Formální jazyky a automaty

Syntaktická analýza, přímá a nepřímá levá rekurze, Greibachové normální forma

Jan Křetínský

Fakulta informatiky, MU Brno

Jaro 2024

Bezkontextové gramatiky $\rightarrow$ zásobníkové automaty

Motivace 1: Jaká je třída jazyků rozpoznávaných zásobníkovými automaty?

Motivace 2: Je dána bezkontextová gramatika $\mathcal{G}$ a slovo w.
Jak zjistit, zda se slovo w dá vygenerovat v gramatice $\mathcal{G}$?

Problém syntaktické analýzy pro bezkontextové gramatiky:
pro danou bezkontextovou gramatiku $\mathcal{G}$ a slovo w rozhodnout, zda $w \in L(\mathcal{G})$.

Nedeterministická syntaktická analýza shora dolů

Věta 3.47.

Ke každé CFG $\mathcal{G}$ lze sestrojit PDA $\mathcal{M}$ takový, že $L(\mathcal{G}) = L_e(\mathcal{M})$.

Důkaz. K dané gramatice $\mathcal{G}$ konstruujeme PDA $\mathcal{M}$ simulující derivace v $\mathcal{G}$

Nedeterministická syntaktická analýza shora dolů

Věta 3.47.

Ke každé CFG $\mathcal{G}$ lze sestrojit PDA $\mathcal{M}$ takový, že $L(\mathcal{G}) = L_e(\mathcal{M})$.

Důkaz. K dané gramatice $\mathcal{G}$ konstruujeme PDA $\mathcal{M}$ simulující levé derivace v $\mathcal{G}$, tj. vždy rozvíjíme nejlevější neterminál:

- V levé derivaci je v jednom kroku odvození nahrazen (nejlevější) neterminál A pravou stranou $X_1 \dots X_n$ nějakého A-pravidla.
- V $\mathcal{M}$ této situaci odpovídá náhrada A na vrcholu zásobníku řetězem $X_1 \dots X_n$.

$\mathcal{M} = (\{q\}, \Sigma, N \cup \Sigma, \delta, q, S, \emptyset)$, kde δ je definována:

- **(expanze)** $\delta(q, \varepsilon, A)$ obsahuje (q, α) právě když $A \to \alpha \in P$
- **(čtení)** $\delta(q, a, a) = \{(q, \varepsilon)\}$ pro všechna $a \in \Sigma$

Příkad

$$
\begin{array}{lcl}
S & \to & aAB \\
A & \to & Aa \mid \varepsilon \\
B & \to & SaA \mid b
\end{array}
\qquad
\begin{array}{rcl}
\delta(q, \varepsilon, S) & = & \{(q, aAB)\} \\
\delta(q, \varepsilon, A) & = & \{(q, Aa), (q, \varepsilon)\} \\
\delta(q, \varepsilon, B) & = & \{(q, SaA), (q, b)\} \\
\delta(q, a, a) & = & \delta(q, b, b) \;=\; \{(q, \varepsilon)\}
\end{array}
$$

$$
\begin{array}{lcl}
S & \Rightarrow & aAB \\
 & \Rightarrow & aB \\
 & \Rightarrow & aSaA \\
 & \Rightarrow & aaABaA \\
 & \Rightarrow & aaBaA \\
 & \Rightarrow & aabaA \\
 & \Rightarrow & aaba
\end{array}
$$

Korektnost

$$A \Rightarrow^* w \iff (q, w, A) \vdash^* (q, \varepsilon, \varepsilon)$$

($\Longrightarrow$) Indukcí vzhledem k délce odvození m.

- $m = 0$: zřejmé.
- $m \geq 1$: nechť tvrzení platí pro všechna $m' < m$.

  $A \Rightarrow X_1 X_2 \ldots X_k \Rightarrow^* x_1 x_2 \ldots x_k = w$, kde $X_i \overset{m_i}{\Rightarrow} x_i$, $0 \leq m_i < m$ z definice δ plyne $(q, w, A) \vdash (q, w, X_1 X_2 \ldots X_k)$.

  Je-li $X_i \in N$, pak dle indukčního předpokladu máme $(q, x_i, X_i) \vdash^* (q, \varepsilon, \varepsilon)$.

  Je-li $X_i \in \Sigma$, pak $X_i = x_i$ a z definice δ plyne $(q, x_i, x_i) \vdash (q, \varepsilon, \varepsilon)$.

  Kompozicí dostáváme $(q, w, A) \vdash^+ (q, \varepsilon, \varepsilon)$.

($\Longleftarrow$) Předpokládejme $(q, w, A) \vdash^{n} (q, \varepsilon, \varepsilon)$ a ukažme $A \Rightarrow^{*} w$.
Indukcí vzhledem k délce výpočtu n.

- $n = 0$: zřejmé.
- $n \geq 1$: nechť tvrzení platí pro všechna $n' < n$.

  $(q, w, A) \vdash (q, w, X_1 X_2 \ldots X_k)$, tj. $A \to X_1 X_2 \ldots X_k \in P$

  w můžeme napsat jako $w = x_1 x_2 \ldots x_k$ takové, že

  - je-li $X_i \in N$, pak $(q, x_i, X_i) \vdash^{n_i} (q, \varepsilon, \varepsilon)$, kde $n_i < n$.
    Dle IP $X_i \Rightarrow^{+} x_i$.
  - je-li $X_i \in \Sigma$, pak $X_i \overset{0}{\Rightarrow} x_i$.

  Vhodnou kompozicí obdržíme
  $A \Rightarrow X_1 X_2 \ldots X_k \Rightarrow^{*} x_1 X_2 \ldots X_k \Rightarrow^{*} x_1 \ldots x_k = w$
  což je levá derivace slova w v gramatice $\mathcal{G}$. □

Nedeterministická syntaktická analýza zdola nahoru

$$
\begin{aligned}
S &\rightarrow XY \\
X &\rightarrow ab \\
Y &\rightarrow c
\end{aligned}
$$

Nedeterministická syntaktická analýza zdola nahoru

Věta 3.55.

Nechť $\mathcal{G}$ je libovolná CFG, pak lze zkonstruovat rozšířený PDA $\mathcal{R}$ takový, že $L(\mathcal{G}) = L(\mathcal{R})$.

Důkaz. Vrchol zásobníku píšeme vpravo.
Konstruujeme rozšířený PDA $\mathcal{R}$, který simuluje pravou derivaci v $\mathcal{G}$ v obráceném pořadí.
PDA $\mathcal{R}$ má kroky dvojího typu:

1. **(čtení)** může kdykoli načíst do zásobníku symbol ze vstupu
2. **(redukce)** je-li na vrcholu zásobníku řetězec tvořící pravou stranu nějakého pravidla v $\mathcal{G}$, může ho nahradit odpovídajícím levostranným neterminálem (a ze vstupu nic nečte)

Nedeterministická syntaktická analýza zdola nahoru

Věta 3.55.

Nechť $\mathcal{G}$ je libovolná CFG, pak lze zkonstruovat rozšířený PDA $\mathcal{R}$ takový, že $L(\mathcal{G}) = L(\mathcal{R})$.

Důkaz. Vrchol zásobníku píšeme vpravo.
Konstruujeme rozšířený PDA $\mathcal{R}$, který simuluje pravou derivaci v $\mathcal{G}$ v obráceném pořadí.
PDA $\mathcal{R}$ má kroky dvojího typu:

1. **(čtení)** může kdykoli načíst do zásobníku symbol ze vstupu
2. **(redukce)** je-li na vrcholu zásobníku řetězec tvořící pravou stranu nějakého pravidla v $\mathcal{G}$, může ho nahradit odpovídajícím levostranným neterminálem (a ze vstupu nic nečte)

Nechť $\mathcal{G} = (N, \Sigma, P, S)$.
Položme $\mathcal{R} = (\{q, r\}, \Sigma, N \cup \Sigma \cup \{\bot\}, \delta, q, \bot, \{r\})$, kde $\bot$ je nově přidaný symbol a kde δ je definována takto:

1. $\delta(q, a, \varepsilon) = \{(q, a)\}$ pro všechna $a \in \Sigma$
2. je-li $A \to \alpha$ pravidlo v P, pak $\delta(q, \varepsilon, \alpha)$ obsahuje (q, A)
3. $\delta(q, \varepsilon, \bot S) = \{(r, \varepsilon)\}$

Příklad

	krok výpočtu	odpovídající pravidlo z $\mathcal{G}$
$(q,\ i+i*i,\ \bot)$	$\overset{i}{\vdash}\ (q,\ +i*i,\ \bot i)$	$F \to i$
	$\overset{\varepsilon}{\vdash}\ (q,\ +i*i,\ \bot F)$	$T \to F$
	$\overset{\varepsilon}{\vdash}\ (q,\ +i*i,\ \bot T)$	$E \to T$
	$\overset{\varepsilon}{\vdash}\ (q,\ +i*i,\ \bot E)$	
	$\overset{+}{\vdash}\ (q,\ \ i*i,\ \bot E+)$	
	$\overset{i}{\vdash}\ (q,\ \ *i,\ \bot E+i)$	$F \to i$
	$\overset{\varepsilon}{\vdash}\ (q,\ \ *i,\ \bot E+F)$	$T \to F$
	$\overset{\varepsilon}{\vdash}\ (q,\ \ *i,\ \bot E+T)$	
	$\overset{*}{\vdash}\ (q,\ \ i,\ \bot E+T*)$	
	$\overset{i}{\vdash}\ (q,\ \ \varepsilon,\ \bot E+T*i)$	$F \to i$
	$\overset{\varepsilon}{\vdash}\ (q,\ \ \varepsilon,\ \bot E+T*F)$	$T \to T*F$
	$\overset{\varepsilon}{\vdash}\ (q,\ \ \varepsilon,\ \bot E+T)$	$E \to E+T$
	$\overset{\varepsilon}{\vdash}\ (q,\ \ \varepsilon,\ \bot E)$	
	$\overset{\varepsilon}{\vdash}\ (r,\ \ \varepsilon,\ \varepsilon)$	

Korektnost

$$S \Rightarrow^* \alpha A y \overset{n}{\Rightarrow} xy \iff (q, xy, \bot) \vdash^* (q, y, \bot\alpha A),$$

kde $S \Rightarrow^* \alpha A y \overset{n}{\Rightarrow} xy$ je pravá derivace a A je nejpravější neterminál.
$(\Longrightarrow)$ indukcí k délce odvození
$(\Longleftarrow)$ indukcí k délce výpočtu

Pro $A = S$ a $\alpha, y = \varepsilon$ dostáváme:

$$S \Rightarrow^* x \iff (q, x, \bot) \vdash^* (q, \varepsilon, \bot S) \quad \left[\vdash (r, \varepsilon) \right]$$

□

Efektivnost syntaktické analýzy

Nederministický PDA $\Longrightarrow$ nedeterministický algoritmus
$\Longrightarrow$ exponenciální deterministický algoritmus

Řešení:

- deterministický algoritmus složitosti $\mathcal{O}(n^3)$, kde $n = |w|$ (algoritmus Cocke - Younger - Kasami)
- deterministické zásobníkové automaty a deterministické bezkontextové jazyky
- lineární algoritmy pro speciální třídy deterministických bezkontextových jazyků

Kanonické tvary bezkontextových gramatik

- redukované bezkontextové gramatiky
- gramatiky bez ε-pravidel
- gramatiky bez jednoduchých pravidel
- vlastní gramatiky
- Chomského normální forma
- gramatiky bez levé rekurze
- Greibachové normální forma

Rekurzivní neterminály a gramatiky

Definice 3.28.

Neterminál A v CFG $\mathcal{G} = (N, \Sigma, P, S)$ se nazývá levorekurzivní jestliže v $\mathcal{G}$ existuje derivace $A \Rightarrow^+ A\beta$.

CFG bez levorekurzivních neterminálů se nazývá nelevorekurzivní.

Je-li v CFG pravidlo tvaru $A \to A\alpha$, hovoříme o přímé levé rekurzi na A.

Praktický význam: některé nástroje pro automatickou tvorbu parserů k zadaným gramatikám vyžadují na vstupu nelevorekurzivní gramatiku (např. ANTLR).

Algoritmus odstranění přímé levé rekurze

Nechť CFG $\mathcal{G} = (N, \Sigma, P, S)$ je necyklická a bez ε-pravidel, v níž všechna A-pravidla (pravidla mající na levé straně A) jsou tvaru

$$A \to A\alpha_1 \mid \ldots \mid A\alpha_m \mid \beta_1 \mid \ldots \mid \beta_n,$$

kde každý řetěz β_i začíná symbolem různým od A.

Algoritmus odstranění přímé levé rekurze

Nechť CFG $\mathcal{G} = (N, \Sigma, P, S)$ je necyklická a bez ε-pravidel, v níž všechna A-pravidla (pravidla mající na levé straně A) jsou tvaru

$$A \to A\alpha_1 \mid \ldots \mid A\alpha_m \mid \beta_1 \mid \ldots \mid \beta_n,$$

kde každý řetěz β_i začíná symbolem různým od A.

Nechť $\mathcal{G}' = (N \cup \{A'\}, \Sigma, P', S)$, kde P' obdržíme z P tak, že všechna výše uvedená pravidla nahradíme pravidly:

$$A \to \beta_1 \mid \ldots \mid \beta_n \mid \beta_1 A' \mid \ldots \mid \beta_n A'$$

$$A' \to \alpha_1 \mid \ldots \mid \alpha_m \mid \alpha_1 A' \mid \ldots \mid \alpha_m A'$$

Algoritmus odstranění přímé levé rekurze

Nechť CFG $\mathcal{G} = (N, \Sigma, P, S)$ je necyklická a bez ε-pravidel, v níž všechna A-pravidla (pravidla mající na levé straně A) jsou tvaru

$$A \to A\alpha_1 \mid \ldots \mid A\alpha_m \mid \beta_1 \mid \ldots \mid \beta_n,$$

kde každý řetěz β_i začíná symbolem různým od A.

Nechť $\mathcal{G}' = (N \cup \{A'\}, \Sigma, P', S)$, kde P' obdržíme z P tak, že všechna výše uvedená pravidla nahradíme pravidly:

$$A \to \beta_1 \mid \ldots \mid \beta_n \mid \beta_1 A' \mid \ldots \mid \beta_n A'$$

$$A' \to \alpha_1 \mid \ldots \mid \alpha_m \mid \alpha_1 A' \mid \ldots \mid \alpha_m A'$$

Pak $L(\mathcal{G}) = L(\mathcal{G}')$ a $\mathcal{G}'$ je necyklická a bez ε-pravidel.

Lemma o substituci

Lemma 3.20. (o substituci)

Nechť $\mathcal{G} = (N, \Sigma, P, S)$ je CFG. Nechť $A \to \alpha_1 B \alpha_2 \in P$.
Nechť $B \to \beta_1 \mid \ldots \mid \beta_r$ jsou všechna pravidla v P tvaru $B \to \alpha$.
Definujme $\mathcal{G}' = (N, \Sigma, P', S)$, kde

$$P' = (P \setminus \{A \to \alpha_1 B \alpha_2\}) \cup \{A \to \alpha_1 \beta_1 \alpha_2 \mid \ldots \mid \alpha_1 \beta_r \alpha_2\}.$$

Pak $L(\mathcal{G}) = L(\mathcal{G}')$.

Lemma o substituci

Lemma 3.20. (o substituci)

Nechť $\mathcal{G} = (N, \Sigma, P, S)$ je CFG. Nechť $A \to \alpha_1 B \alpha_2 \in P$.
Nechť $B \to \beta_1 \mid \ldots \mid \beta_r$ jsou všechna pravidla v P tvaru $B \to \alpha$.
Definujme $\mathcal{G}' = (N, \Sigma, P', S)$, kde

$$P' = (P \setminus \{A \to \alpha_1 B \alpha_2\}) \cup \{A \to \alpha_1 \beta_1 \alpha_2 \mid \ldots \mid \alpha_1 \beta_r \alpha_2\}.$$

Pak $L(\mathcal{G}) = L(\mathcal{G}')$.

Příklad:

$A \to Bd \mid c$
$B \to Bdd \mid Ccc \mid aAd$
$C \to Aa$

Algoritmus odstranění levé rekurze

Vstup: Vlastní CFG $\mathcal{G} = (N, \Sigma, P, S)$
Výstup: Ekvivalentní nelevorekurzivní gramatika bez ε-pravidel

1: Uspořádej libovolně N, $N = \{A_1, \ldots, A_n\}$
2: **for** $i \leftarrow 1$ **to** n **do**
3: **for** $j \leftarrow 1$ **to** $i - 1$ **do**
4: **for all** pravidlo tvaru $A_i \to A_j\alpha$ **do**
5: přidej pravidla $A_i \to \beta_1\alpha \mid \ldots \mid \beta_k\alpha$
6: (kde $A_j \to \beta_1 \mid \ldots \mid \beta_k$ jsou všechna pravidla pro A_j)
7: vypusť pravidlo $A_i \to A_j\alpha$
8: odstraň případnou přímou levou rekurzi na A_i

Algoritmus odstranění levé rekurze

Vstup: Vlastní CFG $\mathcal{G} = (N, \Sigma, P, S)$
Výstup: Ekvivalentní nelevorekurzivní gramatika bez ε-pravidel

1: Uspořádej libovolně N, $N = \{A_1, \ldots, A_n\}$
2: **for** $i \leftarrow 1$ **to** n **do**
3: **for** $j \leftarrow 1$ **to** $i - 1$ **do**
4: **for all** pravidlo tvaru $A_i \to A_j\alpha$ **do**
5: přidej pravidla $A_i \to \beta_1\alpha \mid \ldots \mid \beta_k\alpha$
6: (kde $A_j \to \beta_1 \mid \ldots \mid \beta_k$ jsou všechna pravidla pro A_j)
7: vypusť pravidlo $A_i \to A_j\alpha$
8: odstraň případnou přímou levou rekurzi na A_i

Příklad:

$A \to Bd \mid c$

$B \to Bdd \mid Ccc \mid aAd$

$C \to Aa$

Korektnost algoritmu

Konečnost.

Ekvivalence gramatik: Všechny úpravy jsou dle Lemmatu o substituci nebo odstraňují přímou levou rekurzi.

Výsledná gramatika je nelevorekurzivní:

1. po i-té iteraci vnějšího cyklu začíná každé A_i-pravidlo buď terminálem nebo neterminálem A_k, kde $k > i$.
2. po j-té iteraci vnitřního cyklu začíná každé A_i-pravidlo buď terminálem nebo neterminálem A_k, kde $k > j$.

Výsledná gramatika je bez ε-pravidel.

Greibachové normální forma

Definice 3.33.

Bezkontextová gramatika $\mathcal{G} = (N, \Sigma, P, S)$ je v **Greibachové normální formě** (GNF), právě když

- $\mathcal{G}$ je bez ε-pravidel a
- každé pravidlo z P je tvaru $A \to a\alpha$, kde $a \in \Sigma$ a $\alpha \in N^*$ (s případnou vyjímkou pravidla $S \to \varepsilon$).

Věta 3.34.

Každý bezkontextový jazyk lze generovat bezkontextovou gramatikou v Greibachové normální formě.

Důkaz.

$L = \emptyset$: zřejmé $(S \to aS)$

$L \neq \emptyset$:
1. z vlastní gramatiky eliminujeme levou rekurzi
2. pak převedeme do GNF

□

Příklad

$$
\begin{aligned}
A &\to Ba \mid Db \mid c \\
B &\to CC \\
C &\to aE \\
D &\to CDa \mid Eb \\
E &\to bb
\end{aligned}
$$

Algoritmus transformace do GNF

Vstup: Nelevorekurzivní CFG $\mathcal{G} = (N, \Sigma, P, S)$ bez ε-pravidel
Výstup: Ekvivalentní gramatika v GNF

1: Najdi lineární uspořádání $\prec$ splňující $(A \to B\alpha) \in P \implies A \prec B$
2: Označme $N = \{A_1, \ldots, A_n \mid A_{i-1} \prec A_i, 1 < i \leq n\}$
3: **for** $i \leftarrow n-1$ **downto** 1 **do**
4: **for all** pravidlo tvaru $A_i \to A_j\alpha$, kde $j > i$ **do**
5: přidej pravidlo $A_i \to \beta_1\alpha \mid \ldots \mid \beta_k\alpha$
6: (kde $A_j \to \beta_1 \mid \ldots \mid \beta_k$ jsou všechna A_j-pravidla)
7: vypusť pravidlo $A_i \to A_j\alpha$
8: Nahraď potřebné terminály novými neterminály a přidej příslušná pravidla

Korektnost algoritmu

Konečnost.

Ekvivalence gramatik.

Výsledná gramatika je v GNF.