Automaty a gramatiky 6

Roman Barták, KTIML

bartak@ktiml.mff.cuni.cz
http://ktiml.mff.cuni.cz/~bartak

Co bylo minule

Automaty s výstupem
- Mooreovův stroj (značkuje stavy)
- Mealyho stroj (vypisuje přechody)
- převody mezi stroji

Iterační (pumping) lemma

Úvod do formálních gramatik
- přepisovací systémy (abeceda + pravidla $u \to v$)
- gramatiky (neterminály (start) + terminály + pravidla)
- Chomského hierarchie (dle tvaru pravidel)

  jazyky typu 0,1,2,3 - značíme $\mathcal{L}_0$, $\mathcal{L}_1$, $\mathcal{L}_2$, $\mathcal{L}_3$

Uspořádanost Chomského hierarchie

Chomského hierarchie definuje uspořádání tříd jazyků:

$$\mathcal{L}_0 \supseteq \mathcal{L}_1 \supseteq \mathcal{L}_2 \supseteq \mathcal{L}_3$$

Dokonce vlastní podmnožiny (později):

$$\mathcal{L}_0 \supset \mathcal{L}_1 \supset \mathcal{L}_2 \supset \mathcal{L}_3$$

$\mathcal{L}_0 \supseteq \mathcal{L}_1$ (rekurzivně spočetné jazyky zahrnují kontextové jazyky)

obecná pravidla $\supseteq$ pravidla tvaru $\alpha X\beta \to \alpha w\beta$

$\mathcal{L}_2 \supseteq \mathcal{L}_3$ (bezkontextové jazyky zahrnují regulární jazyky)

„$X \to w$, $w \in (V_N \cup V_T)^*$ “ $\supseteq$ „$X \to wY$, $X \to w$, $Y \in V_N$, $w \in V_T^*$ “

$\mathcal{L}_1 \supseteq \mathcal{L}_2$ (kontextové jazyky zahrnují bezkontextové jazyky)

$\alpha X\beta \to \alpha w\beta$, $|w|>0$ vs. $X \to w$ $|w| \geq 0$

problém s pravidly tvaru $X \to \lambda$

Mužeme z bezkontextových gramatik vyřadit pravidla $X \to \lambda$?

Nevypouštějící bezkontextové gramatiky

Bezkontextová gramatika G je *nevypouštějící* právě tehdy, když nemá pravidla ve tvaru $X \to \lambda$.

Věta: Ke každé bezkontextové gramatice G existuje nevypouštějící bezkontextová gramatika G_1 taková, že $L(G_1) = L(G) - \{\lambda\}$ (jazyky se liší maximálně o prázdné slovo).

Je-li $\lambda \in L(G)$, potom existuje BKG G_2 tak, že $L(G_2) = L(G)$ a jediné pravidlo s λ na pravé straně je $S' \to \lambda$ a S' (počáteční neterminál) se nevyskytuje na pravé straně žádného pravidla G_2 (tedy $\mathcal{L}_1 \supseteq \mathcal{L}_2$).

Příklad:

G: $S \to 0S1 \mid \lambda$

G_1: $S \to 0S1 \mid 01$

G_2: $S' \to S \mid \lambda$, $S \to 0S1 \mid 01$

Převod na nevypouštějící BKG

... aneb, jak se zbavit pravidel ve tvaru $X \to \lambda$?

Základní myšlenka:

- pravidlo $X \to \lambda$ se používá pro vyhození X ze slova
- co když X do slova vůbec nezařadíme?

..., $Y \to uXv$, $X \to \lambda$, ... $\Rightarrow$ ..., $Y \to uv$, ...

1) Nejprve zjistíme neterminály, které se přepisují na λ:

$U = \{X \mid X \in V_N \& X \Rightarrow^* \lambda\}$

Proč tak silně (nestačilo by $X \to \lambda$ místo $X \Rightarrow^* \lambda$)?

Řešení derivací $X \Rightarrow^{X \to Y} Y \Rightarrow^{Y \to Z} Z \Rightarrow^{Z \to \lambda} \lambda$

Iterační algoritmus pro získání U:

$U_1 = \{X \mid X \in V_N \& (X \to \lambda) \in P\}$ přímý přepis

$U_{i+1} = \{X \mid X \in V_N \& (X \to w) \in P, w \in U_i^*\}$ přepis po i+1 krocích

$U_1 \subseteq U_2 \subseteq \ldots \subseteq V_N$ + stabilizace ($\exists k\ U_k = U_{k+1} = \ldots$) + $U = U_k$

Převod na nevypouštějící BKG - pokračování

2) Úprava pravidel

do P_1 dáme pravidla tvaru $X \to u$ taková, že:

- $u \neq \lambda$
- v P je pravidlo $X \to v_1Y_1v_2 \ldots v_mY_mv_{m+1}$, $Y_i \in U$, $v_i \in ((V_N - U) \cup V_T)^*$ a u vzniká z $(v_1Y_1v_2 \ldots v_mY_mv_{m+1})$ vypuštěním některých (všech, žádného) symbolů Y_i.

3) Ještě $L(G_1) = L(G) - \{\lambda\}$

zřejmé: G_1 je nevypouštějící BKG, $L(G_1) \subseteq L(G)$, $\lambda \notin L(G_1)$

nechť $w \in L(G)$ a $w \neq \lambda$, tj. $S \Rightarrow^* w$,

pokud se použilo pravidlo z $P\text{-}P_1$, pak má tvar $X \to \lambda$

v derivaci před ním muselo být užito pravidlo $Y \to uXv$

uděláme novou derivaci s $Y \to uv$ a bez $X \to \lambda$

4) Zbývá situace $\lambda \in L(G)$

$G_2 = (V_N \cup \{S'\}, V_T, S', P_1 \cup \{S' \to \lambda, S' \to S\})$

Příklad - nevypouštějící BKG

$S \to aSc \mid A$
$A \to bAc \mid \lambda$

1) Nejprve zjistíme neterminály, které se přepisují na λ:
   $U = \{A,S\}$

2) Upravíme pravidla:
   $S \to aSc \mid A$
   $S \to ac$ (vzniklo z $S \to aSc$ vypuštěním S)
   $A \to bAc$ (pravidlo $A \to \lambda$ nepřevádíme)
   $A \to bc$ (vzniklo z $A \to bAc$ vypuštěním A)

Původní gramatika přijímá jazyk $\{a^i b^j c^k \mid i+j=k\}$.
Převedená gramatika přijímá jazyk $\{a^i b^j c^k \mid i+j=k, k>0\}$.

Gramatiky typu 3 a regulární jazyky

pouze pravidla ve tvaru $X \to wY$, $X \to w$, $X,Y \in V_N$, $w \in V_T^*$

Podívejme se na derivace generované gramatikami typu 3
P: $S \to 0S \mid 1A \mid \lambda$, $A \to 0A \mid 1B$, $B \to 0B \mid 1S$
$S \Rightarrow 0S \Rightarrow 01A \Rightarrow 011B \Rightarrow 0110B \Rightarrow 01101S \Rightarrow 01101$

Pozorování:
- každé slovo derivace obsahuje právě jeden neterminál
- tento neterminál je vždy umístěn zcela vpravo
- aplikací pravidla $X \to w$ se derivace uzavírá
- krok derivace = generuje symbol(y) +změní neterminál

Idea převodu gramatiky na automat:
neterminál = stav konečného automatu
pravidla = přechodová funkce

Převod konečného automatu na gramatiku

$L \in \mathcal{F} \Rightarrow L \in \mathcal{L}_3$

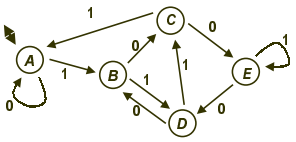
Důkaz:
L=L(A) pro nějaký konečný automat $A=(Q,X,\delta,q_0,F)$
definujme gramatiku $G=(Q,X,q_0,P)$, kde pravidla mají tvar
$p \to aq$, když $\delta(p,a)=q$
$p \to \lambda$, když $p \in F$
ještě L(A)=L(G)?
1) $\lambda \in L(A) \Leftrightarrow q_0 \in F \Leftrightarrow (q_0 \to \lambda) \in P \Leftrightarrow \lambda \in L(G)$
2) $a_1 \ldots a_n \in L(A) \Leftrightarrow \exists q_0,\ldots,q_n \in Q$ tž. $\delta(q_i,a_{i+1})=q_{i+1}$, $q_n \in F$
$\Leftrightarrow (q_0 \Rightarrow a_1 q_1 \Rightarrow \ldots \Rightarrow a_1 \ldots a_n q_n \Rightarrow a_1 \ldots a_n)$ je derivace pro $a_1 \ldots a_n$
$\Leftrightarrow a_1 \ldots a_n \in L(G)$

QED

A co naopak?
- pravidla $X \to aY$ kódujeme do přechodové funkce a $X \to \lambda$ je konec
- ale co pravidla $X \to a_1 \ldots a_n Y$, $X \to Y$, $X \to a_1 \ldots a_n$?

Příklad převodu KA na gramatiku

$A \to 1B \mid 0A \mid \lambda$
$B \to 0C \mid 1D$
$C \to 0E \mid 1A$
$D \to 0B \mid 1C$
$E \to 0D \mid 1E$



Příklady derivací:
$A \Rightarrow 0A \Rightarrow 0$ (0)
$A \Rightarrow 1B \Rightarrow 10C \Rightarrow 101A$ (5)
$A \Rightarrow 1B \Rightarrow 10C \Rightarrow 101A \Rightarrow 1010A \Rightarrow 1010$ (10)
$A \Rightarrow 1B \Rightarrow 11D \Rightarrow 111C \Rightarrow 1111A \Rightarrow 1111$ (15)

L= { w | w ∈{0,1}* & w je binární zápis čísla dělitelného 5}

Standardizace pravidel regulární gramatiky

Ke každé gramatice $G=(V_N,V_T,S,P)$ typu 3 existuje ekvivalentní gramatika G', které obsahuje pouze pravidla ve tvaru: $X \to aY$ a $X \to \lambda$.

Důkaz:
definujme $G'=(V'_N,V_T,S,P')$, kde pravidla P' získáme takto

P	P'
$X \to aY$	$X \to aY$
$X \to \lambda$	$X \to \lambda$
$X \to a_1 \ldots a_n Y$	$X \to a_1 Y_2$, $Y_2 \to a_2 Y_3$, …, $Y_n \to a_n Y$
$Z \to a_1 \ldots a_n$	$Z \to a_1 Z_1$, $Z_1 \to a_2 Z_2$, …, $Z_n \to \lambda$

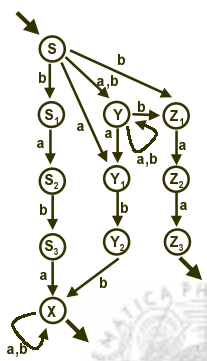
($Y_2,\ldots,Y_n$, $Z_1,\ldots,Z_n$ jsou nové neterminály - pro každé pravidlo jiná sada)

zbývá $X \to Y$
definujme $U(X) = \{Y \mid Y \in V_N \& X \Rightarrow^* Y\}$
efektivní postup $U_1 = \{Y \mid (X \to Y) \in P\}$, $U_{i+1} = U_i \cup \{Y \mid (Z \to Y) \in P, Z \in U_i\}$

(X → Y)	X→ w pro všechna Y→ w z P' a Y∈U(X)

Příklad standardizace regulární gramatiky

Originální	*Převedená*
$S \to babaX$	$S \to bS_1$ $S_1 \to aS_2$ $S_2 \to bS_3$ $S_3 \to aX$
$S \to Y$	$S \to aY \mid bY \mid aY_1 \mid bZ_1$
$Y \to aY \mid bY$	$Y \to aY \mid bY$
$Y \to abbX$	$Y \to aY_1$ $Y_1 \to bY_2$ $Y_2 \to bX$
$Y \to baa$	$Y \to bZ_1$ $Z_1 \to aZ_2$ $Z_2 \to aZ_3$ $Z_3 \to \lambda$
$X \to aX \mid bX \mid \lambda$	$X \to aX \mid bX \mid \lambda$



L={ w | w=babau ∨ w=uabbv ∨ w=ubaa, u,v∈{a,b}* }

Převod gramatiky na konečný automat

$L \in \mathcal{L}_3 \Rightarrow L \in \mathcal{F}$

Důkaz:

$L=L(G)$ pro nějakou gramatiku $G=(V_N,V_T,S,P)$ typu 3 obsahující pouze pravidla ve tvaru: $X \to aY$ a $X \to \lambda$

definujme nedeterministický konečný automat $A=(V_N,V_T,\delta,\{S\},F)$, kde:

$F=\{ X \mid (X \to \lambda) \in P\}$

$\delta(X,a) =\{ Y \mid (X \to aY) \in P\}$

ještě $L(G)=L(A)$?

1) $\lambda \in L(G) \Leftrightarrow (S \to \lambda) \in P \Leftrightarrow S \in F \Leftrightarrow \lambda \in L(A)$

2) $a_1 \dots a_n \in L(G)$

$\Leftrightarrow$ existuje derivace $(S \Rightarrow a_1X_1 \Rightarrow \dots \Rightarrow a_1 \dots a_nX_n \Rightarrow a_1 \dots a_n)$

$\Leftrightarrow \exists X_0,\dots,X_n \in V_N$ tž. $\delta(X_i,a_{i+1}) \ni X_{i+1}$, $X_0=S$, $X_n \in F \Leftrightarrow a_1 \dots a_n \in L(A)$

Levé (a pravé) lineární gramatiky

Gramatiky typu 3 nazýváme také *pravé lineární* (neterminál je vždy vpravo).

Obdobně - gramatika G je *levá lineární*, jestliže má pouze pravidla tvaru $X \to Yw$, $X \to w$, $X,Y \in V_N$, $w \in V_T^*$ (neterminál je vždy vlevo).

Věta: Jazyky generované levou lineární gramatikou jsou právě regulární jazyky.

Důkaz:

- otočením pravidel dostaneme pravou lineární gramatiku $X \to Yw$, $X \to w$ převedeme na $X \to w^RY$, $X \to w^R$
- získaná gramatika generuje jazyk L^R
- víme, že regulární jazyky jsou uzavřené na reverzi tudíž protože L^R je regulární, je i L $(=(L^R)^R)$ regulární
- takto lze získat všechny regulární jazyky

Lineární gramatiky (a jazyky)

Můžeme levě a právě lineární pravidla používat najednou?

Další zobecnění - *gramatika je lineární*, jestliže má pouze pravidla tvaru $X \to uYv$, $X \to w$, $X,Y \in V_N$, $u,v,w \in V_T^*$ (na pravé straně vždy maximálně jeden neterminál).

Lineární jazyky jsou právě jazyky generované lineárními gramatikami.

Zřejmě: regulární jazyky $\subseteq$ lineární jazyky

Platí také: regulární jazyky $\supseteq$ lineární jazyky?

NE!

$\{0^n1^n \mid n \geq 1\}$ není regulární jazyk, ale je lineární $(S \to 0S1 \mid 01)$

Pozorování:

lineární pravidla lze rozložit na levě a pravě lineární pravidla $S \to 0A$, $A \to S1$

Bezkontextové gramatiky

pouze pravidla ve tvaru $X \to w$, $X \in V_N$, $w \in (V_N \cup V_T)^*$

velký praktický význam

- definování syntaxe vyšších programovacích jazyků
- konstrukce kompilátorů

Příklad:

```
<Program> → <Příkaz> | <Příkaz> <Program>
<Příkaz> → <IF-příkaz> | <WHILE-příkaz> | <Přiřazení>
<IF-příkaz> → if <Test> then <Příkaz> else <Příkaz>
<WHILE-příkaz> → while <Test> do <Příkaz>
<Přiřazení> → <Proměnná> := <Výraz>
```

Bude nás zajímat:

- jednoznačnost gramatiky (kvůli překladu)
- analýza pomocí zásobníkových automatů
- vlastnosti BKG obecně

Redukované bezkontextové gramatiky

Redukce (gramatiky) = vyřazení zbytečností

u konečných automatů:

- dosažitelné stavy, které nejsou ekvivalentní
- redukt určen jednoznačně

u bezkontextových gramatik:

- dosažitelné neterminály, které něco generují
- zde slabší význam (nemáme jednoznačnost)

Bezkontextová gramatika G (taková, že $L(G) \neq \emptyset$) se nazývá *redukovaná*, jestliže:

1) pro každý neterminál X existuje alespoň jedno terminální slovo w takové, že $X \Rightarrow^* w$
2) pro každý neterminál X různý od S existují slova u, v tak, že $S \Rightarrow^* uXv$ (dosažitelnost).

Redukce bezkontextových gramatik

Příklad:

$S \to aA \mid ab$	Zjevně stačí pravidlo
$A \to BC$	$S \to ab$,
$B \to ba$	ostatní pravidla (neterminály) jsou
$D \to ab \mid \lambda$	zbytečná

Tvrzení: Ke každé bezkontextové gramatice G takové, že $L(G) \neq \emptyset$ lze sestrojit ekvivalentní redukovanou gramatiku.

Důkaz (idea):

1) vyhoď neterminály, které negenerují terminální slovo
2) vyhoď nedosažitelné neterminály

Poznámka: pořadí akcí nelze prohodit!

$S \to ab \mid A$
$A \to BC$
$B \to b$

~~$S \to ab$~~
~~$B \to b$~~

Algoritmus redukce - krok 1

hledáme $V = \{X \mid X \in V_N, \exists w \in V_T^* : X \Rightarrow^* w\}$

obvyklý postup (iterace po krocích):

$V_0 = V_T$

$V_{i+1} = V_i \cup \{X \mid X \in V_N, \exists w \in V_i^* : (X \rightarrow w) \in P\}$

$V_0 \subseteq V_1 \subseteq \ldots \subseteq V_T \cup V_N$ + stabilizace $(\exists k : V_k = V_{k+1} = \ldots) + V = V_k \cap V_N$

Zároveň víme, zda $L(G) \neq \emptyset$ $(L(G) \neq \emptyset \Leftrightarrow S \in V)$

Nyní z gramatiky odstraníme všechna pravidla obsahující na levé či pravé straně neterminál nepatřící do V.

a) máme splněn bod 1) definice redukované gramatiky

pro $X \in V$ víme: $\exists w \in V_T^* : X \Rightarrow^* w$ + použitá pravidla nebyla odstraněna

b) získaná gramatika G' je ekvivalentní s původní gramatikou G

$L(G') \subseteq L(G)$ zřejmé , $L(G') \supseteq L(G)$ lze ukázat sporem

Příklad:

$S \rightarrow aA \mid ab,\quad A \rightarrow BC,\quad B \rightarrow ba,\quad D \rightarrow ab \mid \lambda$

$V = \{S,B,D\}$

redukovaná pravidla: $S \rightarrow ab,\quad B \rightarrow ba,\quad D \rightarrow ab \mid \lambda$

Algoritmus redukce - krok 2 (dosažitelnost)

hledáme $U = \{X \mid X \in V_N, S \Rightarrow^* uXv\}$

obvyklý postup (iterace po krocích):

$U_0 = \{S\}$

$U_{i+1} = U_i \cup \{X \mid X \in V_N, \exists Y \in U_i : (Y \rightarrow uXv) \in P\}$

$U_0 \subseteq U_1 \subseteq \ldots \subseteq V_N$ + stabilizace $(\exists k : U_k = U_{k+1} = \ldots) + U = U_k$

Podobně jako v kroku 1 odstraníme z gramatiky všechna pravidla obsahující na levé či pravé straně neterminál nepatřící do U.

a) máme splněn bod 2) definice redukované gramatiky

pro $X \in U$ víme: $S \Rightarrow^* uXv$ + použitá pravidla nebyla odstraněna

b) získaná gramatika G'' je ekvivalentní s gramatikou G'

$L(G'') \subseteq L(G')$ zřejmé , $L(G'') \supseteq L(G')$ lze ukázat sporem

c) platnost bodu 1) definice redukované gramatiky nebyla narušena

spojením $X \Rightarrow^* w$ a $S \Rightarrow^* uXv$

Příklad:

$S \rightarrow ab,\quad B \rightarrow ba,\quad D \rightarrow ab \mid \lambda$ $\qquad U = \{S\}$

finální redukovaná pravidla: $S \rightarrow ab$