^\omega, 0)$.

Akceptující konfigurace je každá trojice tvaru (q_{acc}, z, n).

Zamítající konfigurace je každá trojice tvaru (q_{rej}, z, n).

Výpočet Turingova stroje

Definice.
Na množině všech konfigurací stroje $\mathcal{M}$ definujeme binární relaci krok výpočtu $\vdash_{\mathcal{M}}$ takto:

$$(p, z, n) \quad \vdash_{\mathcal{M}} \quad \begin{cases} (q, s_b^n(z), n+1) & \text{pro } \delta(p, z_n) = (q, b, R) \\ (q, s_b^n(z), n-1) & \text{pro } \delta(p, z_n) = (q, b, L) \end{cases}$$

kde z_n označuje n-tý symbol řetězu z (z_0 je nejlevější symbol řetězu z) a $s_b^n(z)$ řetěz vzniklý ze z nahrazením z_n symbolem b.

Výpočet Turingova stroje

Definice.
Na množině všech konfigurací stroje $\mathcal{M}$ definujeme binární relaci krok výpočtu $\vdash_{\mathcal{M}}$ takto:

$$(p, z, n) \quad \vdash_{\mathcal{M}} \quad \begin{cases} (q, s_b^n(z), n+1) & \text{pro } \delta(p, z_n) = (q, b, R) \\ (q, s_b^n(z), n-1) & \text{pro } \delta(p, z_n) = (q, b, L) \end{cases}$$

kde z_n označuje n-tý symbol řetězu z (z_0 je nejlevější symbol řetězu z) a $s_b^n(z)$ řetěz vzniklý ze z nahrazením z_n symbolem b.

Výpočet TM $\mathcal{M}$ na vstupu w je maximální (konečná nebo nekonečná) posloupnost konfigurací $K_0, K_1, K_2, \ldots$, kde K_0 je počáteční konfigurace pro w a $K_i \vdash_{\mathcal{M}} K_{i+1}$ pro všechna $i \geq 0$.

Jazyk Turingova stroje

Stroj $\mathcal{M}$ akceptuje slovo w právě když výpočet $\mathcal{M}$ na w je konečný a jeho poslední konfigurace je akceptující.
Stroj $\mathcal{M}$ zamítá slovo w právě když výpočet $\mathcal{M}$ na w je konečný a jeho poslední konfigurace je zamítající.
Stroj $\mathcal{M}$ pro vstup w cyklí právě když výpočet $\mathcal{M}$ na w je nekonečný.

Jazyk akceptovaný TM $\mathcal{M}$ definujeme jako

$$L(\mathcal{M}) = \{w \in \Sigma^* \mid \mathcal{M} \text{ akceptuje } w\}.$$

Jazyk Turingova stroje

Stroj $\mathcal{M}$ akceptuje slovo w právě když výpočet $\mathcal{M}$ na w je konečný a jeho poslední konfigurace je akceptující.
Stroj $\mathcal{M}$ zamítá slovo w právě když výpočet $\mathcal{M}$ na w je konečný a jeho poslední konfigurace je zamítající.
Stroj $\mathcal{M}$ pro vstup w cyklí právě když výpočet $\mathcal{M}$ na w je nekonečný.

Jazyk akceptovaný TM $\mathcal{M}$ definujeme jako

$$L(\mathcal{M}) = \{w \in \Sigma^* \mid \mathcal{M} \text{ akceptuje } w\}.$$

Stroj $\mathcal{M}$ se nazývá úplný, je-li každý jeho výpočet konečný (akceptující nebo zamítající).

Příklad

$$L = \{xux \mid x \in \{a, b\},\ u \in \{a, b\}^*\} \cup \{a, b\}$$

Příklad

$L = \{xux \mid x \in \{a, b\},\ u \in \{a, b\}^*\} \cup \{a, b\}$

Formálně, $\mathcal{M} = (Q, \Sigma, \Gamma, \triangleright, \sqcup, \delta, s, q_{\text{acc}}, q_{\text{rej}})$, kde

$$Q = \{s, q_{\text{acc}}, q_{\text{rej}}, [q_1, a], [q_2, a], [q_1, b], [q_2, b]\},$$
$$\Sigma = \{a, b\},$$
$$\Gamma = \Sigma \cup \{\triangleright, \sqcup\}.$$

δ:

	$\triangleright$	a	b	$\sqcup$
s	$(s, \triangleright, R)$	$([q_1, a], a, R)$	$([q_1, b], b, R)$	$(q_{\text{rej}}, -, -)$
$[q_1, a]$	$-$	$([q_1, a], a, R)$	$([q_1, a], b, R)$	$([q_2, a], \sqcup, L)$
$[q_1, b]$	$-$	$([q_1, b], a, R)$	$([q_1, b], b, R)$	$([q_2, b], \sqcup, L)$
$[q_2, a]$	$-$	$(q_{\text{acc}}, -, -)$	$(q_{\text{acc}}, -, -)$	$-$
$[q_2, b]$	$-$	$(q_{\text{rej}}, -, -)$	$(q_{\text{rej}}, -, -)$	$-$

Přehled jazykových tříd

Jazyky	Gramatiky (typ)	Automaty
rekursivně spočetné	frázové (0)	Turingovy stroje
rekursivní	-	úplné Turingovy stroje
kontextové	kontextové (1)	lineárně ohraničené TM
bezkontextové	bezkontextové (2)	zásobníkové automaty
deterministické CFL	-	deterministické PDA
regulární	regulární (3)	konečné automaty

Třída na nižším řádku je vždy vlastní podtřídou třídy na vyšším řádku.

Uzávěrové vlastnosti

Věta 4.16.

Třída rekurzivních jazyků je uzavřena vzhledem k operaci komplementu.

Uzávěrové vlastnosti

Věta 4.16.

Třída rekurzivních jazyků je uzavřena vzhledem k operaci komplementu.

Věta 4.17.

Pokud jsou L i $co-L$ oba rekurzivně spočetné, pak jsou oba rekurzivní.

Uzávěrové vlastnosti

Věta 4.16.

Třída rekurzivních jazyků je uzavřena vzhledem k operaci komplementu.

Věta 4.17.

Pokud jsou L i $co-L$ oba rekurzivně spočetné, pak jsou oba rekurzivní.

Věta 4.15.

Třídy rekurzivních a rekurzivně spočetných jazyků jsou uzavřeny vzhledem k operacím sjednocení, průniku, zřetězení a iteraci.