Dnešní téma

Gramatiky, Chomského hierarchie, víceznačnost

Marta Vomlelová

MS 303

Palindromy

Definition (palindrom)

Palindrom je řetězec w stejný při čtení zepředu zedadu, tj. $w = w^R$.

- Příklady: `'otto'`; `'Madam, I'm Adam'`.

Lemma

Jazyk $L_{pal} = \{w | w = w^R, w \in \Sigma^*\}$ není regulární.

Example 5.1 (Gramatika)

$G = (\{S\}, \{o, t\}, P, S)$, $P =$

1. $S \to \epsilon$
2. $S \to o$
3. $S \to t$
4. $S \to oSo$
5. $S \to tSt$

Proof:

- Důkaz sporem. Předpokládejme L_{pal} je regulární, nechť n je konstanta z pumping lemma, uvažujme slovo: $w = o^n t o^n$.
- z pumping lemmatu lze rozložit na $w = xyz$, y obsahuje jednu nebo více z prvních n o-ček. Tedy xz má být v L_{pal} ale má méně o-ček vlevo od 't' než vpravo, tedy není palindrom. Iterační lemma pro jazyk L_{pal} nedokáže najít n, proto L_{pal} není regulární. □

Formální (generativní) gramatiky, Bezkontextové gramatiky

Definition 5.1 (Formální (generativní) gramatika)

Formální (generativní) gramatika je $G = (V, T, P, S)$ složena z

- konečné množiny **neterminálů** (variables) V
- neprázdné konečné množiny **terminálních symbolů** (**terminálů**) T
- **počáteční symbol** $S \in V$.
- konečné množiny **pravidel** (**produkcí**) P reprezentující rekurzivní definici jazyka. Každé pravidlo má tvar:
  - $\beta A \gamma \to \omega,\ A \in V, \beta, \gamma, \omega \in (V \cup T)^*$

    tj. levá strana obsahuje aspoň jeden neterminální symbol.

Definition (Bezkontextová gramatika CFG)

Bezkontextová gramatika (CFG) je $G = (V, T, P, S)$ gramatika, obsahující pouze pravidla tvaru

$$A \to \omega,\ A \in V, \omega \in (V \cup T)^*.$$

Chomského hierarchie

Definition 5.2 (Klasifikace gramatik podle tvaru přepisovacích pravidel)

- **gramatiky typu 0** (**rekurzivně spočetné jazyky** $\mathcal{L}_0$)

  pravidla v obecné formě $\alpha \to \omega$, $\alpha, \omega \in (V \cup T)^*, \alpha$ obsahuje neterminál
- **gramatiky typu 1** (**kontextové gramatiky**, jazyky $\mathcal{L}_1$)
  - pouze pravidla ve tvaru $\gamma A \beta \to \gamma \omega \beta$

    $A \in V, \gamma, \beta \in (V \cup T)^*, \omega \in (V \cup T)^+$**!**
  - jedinou výjimkou je pravidlo $S \to \epsilon$, potom se ale S nevyskytuje na pravé straně žádného pravidla
- **gramatiky typu 2** (**bezkontextové gramatiky**, jazyky $\mathcal{L}_2$)

  pouze pravidla ve tvaru $A \to \omega, A \in V, \omega \in (V \cup T)^*$
- **gramatiky typu 3** (regulární/**pravé lineární gramatiky**, regulární jazyky $\mathcal{L}_3$)

  pouze pravidla ve tvaru $A \to \omega B, A \to \omega, A, B \in V, \omega \in T^*$.

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)
  $\mathcal{L}_0 \supseteq \mathcal{L}_1$ rekurzivně spočetné jazyky zahrnují kontextové jazyky
  pravidla $\gamma A \beta \to \gamma \omega \beta$ obsahují vlevo neterminál A
  $\mathcal{L}_2 \supseteq \mathcal{L}_3$ bezkontextové jazyky zahrnují regulární jazyky
  pravidla $A \to \omega B, A \to \omega$ obsahují vpravo řetězec $(V \cup T)^*$
  $\mathcal{L}_1 \supseteq \mathcal{L}_2$ kontextové jazyky zahrnují bezkontextové jazyky
  problém je s pravidly typu $A \to \epsilon$, ale ta umíme eliminovat.

Example 5.2 (Notace)

$a, b, c, 1, *, ($	terminály
A, B, C	neterminály, proměnné
w, z	řetězec terminálů
X, Y	buď terminál nebo neterminál
α, β, γ	řetězec $(T \cup V)^*$
$A \to \alpha \mid \beta$	$\{A \to \alpha, A \to \beta\}$, OR, kompaktní zápis více pravidel.

Definition 5.3 (Derivace $\Rightarrow^*$)

Mějme gramatiku $G = (V, T, P, S)$.

- Říkáme, že α se **přímo přepíše** na ω (píšeme $\alpha \Rightarrow_G \omega$ nebo $\alpha \Rightarrow \omega$) jestliže
  $$\exists \beta, \gamma, \eta, \nu \in (V \cup T)^* : \alpha = \eta\beta\nu,\ \omega = \eta\gamma\nu \text{ a } (\beta \to \gamma) \in P.$$
- Říkáme, že α se **přepíše** na ω (píšeme $\alpha \Rightarrow^* \omega$) jestliže
  $$\exists \beta_1, \ldots, \beta_n \in (V \cup T)^* : \alpha = \beta_1 \Rightarrow \beta_2 \Rightarrow \ldots \Rightarrow \beta_n = \omega,$$
  tj. také $\alpha \Rightarrow^* \alpha$.
- Posloupnost $\beta_1, \ldots, \beta_n$ nazýváme **derivací (odvozením)**.
- Pokud $\forall i \neq j : \beta_i \neq \beta_j$, hovoříme o **minimálním odvození**.
- Libovolný řetězec $\omega \in (T \cup V)^*$ odvoditelný z počátečního symbolu nazýváme **sentenciální forma**.

Definition 5.4 (Jazyk generovaný gramatikou G)

Jazyk $L(G)$ generovaný gramatikou $G = (V, T, P, S)$ je množina terminálních řetězců, pro které existuje derivace ze startovního symbolu

$$L(G) = \{w \in T^* | S \Rightarrow_G^* w\}.$$

Jazyk neterminálu $A \in V$ definujeme $L(A) = \{w \in T^* | A \Rightarrow_G^* w\}$.

Gramatiky typu 3 a regulární jazyky

Definition (Gramatika typu 3, pravá lineární)

Gramatika G je **pravá lineární, tj. regulární, Typu 3**, pokud obsahuje pouze pravidla tvaru $A \to wB, A \to w, A, B \in V, w \in T^*$.

Example 5.3 (Příklad derivace gramatiky typu 3)

$$P = \{S \to 0S|1A|\epsilon,\ A \to 0A|1B,\ B \to 0B|1S\}$$

$$S \Rightarrow 0S \Rightarrow 01A \Rightarrow 011B \Rightarrow 0110B \Rightarrow 01101S \Rightarrow 01101$$

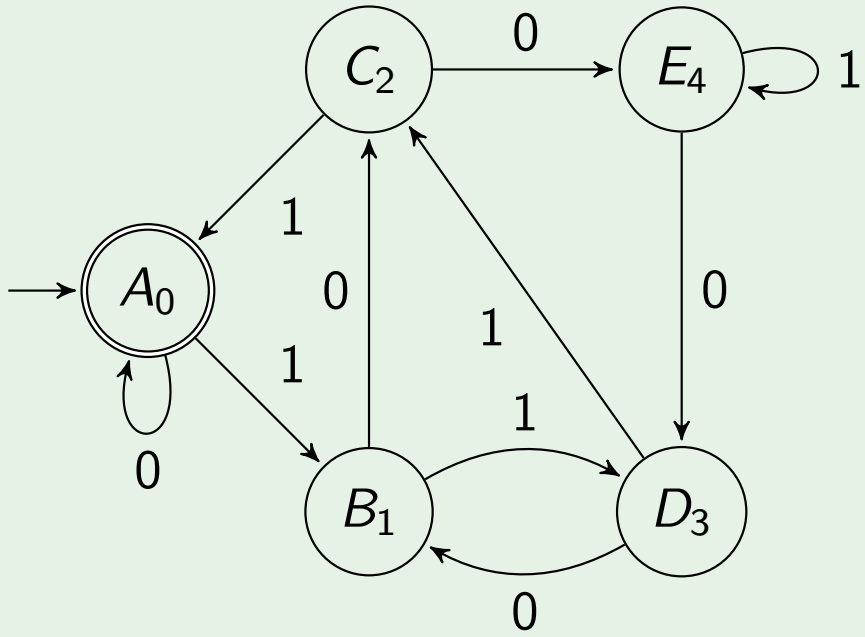
- Pozorování:
  - každá sentenciální forma derivace obsahuje právě jeden neterminál
  - tento neterminál je vždy umístěn zcela vpravo
  - aplikací pravidla $A \to w$ se derivace uzavírá
  - krok derivace generuje symboly a změní neterminál
- Idea vztahu gramatiky a konečného automatu
- neterminál = stav konečného automatu
- pravidla = přechodová funkce.

Příklad převodu FA na gramatiku

Example 5.4 (G, FA binární zápis čísla dělitelného 5)

$L = \{w | w \in \{0,1\}^* \& w$ je binární zápis čísla dělitelného 5$\}$

$$
\begin{aligned}
A &\rightarrow 1B|0A|\epsilon \\
B &\rightarrow 0C|1D \\
C &\rightarrow 0E|1A \\
D &\rightarrow 0B|1C \\
E &\rightarrow 0D|1E
\end{aligned}
$$



Příklady derivací

$$
\begin{aligned}
&A \Rightarrow 0A \Rightarrow 0 && (0)\\
&A \Rightarrow 1B \Rightarrow 10C \Rightarrow 101A \Rightarrow 101 && (5)\\
&A \Rightarrow 1B \Rightarrow 10C \Rightarrow 101A \Rightarrow 1010A \Rightarrow 1010 && (10)\\
&A \Rightarrow 1B \Rightarrow 11D \Rightarrow 111C \Rightarrow 1111A \Rightarrow 1111 && (15)
\end{aligned}
$$

Převod konečného automatu na gramatiku typu 3

Theorem 5.1 ($L \in RE \Rightarrow L \in \mathcal{L}_3$)

Pro každý jazyk rozpoznávaný konečným automatem existuje gramatika typu 3, která ho generuje.

Proof: Převod konečného automatu na gramatiku typu 3

- $L = L(A)$ pro deterministický konečný automat $A = (Q, \Sigma, \delta, q_0, F)$.
- definujme gramatiku $G = (Q, \Sigma, P, q_0)$, kde pravidla P mají tvar
  $p \to aq$, když $\delta(p, a) = q$
  $p \to \epsilon$, když $p \in F$
- je $L(A) = L(G)$?
  - $\epsilon \in L(A) \Leftrightarrow q_0 \in F \Leftrightarrow (q_0 \to \epsilon) \in P \Leftrightarrow \epsilon \in L(G)$
  - $a_1 \dots a_n \in L(A) \Leftrightarrow \exists q_0, \dots, q_n \in Q$ tž. $\delta(q_i, a_{i+1}) = q_{i+1}, q_n \in F$
    $\Leftrightarrow (q_0 \Rightarrow a_1 q_1 \Rightarrow \dots a_1 \dots a_n q_n \Rightarrow a_1 \dots a_n)$ je derivace pro $a_1 \dots a_n$
    $\Leftrightarrow a_1 \dots a_n \in L(G)$

Příprava převodu gramatiky typu 3 na FA

- Opačný směr
  - pravidla $A \to aB$ kódujeme do přechodové funkce
  - pravidla $A \to \epsilon$ určují koncové stavy
  - pravidla $A \to a_1 \dots a_n B, A \to a_1 \dots a_n$ s více neterminály rozepíšeme
    - ★ zavedeme nové neterminály $Y_2, \dots, Y_n, Z_1, \dots, Z_n$
    - ★ vytvoříme pravidla $A \to a_1 Y_2, Y_2 \to a_2 Y_3, \dots, Y_n \to a_n B$
    - ★ resp. $Z \to a_1 Z_1, Z_1 \to a_2 Z_2, \dots, Z_{n-1} \to a_n Z_n, Z_n \to \epsilon$
  - pravidla $A \to B$ odpovídají ϵ přechodům
    - ★ zbavíme se jich tranzitivním uzávěrem
    - ★ nebo musíme tranzitivně uzavřít $S \to B$ pro hledání $S \to \epsilon$.

Lemma

Ke každé gramatice typu 3 existuje gramatika typu 3, která generuje stejný jazyk a obsahuje pouze pravidla ve tvaru: $A \to aB, A \to \epsilon$, $A, B \in V, a \in T$.

Standardizace gramatiky typu 3

Lemma

Ke každé gramatice typu 3 existuje gramatika typu 3, která generuje stejný jazyk a obsahuje pouze pravidla ve tvaru: $A \to aB, A \to \epsilon$, $A, B \in V, a \in T$.

Proof.

Pro gramatiku $G = (V, T, S, P)$ definujeme $G^| = (V^|, T, S, P^|)$, kde pro každé pravidlo zavedeme dostatečný počet nových neterminálů $Y_2, \ldots, Y_n, Z_1, \ldots, Z_n$ a definujeme

P	$P^|$
$A \to aB$ $A \to \epsilon$ $A \to a_1 \ldots a_n B$ $Z \to a_1 \ldots a_n$	$A \to aB$ $A \to \epsilon$ $A \to a_1 Y_2, Y_2 \to a_2 Y_3, \ldots Y_n \to a_n B$ $Z \to a_1 Z_1, Z_1 \to a_2 Z_2, \ldots, Z_{n-1} \to a_n Z_n, Z_n \to \epsilon$
odstraníme i pravidla: $A \to B$	tranzitivní uzávěr $U(A) = \{B \mid B \in V \& A \Rightarrow^* B\}$ $A \to \gamma$ pro všechna $Z \in U(A)$ a $(Z \to \gamma) \in P^|$

□

Pouze pravidla $A \to aB, A \to \epsilon$

Example 5.5

P	$P^{\mid}$
$B \to a_1$	$B \to a_1 H_1, H_1 \to \epsilon$
	$U(A) = \{A, B\}$, proto
$A \to B$	$A \to a_1 H_2, H_2 \to \epsilon$
$A \to a_2$	$A \to a_2 H_3, H_3 \to \epsilon$

Převod gramatiky typu 3 na konečný automat

Theorem 5.2 (ϵ–NFA pro gramatiku typu 3 rozpoznávající stejný jazyk)

Pro každý jazyk L generovaný gramatikou typu 3 existuje ϵ–NFA rozpoznávající L.

Proof: Převod gramatiky typu 3 na konečný automat

- Vezmeme $G = (V, T, P, S)$ obsahující jen pravidla tvaru $A \to aB$, $A \to \epsilon$, $A, B \in V, a \in T$ generující L (předchozí lemma)
- definujeme nedeterministický ϵ–NFA $A = (V, T, \delta, S, F)$, kde:

  $F = \{A | (A \to \epsilon) \in P\}$
  $\delta(A, a) = \{B | (A \to aB) \in P\}$
- $L(G) = L(A)$
  - $\epsilon \in L(G) \Leftrightarrow (S \to \epsilon) \in P \Leftrightarrow S \in F \Leftrightarrow \epsilon \in L(A)$
  - $a_1 \dots a_n \in L(G) \Leftrightarrow$ existuje derivace $(S \Rightarrow a_1 H_1 \Rightarrow \dots \Rightarrow a_1 \dots a_n H_n \Rightarrow a_1 \dots a_n)$

  $\Leftrightarrow$ $\exists H_0, \dots, H_n \in V$ tak že $H_0 = S, H_n \in F$
  $H_{i+1} \in \delta(H_i, a_k)$ pro krok $a_1 \dots a_{k-1} H_i \Rightarrow a_1 \dots a_{k-1} a_k H_{i+1}$

  $\Leftrightarrow$ $a_1 \dots a_n \in L(A)$

Levé (a pravé) lineání gramatiky

Definition 5.5 (Levé (a pravé) lineání gramatiky)

Gramatiky typu 3 nazýváme také **pravé lineární** (neterminál je vždy vpravo).
Gramatika G je **levá lineání**, jestliže má pouze pravidla tvaru
$A \to Bw, A \to w, A, B \in V, w \in T^*$.

Lemma

Jazyky generované levou lineání gramatikou jsou právě regulární jazyky.

Proof:

- $\Rightarrow$ 'otočením' pravidel levé lineární gramatiky dostaneme pravou lineární
  $A \to Bw, A \to w$ převedeme na $A \to w^R B, A \to w^R$
  - získaná gramatika generuje jazyk L^R, najdeme automat
  - víme, že regulární jazyky jsou uzavřené na reverzi,
    L^R je regulární, tudíž i $L = (L^R)^R$ je regulární
- $\Leftarrow$ takto lze získat všechny regulární jazyky
  - (FA$\Rightarrow$reverze$\Rightarrow$ pravá lineární gramatika$\Rightarrow$ levá lineární gramatika) $\square$

Lineární gramatiky (a jazyky)

- Levá a pravá lineární pravidla dohromady jsou už silnější.

Definition 5.6 (lineární gramatika, jazyk)

Gramatika je lineární, jestliže má pouze pravidla tvaru $A \to uBw, A \to w, A, B \in V, u, w \in T^*$ (na pravé straně vždy maximálně jeden neterminál).
Lineární jazyky jsou právě jazyky generované lineáními gramatikami.

- Zřejmě platí: regulární jazyky $\subseteq$ lineární jazyky.
- Jde o vlastní podmnožinu $\subsetneq$.

Example 5.6 (lineární, neregulární jazyk)

Jazyk $L = \{0^i 1^i | i \geq 1\}$ není regulární jazyk, ale je lineární, generovaný gramatikou s pravidly $S \to 0S1|01$.

Pozorování:

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

Bezkontextová gramatika pro jednoduché výrazy

Definition (Bezkontextová gramatika)

Bezkontextová gramatika je gramatika, kde všechna pravidla jsou tvaru $A \to \omega, \omega \in (V \cup T)^*$.

Example 5.7 (CFG pro jednoduché výrazy)

Gramatika pro jednoduché výrazy $G = (\{E, I\}, \{+, *, (,), a, b, 0, 1\}, P, E)$, P jsou pravidla vypsaná vpravo.

- Pravidla 1–4 definují výraz.
- Pravidla 5–10 definují identifikátor I, odpovídající regulárnímu výrazu $(\mathbf{a} + \mathbf{b})(\mathbf{a} + \mathbf{b} + \mathbf{0} + \mathbf{1})^*$.

CFG pro jednoduché výrazy

1. $E \to I$
2. $E \to E + E$
3. $E \to E * E$
4. $E \to (E)$
5. $I \to a$
6. $I \to b$
7. $I \to Ia$
8. $I \to Ib$
9. $I \to I0$
10. $I \to I1$

Derivační strom

Definition 5.7 (Derivační strom)

Mějme gramatiku $G = (V, T, P, S)$. **Derivační strom** pro G je strom, kde:

- Kořen (kreslíme nahoře) je označen startovním symbolem S,
- každý vnitřní uzel je ohodnocen neterminálem V.
- Každý uzel je ohodnocen prvkem $\in V \cup T \cup \{\epsilon\}$.
- Je–li uzel ohodnocen ϵ, je jediným dítětem svého rodiče.
- Je–li A ohodnocení vrcholu a jeho děti zleva pořadě jsou ohodnoceny $X_1, \ldots, X_k$, pak $(A \to X_1 \ldots X_k) \in P$ je pravidlo gramatiky.

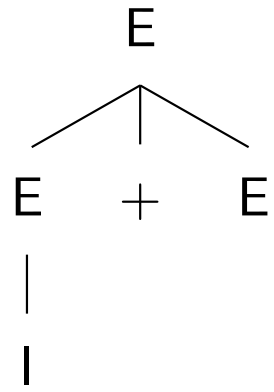
Notation 1 (Terminologie stromů)

Uzly, rodiče, děti, kořen, vnitřní uzly, listy, následníci, předci.

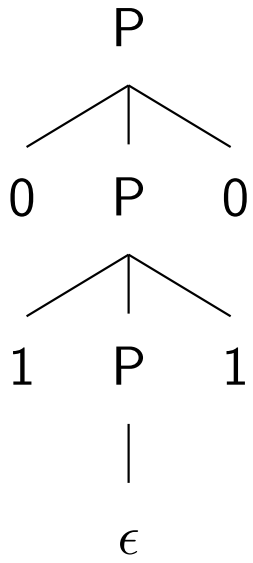
- Stromová struktura reprezentuje zdrojový program v překladači. Struktura usnadňuje překlad do strojového kódu.

Příklady stromů, Strom dává sentenciální formu, slovo

Derivační strom $E \Rightarrow^* I + E$.



Derivační strom $P \Rightarrow^* 0110$.



Definition 5.8 (Strom dává slovo (yield))

Říkáme, že **derivanční strom dává sentenciální formu α (slovo w)**, jestliže je α (w) zřetězení listů bráno zleva doprava.

Levá a pravá derivace

Definition 5.9 (Levá a pravá derivace)

Levá derivace (leftmost) $\Rightarrow_{lm}$, $\Rightarrow_{lm}^{*}$ v každém kroku přepisuje nejlevnější neterminál.
Pravá derivace (rightmost) $\Rightarrow_{rm}$, $\Rightarrow_{rm}^{*}$ v každém kroku přepisuje nejpravější neterminál.

Example 5.8 (levá derivace)

$E \Rightarrow_{lm} E * E \Rightarrow_{lm} I * E \Rightarrow_{lm} a * E \Rightarrow_{lm} a * (E) \Rightarrow_{lm} a * (E + E) \Rightarrow_{lm} a * (I + E) \Rightarrow_{lm}$
$\Rightarrow_{lm} a * (a + E) \Rightarrow_{lm} a * (a + I) \Rightarrow_{lm} a * (a + I0) \Rightarrow_{lm} a * (a + I00) \Rightarrow_{lm} a * (a + b00)$

Pravá derivace používá stejné přepisy, jen je provádí v jiném pořadí.

Example 5.9 (rightmost derivation)

$E \Rightarrow_{rm} E * E \Rightarrow_{rm} E * (E) \Rightarrow_{rm} E * (E + E) \Rightarrow_{rm} E * (E + I) \Rightarrow_{rm}$
$\Rightarrow_{rm} E * (E + I0) \Rightarrow_{rm} E * (E + I00) \Rightarrow_{rm} E * (E + b00) \Rightarrow_{rm}$
$\Rightarrow_{rm} E * (I + b00) \Rightarrow_{rm} E * (a + b00) \Rightarrow_{rm} I * (a + b00) \Rightarrow_{rm} a * (a + b00)$

Derivace a derivační stromy

Theorem 5.3

Pro danou gramatiku $G = (V, T, P, S)$ *a* $w \in T^*$ *jsou následující tvrzení ekvivalentní:*

1. $A \Rightarrow_{lm}^* w$.
2. $A \Rightarrow^* w$.
3. *Existuje derivační strom s kořenem* A *dávající slovo* w.

Proof

$1 \Rightarrow 2$ Levá derivace je derivace, druhý bod z prvního plyne triviálně.

$2 \Rightarrow 3$ Z derivace vytvoříme strom tak, že kořen ohodnotíme počátečním neterminálem a každé pravidlo v derivaci 'zavěsíme' pod uzel odpovídající přepisovanému neterminálu.

$3 \Rightarrow 1$ Pro důkaz ekvivalence zbývá pro libovolný derivační strom najít levou derivaci.

Od stromů k derivaci

Lemma

Mějme CFG $G = (V, T, P, S)$ a derivační strom s kořenem A dávající slovo $w \in T^*$.
Pak existuje levá derivace $A \Rightarrow_{lm}^* w$ v G.

Příprava důkazu: 'obalení derivace'

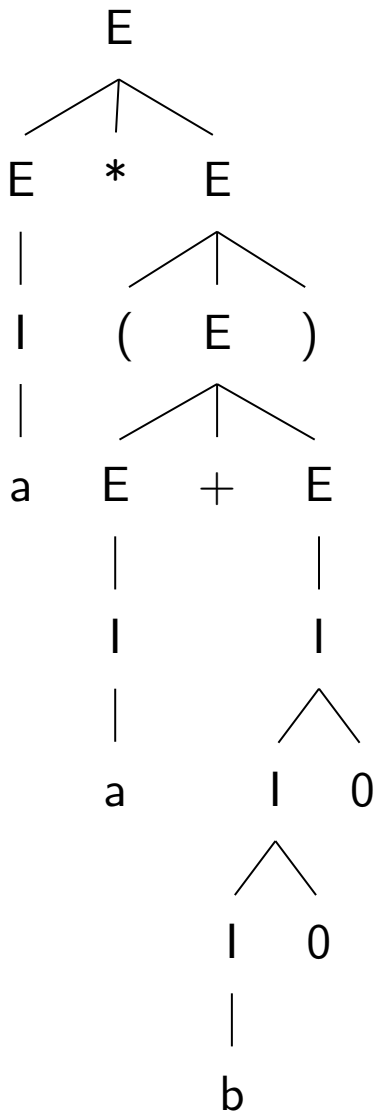
Mějme následující derivaci:

$$E \Rightarrow I \Rightarrow Ib \rightarrow ab.$$

Pro libovolná slova $\alpha, \beta \in (V \cup T)^*$ je také derivace:

$$\alpha E \beta \Rightarrow \alpha I \beta \Rightarrow \alpha Ib \beta \Rightarrow \alpha ab \beta.$$

Příklad levé derivace z derivačního stromu



Je příjemnější zachytit derivaci stromem.

- Kořen: $E \Rightarrow_{lm} E*E$
- Levé dítě kořene: $E \Rightarrow_{lm} I \Rightarrow_{lm} a$
- Kořen a levé dítě: $E \Rightarrow_{lm} E*E \Rightarrow_{lm} I*E \Rightarrow_{lm} a*E$
- Pravé dítě kořene:
  $E \Rightarrow_{lm} (E) \Rightarrow_{lm} (E+E) \Rightarrow_{lm} (I+E) \Rightarrow_{lm} (a+E)$
  $\Rightarrow_{lm} (a+I) \Rightarrow_{lm} (a+I0) \Rightarrow_{lm} (a+I00) \Rightarrow_{lm} (a+b00)$
- Plná derivace:
  $E \Rightarrow_{lm} E*E \Rightarrow_{lm} I*E \Rightarrow_{lm} a*E \Rightarrow_{lm}$
  $\Rightarrow_{lm} a*(E) \Rightarrow_{lm} a*(E+E) \Rightarrow_{lm} a*(I+E) \Rightarrow_{lm}$
  $\Rightarrow_{lm} a*(a+E) \Rightarrow_{lm} a*(a+I) \Rightarrow_{lm} a*(a+I0) \Rightarrow_{lm}$
  $\Rightarrow_{lm} a*(a+I00) \Rightarrow_{lm} a*(a+b00).$

Proof: ∃ derivační strom pak existuje levá derivace $\Rightarrow_{lm}$

Indukcí podle výšky stomu.

- Základ: výška 1: Kořen A s dětmi dávajícími w. Je to derivační strom, proto, $A \to w$ je pravidlo $\in P$, tedy $A \Rightarrow_{lm} w$ v jednom kroku.
- Indukce: výška $n > 1$. Kořen A s dětmi $X_1, X_2, \ldots, X_k$.
  - Je–li $X_i \in T$, definujeme $w_i \equiv X_i$.
  - Je–li $X_i \in V$, z indukčního předpokladu $X_i \Rightarrow_{lm}^* w_i$.

  Levou derivaci konstruujeme induktivně pro $i = 1, \ldots, k$ složíme $A \Rightarrow_{lm}^* w_1 w_2 \ldots w_i X_{i+1} X_{i+2} \ldots X_k$.
  - Pro $X_i \in T$ jen zvedneme čítač $i++$.
  - Pro $X_i \in V$ přepíšeme derivaci: $X_i \Rightarrow_{lm} \alpha_1 \Rightarrow_{lm} \alpha_2 \ldots \Rightarrow_{lm} w_i$ na

$$
\begin{aligned}
& w_1 w_2 \ldots w_{i-1} X_i X_{i+1} X_{i+2} \ldots X_k \Rightarrow_{lm} \\
& w_1 w_2 \ldots w_{i-1} \alpha_1 X_{i+1} X_{i+2} \ldots X_k \Rightarrow_{lm} \\
& \qquad \ldots \\
& \Rightarrow_{lm} w_1 w_2 \ldots w_{i-1} w_i X_{i+1} X_{i+2} \ldots X_k .
\end{aligned}
$$

Pro $i = k$ dostaneme levou derivaci w z A. □

Ekvivalence gramatik

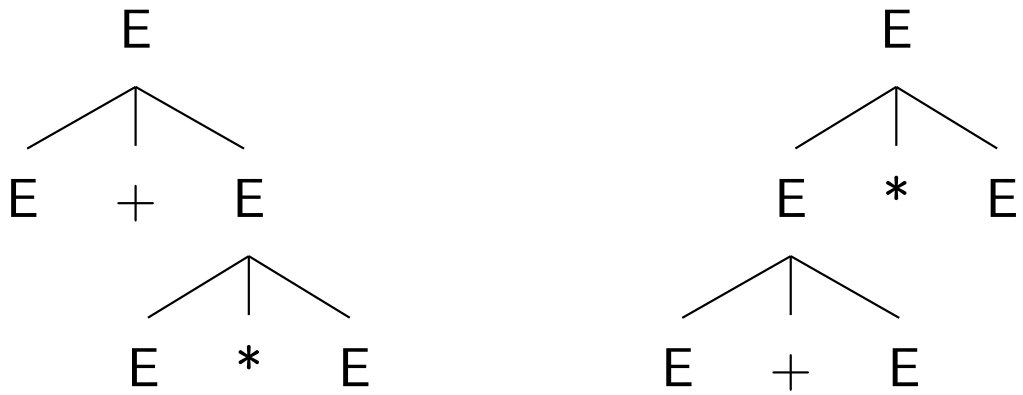
Definition 5.10 (ekvivalence gramatik)

Gramatiky G_1, G_2 jsou **ekvivalentní**, jestliže $L(G_1) = L(G_2)$, tj. generují stejný jazyk.

Víceznačnost gramatik

Dvě derivace téhož výrazu:

$E \Rightarrow E + E \Rightarrow E + E * E$ $\quad$ $E \Rightarrow E * E \Rightarrow E + E * E$



- Rozdíl je důležitý, vlevo $1 + (2 * 3) = 7$, vpravo $(1 + 2) * 3 = 9$.
- Tato gramatika může být modifikovaná na jednoznačnou.

Example 5.10

Různé derivace mohou reprezentovat stejný derivační strom, pak není problém.

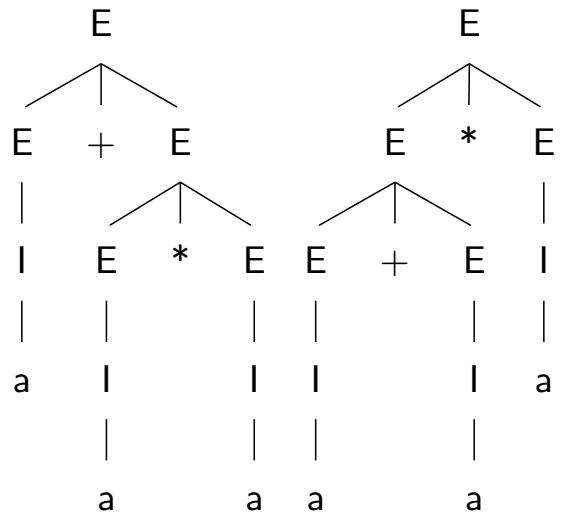
1. $E \Rightarrow E + E \Rightarrow I + E \Rightarrow a + E \Rightarrow a + I \Rightarrow a + b$
2. $E \Rightarrow E + E \Rightarrow E + I \Rightarrow I + I \Rightarrow I + b \Rightarrow a + b$.

Definition 5.11 (Jednoznačnost a víceznačnost CFG)

- Bezkontextová gramatika $G = (V, T, P, S)$ je **víceznačná** pokud existuje aspoň jeden řetězec $w \in T^*$ pro který můžeme najít dva různé derivační stromy, oba s kořenem S dávající slovo w.
- V opačném případě nazýváme gramatiku **jednoznačnou**.
- Bezkontextový jazyk L je **jednoznačný**, jestliže existuje jednoznačná CFG G tak, že $L = L(G)$.
- **Bezkontextový jazyk L je (podstatně) nejednoznačný**, jestliže každá CFG G taková, že $L = L(G)$, je nejednoznačná. Takovému jazyku říkáme i **víceznačný**.

Example 5.11 (nejednoznačnost CFG)

Dva derivační stromy dávající $a + a * a$ ukazující víceznačnost gramatiky.



Příklad víceznačného jazyka

Example 5.12 (Víceznačný jazyk)

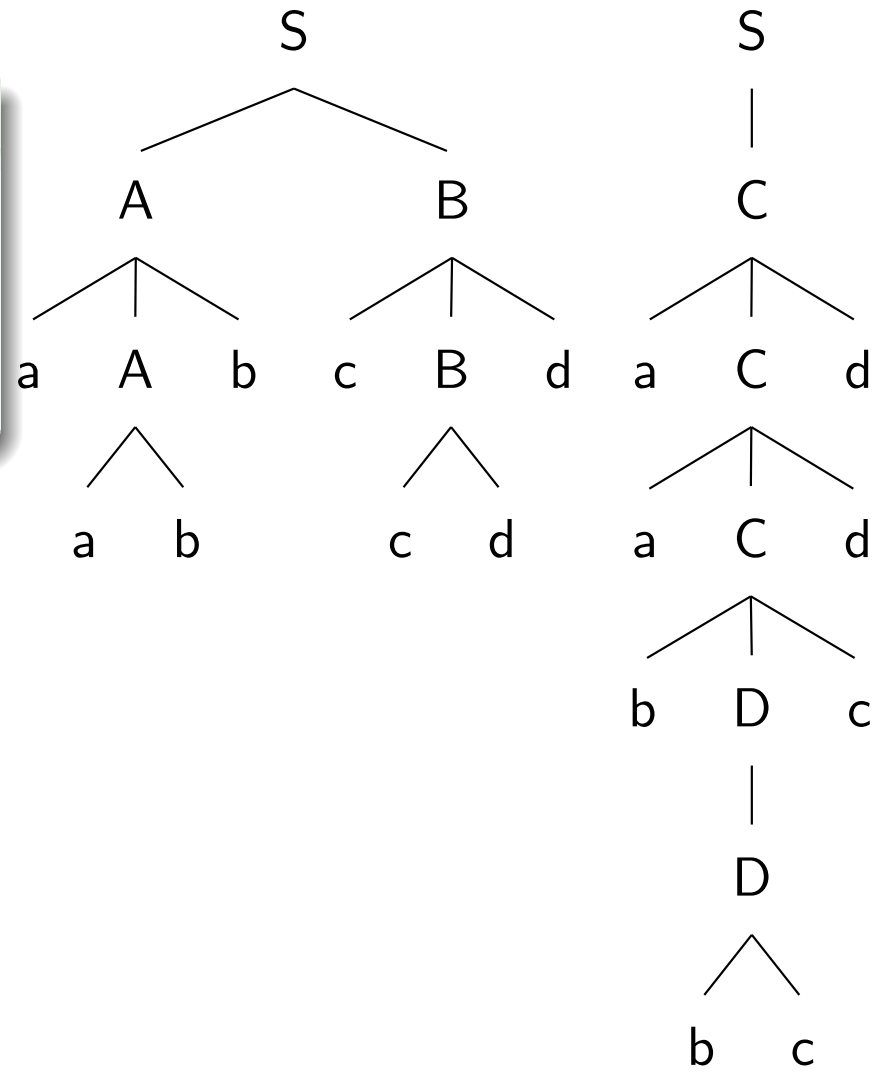
Příklad víceznačného jazyka:

$$L = \{a^n b^n c^m d^m | n \geq 1, m \geq 1\} \cup \cup \{a^n b^m c^m d^n | n \geq 1, m \geq 1\}.$$

1. $S \to AB|C$
2. $A \to aAb|ab$
3. $B \to cBd|cd$
4. $C \to aCd|aDd$
5. $D \to bDc|bc$.

Jakákoli gramatika pro daný jazyk bude generovat pro některá slova typu $a^n b^n c^n d^n$ dva různé derivační stromy.

Dva derivační stromy pro *aabbccdd*.



Odstanění víceznačnosti gramatiky

- Neexistuje algoritmus, který nám řekne, zda je daná gramatika víceznačná.
- Existují bezkontextové jazyky, pro které neexistuje jednoznačná bezkontextová gramatika, pouze víceznačné CFG.
- Existují určitá doporučení pro odstranění víceznačnosti.

Víceznačnost má různé příčiny:

- Není respektovaná priorita operátorů.
- Posloupnost identických operátorů lze shlukovat zleva i zprava.
- $S \rightarrow$ if then S else $S|$ if then $S|\epsilon$

  slovo 'if then if then else' má dva významy

  'if then (if then else)' nebo 'if then (if then) else'

  Řešení:
  - syntaktická chyba (Algol 60)
  - else patří k bližšímu if (preference pořadí pavidel)
  - závorky begin-end,odsazení v Python (asi nejčistší řešení).

Vynucení priority

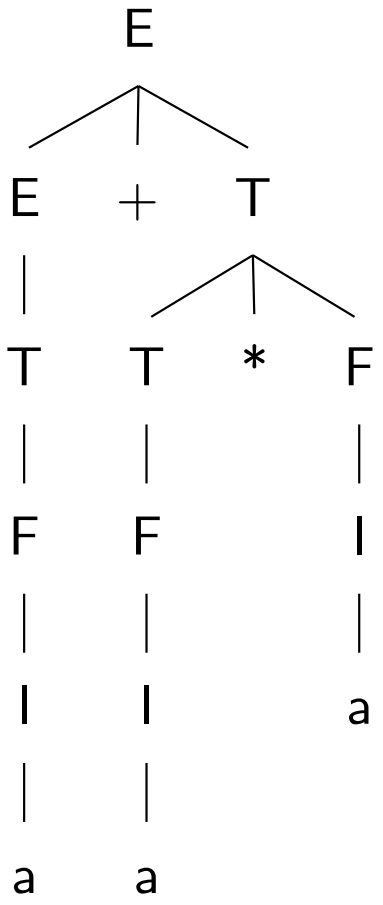
Řešením je zavést více různých proměnných, každou pro jednu úroveň 'priority'.
Konkrétně:

- **Faktor** je výraz který nesmí rozdělit žádný operátor.
  - identifikátory
  - výraz v závorkách
- **Term** je výraz, který nemůže rozdělit operátor $+$.
- **Výraz** může být rozdělen $*$ i $+$.

Jednoznačná gramatika pro výrazy:

1. $I \to a|b|Ia|Ib|I0|I1$
2. $F \to I|(E)$
3. $T \to F|T * F$
4. $E \to T|E + T$.

Jediný derivační strom pro $a + a * a$.



Jednoznačnost a kompilátory

Kompilace výrazu (zásobník na mezivýsledky + dva registry):

(1) $E \to E + T$ ... pop r1; pop r2; add r1,r2; push r2
(2) $E \to T$
(3) $T \to T * F$ ... pop r1; pop r2; mul r1,r2; push r2
(4) $T \to F$
(5) $F \to (E)$
(6) $F \to a$ ... push a

- 'a+a*a' získáme postupnou aplikací pravidel 1,2,4,6,3,4,6,6
- posloupnost obrátíme a vybereme pouze pravidla generující kód

  6,6,3,6,1
- nyní nahradíme pravidla příslušným kódem

  push a; push a; pop r1; pop r2; mul r1,r2; push r2; push a; pop r1; pop r2; add r1,r2; push r2

Shrnutí

- Gramatiky
  - obecné
  - kontextové
  - bezkontextové
  - regulární, pravé lineární
- jazyk gramatiky, derivace, derivace dává slovo, derivační strom (pro bezkontextové gramatiky), ekvivalentní gramatiky
- ne každá lineární gramatika má ekvivalentní pravou lineární
- bezkontextové gramatiky
- jednoznačné a (podstatně) víceznačné gramatiky.

Dnešní téma

Chomského NF, Pumping Lemma pro CFL,CYK – náležení do CFL

Marta Vomlelová

MS 303

Bezkontextové gramatiky (CFG) v Chomského normální formě

- Chomského normální forma bezkontextové gramatiky:
  - neobsahuje zbytečné symboly
  - všechna pravidla tvaru $A \to BC$ nebo $A \to a$, A, B, C jsou neterminály, a terminál.
  - Pro každý bezkontextový jazyk L, kde $L \setminus \{\epsilon\} \neq \emptyset$ existuje gramatika v Chomského normálním tvaru, která generuje $L \setminus \{\epsilon\}$.

Postupně provedeme zjednodušení gramatiky, nejdřív:

- Eliminace *zbytečných symbolů*
- eliminace ϵ*–pravidel* $A \to \epsilon$; $A \in V$
- eliminace *jednotkových pravidel* $A \to B$ pro $A, B \in V$.

- To vše potřebujeme k formulaci iteračního lemmatu pro bezkontextové jazyky.
- A ověření, zda zadaný řetězec patří do jazyka gramatiky pomocí Cocke-Younger-Kasami algoritmu.

Eliminace zbytečných symbolů

Definition 6.1 (zbytečný, užitečný, generující, dosažitelný symbol)

- Symbol X je **užitečný** v gramatice $G = (V, T, P, S)$ pokud existuje derivace tvaru $S \Rightarrow^* \alpha X \beta \Rightarrow^* w$ kde $w \in T^*, X \in (V \cup T)$, $\alpha, \beta \in (V \cup T)^*$.
- Pokud X není užitečný, říkáme, že je **zbytečný**.
- X je **generující** pokud $X \Rightarrow^* w$ pro nějaké slovo $w \in T^*$. Vždy $w \Rightarrow^* w$ v nula krocích.
- X je **dosažitelný** pokud $S \Rightarrow^* \alpha X \beta$ pro nějaká $\alpha, \beta \in (V \cup T)^*$.

Chceme eliminovat ne–generující a ne–dosažitelné symboly.

Example 6.1

Uvažujme gramatiku:
$S \to AB|a$
$A \to b$

Eliminujeme B (ne–generující):
$S \to a$
$A \to b$.

Eliminujeme A (nedosažitelný):
$S \to a$.

Lemma 6.1 (Eliminace zbytečných symbolů)

Nechť $G = (V, T, P, S)$ je CFG, předpokládejme $L(G) \neq \emptyset$. Zkonstruujeme $G_1 = (V_1, T_1, P_1, S)$ následovně:

- Eliminujeme ne–generující symboly a pravidla je obsahující
- poté eliminujeme všechny nedosažitelné symboly

Pak G_1 nemá zbytečné symboly a $L(G_1) = L(G)$.

Algorithm: Generující symboly

Základ Každý $a \in T$ je generující.

Indukce Pro každé pravidlo $A \to \alpha$, kde každý symbol v α je generující. Pak i A je generující.
(Včetně $A \to \epsilon$).

Algorithm: Dosažitelné symboly

Základ S je dosažitelný.

Indukce Je–li A dosažitelný, pro všechna pravidla $A \to \alpha$ jsou všechny symboly v α dosažitelné.

Lemma 6.2 (generující/dosažitelné symboly)

Výše uvedené algoritmy najdou právě všechny generující / dosažitelné symboly.

Eliminace ϵ pravidel

Definition 6.2 (nulovatelný neterminál)

Neterminál A je **nulovatelný** pokud $A \Rightarrow^* \epsilon$.

Pro nulovatelné neterminály na pravé straně pravidla $B \to CAD$, vytvoříme dvě verze pravidla – s a bez nulovatelného neterminálu.

Algorithm: Nalezení nulovatelných symbolů v G

Základ Pokud $A \to \epsilon$ je pravidlo G, pak A je nulovatelné.

Indukce Pokud $B \to C_1 \dots C_k$, kde jsou všechna C_i nulovatelná, je i B nulovatelné (terminály nejsou nulovatelné nikdy).

Algorithm: Konstrukce gramatiky bez ϵ–pravidel z G

- Najdi nulovatelné symboly
- Pro každé pravidlo $A \to X_1 \dots X_k \in P, k \geq 1$, nechť m z X_i je nulovatelných. Nová gramatika G_1 bude mít 2^m verzí tohoto pravidla s/bez každého nulovatelného symbolu kromě ϵ v případě $m = k$.

Příklad eliminace ϵ–pravidel

Example 6.2

Mějme gramatiku:

$S \to AB$
$A \to aAA|\epsilon$
$B \to bBB|\epsilon$

$S \to AB|A|B$
$A \to aAA|aA|aA|a$
$B \to bBB|bB|bB|b$

Výsledná gramatika:

$S \to AB|A|B$
$A \to aAA|aA|a$
$B \to bBB|bB|b.$

Eliminace jednotkových pravidel

Definition 6.3 (jednotkové pravidlo)

Jednotkové pravidlo je $A \to B \in P$ kde A, B jsou oba neterminály.

Example 6.3

$I \to a|b|Ia|Ib|I0|I1$
$F \to I|(E)$
$T \to F|T * F$
$E \to T|E + T$

Expanze T v $E \to T$
$E \to F|T * F$
Expanze $E \to F$
$E \to I|(E)$

Expanze $E \to I$
$E \to a|b|Ia|Ib|I0|I1$

Dohromady: $E \to a|b|Ia|Ib|I0|I1|(E)|T * F|E + T$.

Musíme se vyhnout možným cyklům.

Definition 6.4 (jednotkový pár)

Dvojici $A, B \in V$ takovou, že $A \Rightarrow^* B$ pouze jednotkovými pravidly nazýváme **jednotkový pár** (jednotková dvojice).

Algorithm: Nalezení jednotkových párů

Základ (A, A) pro každý $A \in V$ je jednotkový pár.

Indukce Je–li (A, B) jednotkový pár a $(B \to C) \in P$, pak (A, C) je jednotkový pár.

Example 6.4 (Jednotkové páry z předchozí gramatiky)

$(E, E), (T, T), (F, F), (I, I), (E, T), (E, F), (E, I), (T, F), (T, I), (F, I)$.

Algorithm: Eliminace jednotkových pravidel z G

- najdi všechny jednotkové páry v G
- pro každý jednotkový pár (A, B) dáme do nové gramatiky všechna pravidla $A \to \alpha$ kde $B \to \alpha \in P$ a $B \to \alpha$ není jednotkové pravidlo.

Example 6.5

$I \to a|b|Ia|Ib|I0|I1$
$F \to I|(E)$

$I \to a|b|Ia|Ib|I0|I1$
$F \to (E)|a|b|Ia|Ib|I0|I1$

Gramatiky v normálním tvaru

Lemma 6.3 (Gramatika v normálním tvaru, redukovaná)

Mějme bezkontextovou gramatiku G, $L(G) - \{\epsilon\} \neq \emptyset$. pak existuje CFG G_1 taková že $L(G_1) = L(G) - \{\epsilon\}$ a G_1 neobsahuje ϵ–pravidla, jednotková pravidla ani zbytečné symboly. Gramatika G_1 se nazývá **redukovaná**.

Redukce bezkontextové gramatiky.

Idea důkazu:

- Začneme eliminací ϵ–pravidel.
- Eliminujeme jednotková pravidla. Tím nepřidáme ϵ–pravidla.
- Eliminujeme zbytečné symboly. Tím nepřidáme žádná pravidla.
  - Nejdříve negenerující,
  - pak nedosažitelné.

Definition 6.5 (Chomského normální tvar)

O bezkontextové gramatice $G = (V, T, P, S)$ bez zbytečných symbolů kde jsou všechna pravidla v jednom ze dvou tvarů

- $A \to BC$, $A, B, C \in V$,
- $A \to a$, $A \in V$, $a \in T$,

říkáme, že je v **Chomského normálním tvaru (ChNF)**.

Potřebujeme dva další kroky:

- pravé strany délky 2 a více předělat na samé neterminály
- rozdělit pravé strany délky 3 a více neterminálů na více pravidel

Algorithm: neterminály

- Pro každý terminál a vytvoříme nový neterminál, řekněme A,
- přidáme pravidlo $A \to a$,
- použijeme A místo a na pravé straně pravidel délky 2 a více.

Algorithm: rozdělení pravidel

- Pro pravidlo $A \to B_1 \dots B_k$ zavedeme $k-2$ neterminálů C_i
- Přidáme pravidla $A \to B_1 C_1, C_1 \to B_2 C_2$, $\dots, C_{k-2} \to B_{k-1} B_k$.

Theorem 6.1 (Chomského normální tvar bezkontextové gramatiky)

Mějme bezkontextovou gramatiku G, $L(G) - \{\epsilon\} \neq \emptyset$. Pak existuje CFG G_1 v Chomského normálním tvaru taková, že $L(G_1) = L(G) - \{\epsilon\}$.

Example 6.6

$I \to a|b|Ia|Ib|I0|I1$
$F \to I|(E)$
$T \to F|T * F$
$E \to T|E + T$

$I \to a|b|IA|IB|IZ|IU$
$F \to LER|a|b|IA|IB|IZ|IU$
$T \to TMF|LER|a|b|IA|IB|IZ|IU$
$E \to EPT|TMF|LER|a|b|IA|IB|IZ|IU$
$A \to a$
$B \to b$
$Z \to 0$
$U \to 1$
$P \to +$
$M \to *$
$L \to ($
$R \to)$

$F \to LC_3|a|b|IA|IB|IZ|IU$
$T \to TC_2|LC_3|a|b|IA|IB|IZ|IU$
$E \to EC_1|TC_2|LC_3|a|b|IA|IB|IZ|IU$
$C_1 \to PT$
$C_2 \to MF$
$C_3 \to ER$
$I, A, B, Z, U, P, M, L, R$ jako vlevo

Příprava na (pumping) lemma o vkládání

Lemma (Velikost derivačního stromu gramatiky v CNF)

Mějme derivační strom podle gramatiky $G = (V, T, P, S)$ v Chomského normálním tvaru, který dává slovo w. Je–li délka nejdelší cesty n, pak $|w| \leq 2^{n-1}$.

Proof.

Indukcí podle n,

Základ $|a| = 1 = 2^0$

Indukce $2^{n-2} + 2^{n-2} = 2^{n-1}$.

□

Lemma (Důsledek)

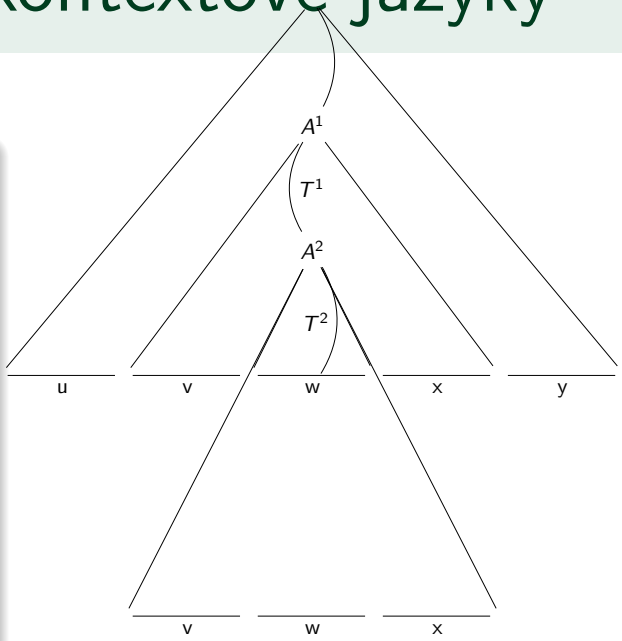
Mějme derivační strom podle gramatiky $G = (V, T, P, S)$ v Chomského normální formě, který dává slovo w, $|w| > p = 2^{n-1}$. Pak ve stromě existuje cesta delší než n.

Lemma o vkládání (pumping) pro bezkontextové jazyky

Theorem 6.2 (!!Lemma o vkládání (pumping) pro bezkontextové jazyky)

Mějme bezkontextový jazyk L. Pak existuje přirozené číslo $n \in \mathbb{N}$ takové, že každé $z \in L, |z| > n$ lze rozložit na $z = uvwxy$ kde:

- $|vwx| \leq n$
- $vx \neq \epsilon$
- $\forall i \geq 0$, $uv^i wx^i y \in L$.



Idea důkazu:

- vezmeme derivační strom pro z
- najdeme nejdelší cestu
- na ní dva stejné neterminály
- tyto neterminály určí dva podstromy
- podstromy definují rozklad slova
- nyní můžeme větší podstrom posunout ($i > 1$)
- nebo nahradit menším podstromem ($i = 0$)

Proof: $|z| > n : z = uvwxy, |vwx| \leq n, vx \neq \epsilon, \forall i \geq 0 uv^i wx^i y \in L$

- vezmeme gramatiku v Chomského NF (pro $L = \{\epsilon\}$ a $\emptyset$ zvol $n = 1$).
- Nechť $|V| = k$. Položíme $n = 2^k$.
- Pro $z \in L, |z| > 2^k$, má v derivačním stromu z cestu délky $> k$
- vezmeme cestu maximální délky; terminál kam vede označíme t
- Aspoň dva z posledních $(k+1)$ neterminálů na cestě do t jsou stejné
- vezmeme dvojici A^1, A^2 nejblíže k t (určuje podstromy T^1, T^2)
- cesta z A^1 do t je nejdelší v podstromu T^1 a má délku maximálně $(k+1)$

  tedy slovo dané stromem T^1 není delší než 2^k (tedy $|vwx| \leq n$)
- z A^1 vedou dvě cesty (ChNF), jedna do T^2 druhá do zbytku vx

  ChNF je nevypouštějící, tedy $vx \neq \epsilon$
- derivace slova $(A^1 \Rightarrow^* vA^2x, A^2 \Rightarrow^* w)$

  $S \Rightarrow^* uA^1y \Rightarrow^* uvA^2xy \Rightarrow^* uvwxy$
- posuneme–li A^2 do A^1 $(i = 0)$

  $S \Rightarrow^* uA^2y \Rightarrow^* uwy$
- posuneme–li A^1 do A^2 $(i = 2, 3, \ldots)$

  $S \Rightarrow^* uA^1y \Rightarrow^* uvA^1xy \Rightarrow^* uvvA^2xxy \Rightarrow^* uvvwxxy.$

□

Použití lemma o vkládání

Example 6.7 (ne–bezkontextový jazyk)

Následující jazyk není bezkontextový

- $\{0^i 1^i 2^i | i \geq 1\}$

- důkaz sporem: předpokládejme bezkontextovost
- z lemmatu o vkládání máme n
- zvolme $z = |0^n 1^n 2^n| > n$

žádné dělení nesplňuje PL neboť

- pumpovací slovo $|vwx| \leq n$
- tj. vždy lze pumpovat maximálně dva různé symboly
- $vx \neq \epsilon$, iterací se slovo změní
- poruší se rovnost počtu symbolů – SPOR.

Example 6.8 (ne–bezkontextový jazyk)

Následující jazyk není bezkontextový

- $\{0^i 1^j 2^k | 0 \leq i \leq j \leq k\}$

- důkaz sporem: předpokládejme bezkontextovost
- z lemmatu o vkládání máme n
- zvolme $z = |0^n 1^n 2^n| > n$

žádné dělení nesplňuje PL neboť

- pumpovací slovo $|vwx| \leq n$
- tj. vždy lze pumpovat maximálně dva různé symboly
  - pokud 0 (nebo 1), pumpujeme nahoru – SPOR $i \leq j$ (nebo $j \leq k$)
  - pokud 2 (nebo 1), pumpujeme dolů – SPOR $j \leq k$ (nebo $i \leq j$)

Použití lemma o vkládání

Example 6.9 (ne–bezkontextový jazyk)

Následující jazyk není bezkontextový

- $\{0^i 1^j 2^i 3^j | i, j \geq 1\}$

- důkaz sporem: předpokládejme bezkontextovost
- z lemmatu o vkládání máme n
- zvolme $z = |0^n 1^n 2^n 3^n| > n$
- pumpovací slovo $|vwx| \leq n$
- tj. vždy lze pumpovat maximálně dva různé sousední symboly
- poruší se rovnost počtu symbolů 0 a 2 nebo 1 a 3 – SPOR.

Example 6.10 (ne–bezkontextový jazyk)

Následující jazyk není bezkontextový

- $\{ww | w \in \{0,1\}^*\}$

- důkaz sporem: předpokládejme bezkontextovost
- z lemmatu o vkládání máme n
- zvolme $z = |0^n 1^n 0^n 1^n| > n$
- pumpovací slovo $|vwx| \leq n$
- tj. vždy lze pumpovat maximálně dva různé sousední symboly
- poruší se buď rovnost nul či jedniček
  - rozeberete 4 případy, vx obsahuje znak z prvních nul, prvních jedniček, druhých nul, druhých jedniček.

Kdy lemma o vkládání nezabere

- Lemma o vkládání je pouze implikace!

Example 6.11 (pumpovatelný, ne–bezkontextový jazyk)

$L = \{a^i b^j c^k d^l | i = 0 \vee j = k = l\}$ není bezkontextový jazyk, přesto lze pumpovat.

$i = 0 : b^j c^k d^l$ lze pumpovat v libovolném písmenu
$i > 0 : a^i b^n c^n d^n$ lze pumpovat v části obsahující a

Co s tím?

- zobecnění pumping lemmatu (Ogdenovo lemma)
  - pumpování vyznačených symbolů
- uzávěrové vlastnosti.

Příklad gramatiky

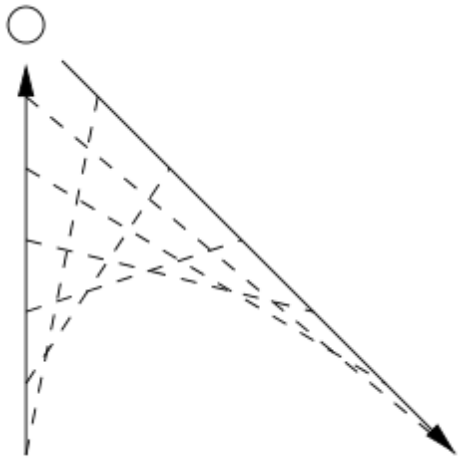
Example 6.12 (Uzávorkování v ChNF)

Mějme gramatiku $G = (\{S, L, R\}, \{(,)\}, P, S)$, kde $P = \left\{ \begin{array}{lll} S & \to & LR|SS|LA \\ A & \to & SR \\ L & \to & (\\ R & \to &) \end{array} \right\}$.

- $S \Rightarrow LR \Rightarrow (R \Rightarrow ()$
- $S \Rightarrow SS \Rightarrow LRS \Rightarrow (RS \Rightarrow ()S \Rightarrow ()LR \Rightarrow ()(R \Rightarrow ()()$
- $S \Rightarrow LA \Rightarrow (A \Rightarrow (SR \Rightarrow (LRR \Rightarrow ((RR \Rightarrow (()R \Rightarrow (())$

Příklad CYK: pro slovo najít derivační strom

- Chceme rozhodnout, zda dané slovo w patří do jazyka bezkontextové gramatiky
  - Vezmeme gramatiku v Chomského normální formě
  - exponenciální složitost: prozkoušíme všechny derivační stromy do hloubky $|n|$,
  - polynomiálně: Cocke-Younger-Kasami algoritmus.



Example 6.13 (CYK algoritmus)

Gramatika
$G = (\{S, A, L, R\}, \{(,)\}, P, S)$:

$$P = \left\{ \begin{array}{lcl} S & \to & LR|SS|LA \\ A & \to & SR \\ L & \to & (\\ R & \to &) \end{array} \right\}$$

Tabulku vyplňujeme odspodu:

$\{S\}$					
$\{\}$	$\{A\}$				
$\{\}$	$\{S\}$	$\{\}$			
$\{\}$	$\{\}$	$\{\}$	$\{A\}$		
$\{\}$	$\{S\}$	$\{\}$	$\{S\}$	$\{\}$	
$\{L\}$	$\{L\}$	$\{R\}$	$\{L\}$	$\{R\}$	$\{R\}$
(	(	)	(	)	)

Cocke-Younger-Kasami algoritmus náležení slova do CFL

Algorithm: CYK algoritmus, v čase $O(n^3)$

- Mějme gramatiku v ChNF $G = (V, T, P, S)$ pro jazyk L a slovo $w = a_1 a_2 \dots a_n \in T^*$.
- Vytvořme trojúhelníkovou tabulku (vpravo),
  - horizontální osa je w
  - X_{ij} jsou množiny neterminálů A takových, že $A \Rightarrow^* a_i a_{i+1} \dots a_j$.

  Základ: $X_{ii} = \{A; A \to a_i \in P\}$

  Indukce: $X_{ij} = \{A \to BC; B \in X_{ik}, C \in X_{k+1,j}\}$
- Vyplňujeme tabulku zdola nahoru.

X_{15}				
X_{14}	X_{25}			
X_{13}	X_{24}	X_{35}		
X_{12}	X_{23}	X_{34}	X_{45}	
X_{11}	X_{22}	X_{33}	X_{44}	X_{55}
a_1	a_2	a_3	a_4	a_5

Theorem 6.3 (CYK)

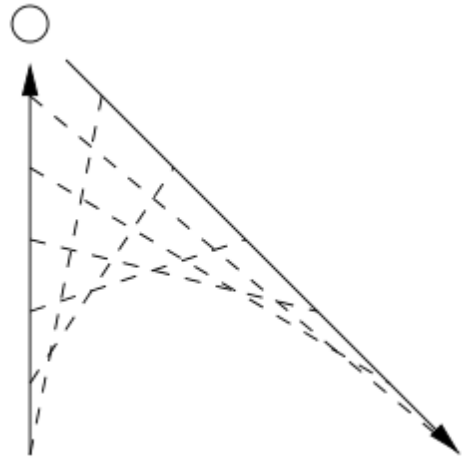
Slovo w patří do jazyka gramatiky G právě když je v CYK algoritmu $S \in X_{1,n}$.

Obecný příklad CYK

Example 6.14 (CYK algoritmus)

Uvažujme gramatiku
$G = (\{S, A, B, C\}, \{a, b\}, P, S)$:

$$P = \left\{ \begin{array}{lcl} S & \to & AB|BC \\ A & \to & BA|a \\ B & \to & CC|b \\ C & \to & AB|a \end{array} \right\}.$$



Pravidla pozpátku:

$$\begin{array}{rcl} AB & \to & \{S, C\} \\ BA & \to & \{A\} \\ BC & \to & \{S\} \\ CC & \to & \{B\} \\ b & \to & \{B\} \\ a & \to & \{A, C\} \end{array}$$

Tabulku vyplňujeme odspodu:

$\{S, A, C\}$				
-	$\{S, A, C\}$			
-	$\{B\}$	$\{B\}$		
$\{S, A\}$	$\{B\}$	$\{S, C\}$	$\{S, A\}$	
$\{B\}$	$\{A, C\}$	$\{A, C\}$	$\{B\}$	$\{A, C\}$
b	a	a	b	a