Bezkontextové jazyky

Jazyky typu 2

Definice 4.1 Gramatika $G = (N, \Sigma, P, S)$ si nazývá bezkontextovou gramatikou, jestliže všechna pravidla z P mají tvar

$$A \to \alpha, \quad A \in N, \quad \alpha \in (N \cup \Sigma)^*$$

Lemma 4.1 Každý regulární jazyk je jazykem bezkontextovým.

❖ Proč studujeme bezkontextové jazyky?

Příklad 4.1 Jazyk $L = \{a^n b^n \mid n \geq 0\}$, jak víme, není jazykem regulárním, je však jazykem bezkontextovým:
$L = L(G)$ kde
$G = (\{S\}, \{a, b\}, \{S \to aSb, S \to \varepsilon\}, S)$

Proč mohou BG „počítat“

- Sebevkládání pomocí pravidel $A \to \alpha A \beta$ kde $\alpha, \beta \in (N \cup \Sigma)^*$

Zkonstruujte bezkontextovou gramatiku pro jazyk $L = \{a^{3n} b^{2n} \mid n \geq 0\}$

Proč mohou BG „počítat“

- Sebevkládání pomocí pravidel $A \to \alpha A \beta$ kde $\alpha, \beta \in (N \cup \Sigma)^*$

Zkonstruujte bezkontextovou gramatiku pro jazyk $L = \{a^{3n} b^{2n} \mid n \geq 0\}$

$G = (\{S\}, \{a, b\}, \{S \to aaaSbb, S \to \varepsilon\}, S)$

Proč mohou BG „počítat“

- Sebevkládání pomocí pravidel $A \to \alpha A \beta$ kde $\alpha, \beta \in (N \cup \Sigma)^*$

Zkonstruujte bezkontextovou gramatiku pro jazyk $L = \{a^{3n} b^{2n} \mid n \geq 0\}$

$G = (\{S\}, \{a, b\}, \{S \to aaaSbb, S \to \varepsilon\}, S)$

- Jak docílit libovolné pořadí symbolů?

Zkonstruujte bezkontextovou gramatiku pro jazyk $L = \{w \in \{a, b\}^* \mid \#_a(w) = \#_b(w)\}$

Proč mohou BG „počítat“

- Sebevkládání pomocí pravidel $A \to \alpha A \beta$ kde $\alpha, \beta \in (N \cup \Sigma)^*$

Zkonstruujte bezkontextovou gramatiku pro jazyk $L = \{a^{3n} b^{2n} \mid n \geq 0\}$

$G = (\{S\}, \{a, b\}, \{S \to aaaSbb, S \to \varepsilon\}, S)$

- Jak docílit libovolné pořadí symbolů?

Zkonstruujte bezkontextovou gramatiku pro jazyk $L = \{w \in \{a, b\}^* \mid \#_a(w) = \#_b(w)\}$

$G = (\{S\}, \{a, b\}, \{S \to aSb, S \to bSa, S \to SS, S \to \varepsilon\}, S)$

Příklad bezkontextové gramatiky

❖ Pro účely demonstrace vysvětlovaných pojmů budeme v následujících příkladech používat následující gramatiku.

Příklad 4.2 $G = (\{S, A, B\}, \{a, b, c\}, P, S)$, kde P obsahuje pravidla

$$S \to AB$$
$$A \to aAb \mid ab$$
$$B \to bBc \mid bc$$

Gramatika G generuje bezkontextový jazyk $L(G) = \{a^m b^{m+n} c^n \mid n \geq 1, m \geq 1\}$

Derivační strom

❖ Důležitým prostředkem pro grafické vyjádření struktury věty (její derivace) je strom, který se nazývá derivačním nebo syntaktickým stromem.

Definice 4.2 Nechť δ je věta nebo větná forma generovaná v gramatice $G = (N, \Sigma, P, S)$ a nechť $S = v_0 \Rightarrow v_1 \Rightarrow \ldots \Rightarrow v_k = \delta$ její derivace v G. Derivační strom příslušející této derivaci je vrcholově ohodnocený strom s těmito vlastnostmi:

1. Vrcholy derivačního stromu jsou ohodnoceny symboly z množiny $N \cup \Sigma \cup \{\varepsilon\}$; kořen stromu je označen výchozím symbolem S.
2. Přímé derivaci $v_{i-1} \Rightarrow v_i, i = 1, 2, \ldots, k$ kde
   - $v_{i-1} = \mu A \lambda,\ \mu, \lambda \in (N \cup \Sigma)^*,\ A \in N$
   - $v_i = \mu \alpha \lambda$
   - $A \to \alpha,\ \alpha = X_1 \ldots X_n$ je pravidlo z P,

   odpovídá právě n hran $(A, X_j), j = 1, \ldots, n$ vycházejících z uzlu A, jež jsou uspořádány zleva doprava v pořadí $(A, X_1), (A, X_2), \ldots, (A, X_n)$.
3. Ohodnocení koncových uzlů derivačního stromu vytváří zleva doprava větnou formu nebo větu δ (plyne z 1. a 2.).

Příklad derivačního stromu

Příklad 4.3 V gramatice z příkladu 4.2 můžeme generovat řetězec $aabbbbcc$ např. derivací:

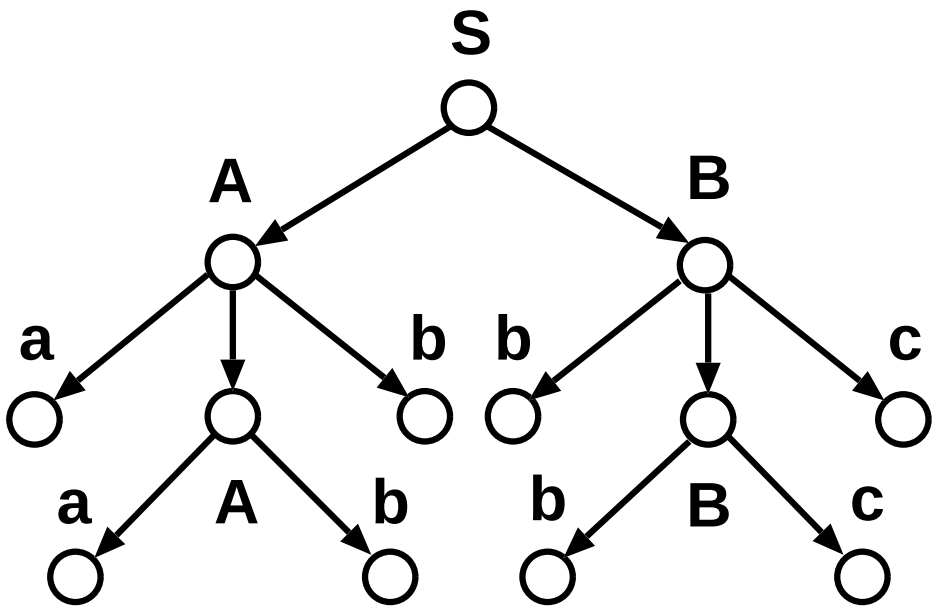
$$S \Rightarrow AB \Rightarrow aAbB \Rightarrow aAbbBc \Rightarrow aAbbbcc \Rightarrow aabbbbcc$$

Derivační strom odpovídající této derivaci vypadá takto (po stranách jsou uvedena použitá pravidla):

$S \to AB$

$A \to aAb$

$A \to ab$



$B \to bBc$

$B \to bc$

Levá a pravá derivace

❖ Ukažme si i jiné derivace věty $aabbbbcc$, které se liší v pořadí, v němž byly vybírány nonterminály pro přímé derivace.

1. $S \Rightarrow AB \Rightarrow aAbB \Rightarrow aabbB \Rightarrow aabbbBc \Rightarrow aabbbbcc$
2. $S \Rightarrow AB \Rightarrow AbBc \Rightarrow Abbcc \Rightarrow aAbbbc \Rightarrow aabbbbcc$

Definice 4.3 Nechť $S \Rightarrow \alpha_1 \Rightarrow \alpha_2 \Rightarrow \ldots \Rightarrow \alpha_n = \alpha$ je derivace větné formy α. Jestliže byl v každém řetězci $\alpha_i, i = 1, \ldots, n-1$ přepsán nejlevější (nejpravější) nonterminál, pak tuto derivaci nazýváme levou (pravou) derivací větné formy α.

Výše uvedené příklady derivací představují levou (1.) a pravou (2.) derivaci.

Lemma 4.2 Je-li $S \equiv \alpha_0 \Rightarrow \alpha_1 \Rightarrow \ldots \Rightarrow \alpha_n \equiv w$ levá, resp. pravá derivace věty w, pak každá z větných forem α_i, $i = 1, 2, \ldots, n-1$ má tvar:

$$x_i A_i \beta_i \text{ kde } x_i \in \Sigma^*, A_i \in N, \beta_i \in (N \cup \Sigma)^*$$

resp.

$$\gamma_i B_i y_i \text{ kde } y_i \in \Sigma^*, B_i \in N, \gamma_i \in (N \cup \Sigma)^*$$

t.j. větné formy levé, resp. pravé derivace mají terminální prefixy, resp. sufixy.

Víceznačnost gramatik

Definice 4.4 Nechť G je gramatika. Říkáme, že věta w generovaná gramatikou G je víceznačná, existují-li alespoň dva různé derivační stromy s koncovými uzly tvořícími větu w. Gramatika G je víceznačná, pokud generuje alespoň jednu víceznačnou větu. V opačném případě mluvíme o jednoznačné gramatice.
Jazyky, které lze generovat víceznačnou gramatikou, ale které nelze generovat jednoznačnou gramatikou, se nazývají jazyky s inherentní víceznačností.

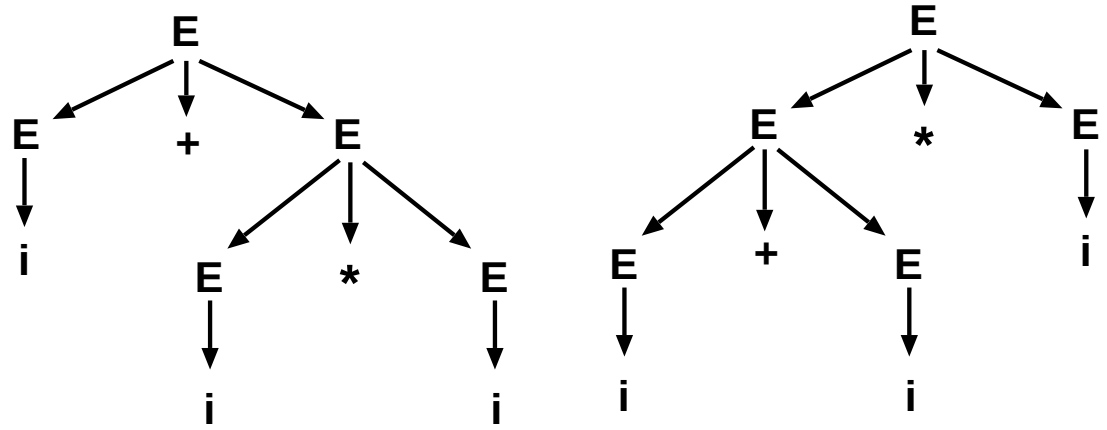
- Problém víceznačnosti gramatik je nerozhodnutelný, tj. neexistuje algoritmus, který by byl schopen v konečném čase rozhodnout, zda daná gramatika je nebo není víceznačná.
- Víceznačnost gramatiky je pokládána za negativní rys (vede k větám, které mají několik interpretací). Na druhé straně může být víceznačná gramatika jednodušší než odpovídající jednoznačná gramatika.

Víceznačnost gramatik

Příklad 4.4 Uvažujme gramatiku $G = (\{E\}, \{+, -, *, /, (,), P, E)$, kde P je množina pravidel

$$E \to E + E \mid E - E \mid E * E \mid E/E \mid (\ E \ ) \mid i$$

Jazyk $L(G)$ je tvořen aritmetickými výrazy s binárními operacemi. Gramatika G je na rozdíl od gramatiky z příkladu 4.2 víceznačná. Vezměme například větu $i + i * i$ a uvažujme všechny možné derivační stromy.



Není jasné, zda první operací bude násobení (derivační strom vlevo), nebo sčítání (derivační strom vpravo).

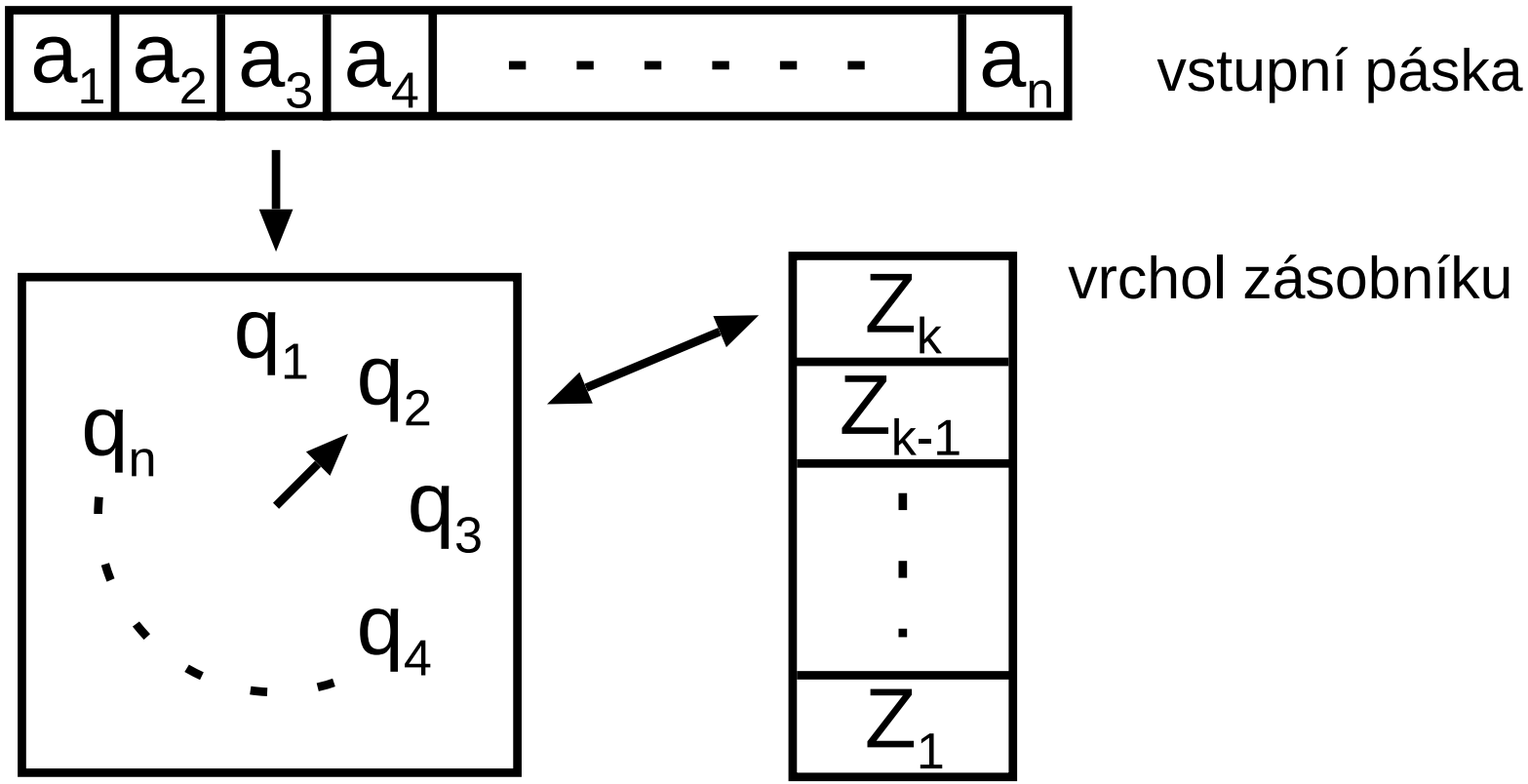
Příklad 4.5 Jednoznačnou gramatikou generující tentýž jazyk je gramatika $G = (\{E, T, F\}, \{+, -, *, /, (,), i\}, P, E)$ s množinou přepisovacích pravidel P definovanou následujícím způsobem:

$E \to T \mid E + T \mid E - T$
$T \to F \mid T * F \mid T/F$
$F \to (\ E \ ) \mid i$

Zásobníkové automaty

Základní schéma

Schéma zásobníkového automatu:



Základní definice

Definice 4.5 Zásobníkový automat P je n-tice $P = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$

1. Q je konečná množina vnitřních stavů
2. Σ je konečná vstupní abeceda
3. Γ je konečná zásobníková abeceda
4. δ je přechodová funkce ve tvaru $\delta : Q \times (\Sigma \cup \{\varepsilon\}) \times \Gamma \to 2^{Q \times \Gamma^*}$
5. $q_0 \in Q$ je počáteční stav
6. $Z_0 \in \Gamma$ je startovací symbol zásobníku
7. $F \subseteq Q$ je množina koncových stavů

Konfigurace a přechod ZA

Definice 4.6 Nechť $P = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$ je zásobníkový automat. Konfigurací automatu P nazveme trojici $(q, w, \alpha) \in Q \times \Sigma^* \times \Gamma^*$, kde

1. q je přítomný stav vnitřního řízení
2. w je dosud nezpracovaná část vstupního řetězce
3. α je obsah zásobníku ($\alpha = Z_{i_1} Z_{i_2} \dots Z_{i_k}$, Z_{i_1} je vrchol)

Přechod ZA P je binární relace $\vdash_P$ definovaná na množině konfigurací:

$$(q, w, \beta) \vdash_P (q', w', \beta') \stackrel{def}{\iff} w = aw' \wedge \beta = Z\alpha \wedge \beta' = \gamma\alpha \wedge (q', \gamma) \in \delta(q, a, Z),$$

kde $q, q' \in Q$, $a \in \Sigma \cup \{\varepsilon\}$, $w, w' \in \Sigma^*$, $Z \in \Gamma$ a $\alpha, \beta, \beta', \gamma \in \Gamma^*$.

- Je-li $a = \varepsilon$, pak odpovídající přechod nazýváme ε-přechodem.
- Relace $\vdash_P^i$, $\vdash_P^*$, $\vdash_P^+$ jsou definovány obvyklým způsobem.
- Platí-li pro řetězec $w \in \Sigma^*$ relace $(q_0, w, Z_0) \vdash_P^* (q, \varepsilon, \gamma)$, kde $q \in F$ a $\gamma \in \Gamma^*$, pak říkáme, že w je přijímán zásobníkovým automatem P (q_0, w, Z_0), resp. (q, ε, γ) je počáteční, resp. koncová konfigurace.
- Definujeme jazyk přijímaný zásobníkovým automatem P: $L(P) = \{w | (q_0, w, Z_0) \vdash_P^* (q, \varepsilon, \gamma) \wedge q \in F\}$.

Příklad zásobníkového automatu

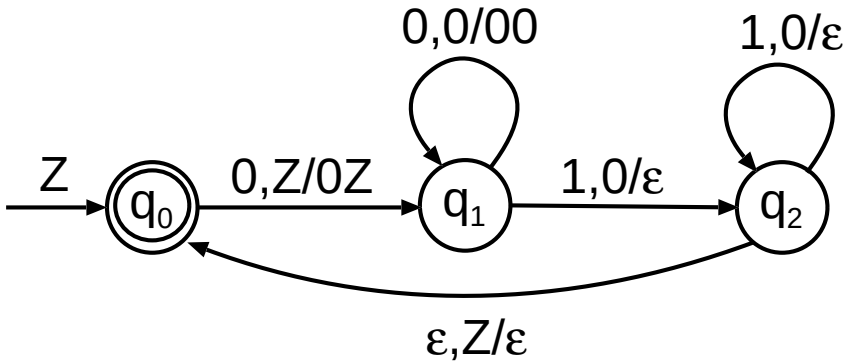
Příklad 4.6 Sestrojme zásobníkový automat, který přijímá jazyk $L = \{0^n 1^n \mid n \geq 0\}$.
– Řešením je $P = (\{q_0, q_1, q_2\}, \{0, 1\}, \{Z, 0\}, \delta, q_0, Z, \{q_0\})$, kde

$\delta(q_0, 0, Z) = \{(q_1, 0Z)\}$
$\delta(q_1, 0, 0) = \{(q_1, 00)\}$
$\delta(q_1, 1, 0) = \{(q_2, \varepsilon)\}$
$\delta(q_2, 1, 0) = \{(q_2, \varepsilon)\}$
$\delta(q_2, \varepsilon, Z) = \{(q_0, \varepsilon)\}$

– Při přijetí řetězce 0011 projde P těmito konfiguracemi:

$(q_0, 0011, Z) \vdash (q_1, 011, 0Z) \vdash (q_1, 11, 00Z) \vdash (q_2, 1, 0Z) \vdash (q_2, \varepsilon, Z) \vdash (q_0, \varepsilon, \varepsilon)$

– Zásobníkové automaty lze také popsat přechodovým diagramem, jak je ilustrováno níže na právě sestrojeném automatu P:



Návrh složitějších automatů

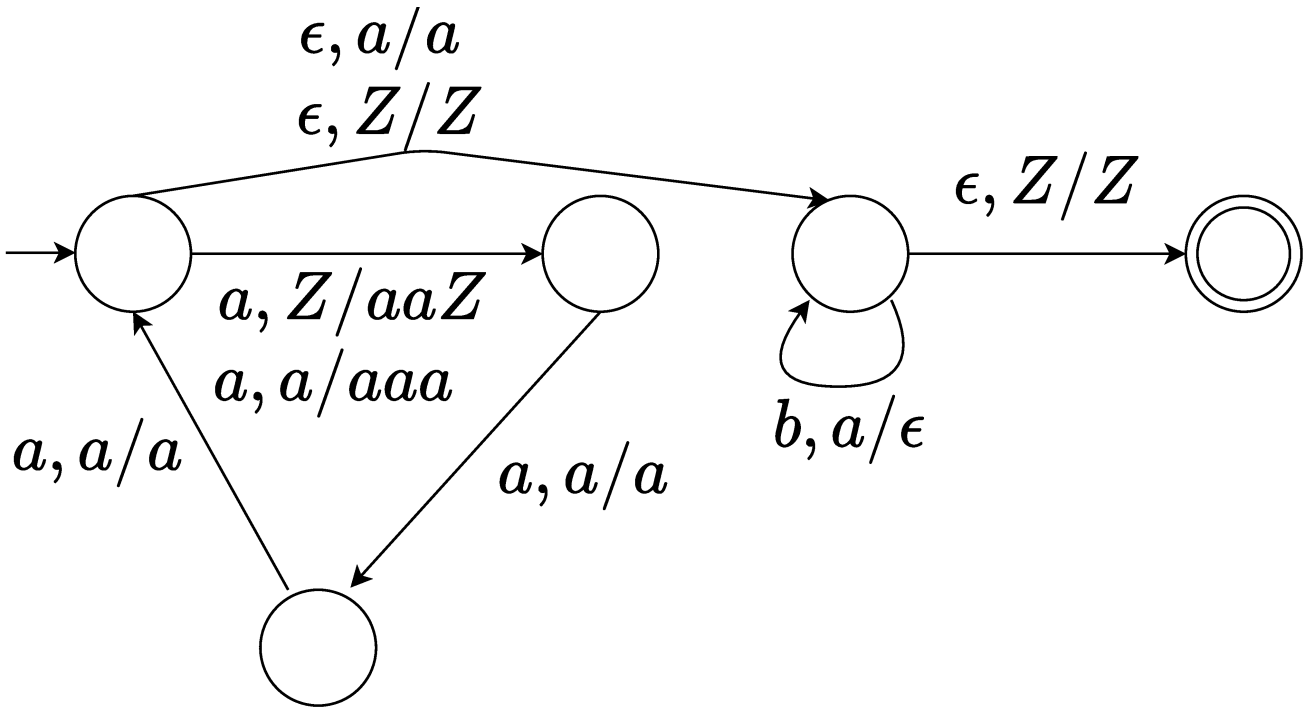
- Pokročilejší práce se zásobníkem.

Zkonstruujte zásobníkový automat pro jazyk $L = \{a^{3n}b^{2n} \mid n \geq 0\}$

Návrh složitějších automatů

- Pokročilejší práce se zásobníkem.

Zkonstruujte zásobníkový automat pro jazyk $L = \{a^{3n}b^{2n} \mid n \geq 0\}$



Varianty zásobníkových automatů

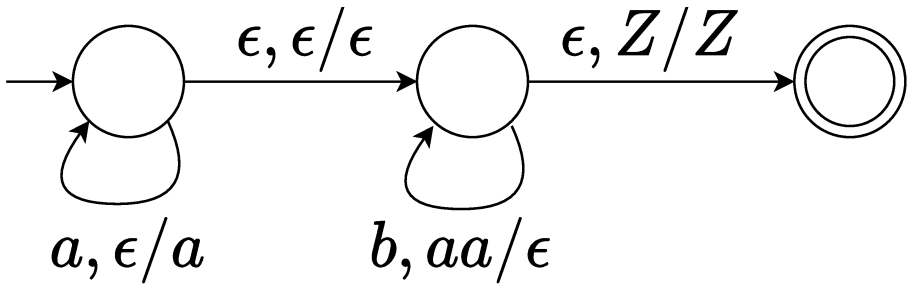
Rozšířený zásobníkový automat

Definice 4.7 Rozšířený zásobníkový automat P je sedmice $P = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$, kde δ je přechodová funkce definovaná takto:

$$\delta : Q \times (\Sigma \cup \{\varepsilon\}) \times \Gamma^* \to 2^{Q \times \Gamma^*}$$

Ostatní složky mají stejný význam jako v definici 4.5.

Příklad 4.7 Zkonstruujte RZA pro jazyk $L = \{a^{2n}b^n \mid n \geq 0\}$



Ekvivalence RZA a ZA

Věta 4.1 Nechť $P = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$ je rozšířený zásobníkový automat. Pak existuje zásobníkový automat P_1 takový, že $L(P_1) = L(P)$.

Důkaz. Položme $m = max\{|\alpha| \mid \delta(q, a, \alpha) \neq \emptyset$ pro nějaké $q \in Q$, $a \in \Sigma \cup \{\varepsilon\}$ a $\alpha \in \Gamma^*\}$.

Zásobníkový automat P_1 budeme konstruovat tak, aby simuloval automat P.

Protože automat P neurčuje přechody podle vrcholu zásobníku, ale podle vrcholového řetězce zásobníku, bude automat P_1 ukládat m vrcholových symbolů v jakési vyrovnávací paměti řídící jednotky tak, aby na počátku každého přechodu věděl, jakých m vrcholových symbolů je v zásobníku automatu P.

Nahrazuje-li automat P k vrcholových symbolů řetězcem délky l, pak se totéž provede ve vyrovnávací paměti automatu P_1.

Jestliže $l < k$, pak P_1 realizuje $k - l$ ε-přechodů, které přesouvají $k - l$ symbolů z vrcholu zásobníku do vyrovnávací paměti. Automat P_1 pak může simulovat další přechod automatu P.

Je-li $l \geq k$ pak se symboly přesouvají z vyrovnávací paměti do zásobníku.

Formálně můžeme konstrukci zásobníkového automatu P_1 popsat takto:
$P_1 = (Q_1, \Sigma_1, \Gamma_1, \delta_1, Z_1, F_1)$, kde

1. $Q_1 = \{[q, \alpha] | q \in Q, \alpha \in \Gamma_1^* \wedge 0 \leq |\alpha| \leq m\}$
2. $\Gamma_1 = \Gamma \cup \{Z_1\}$
3. Zobrazení δ_1 je definováno takto:
   (a) Předpokládejme, že $\delta(q, a, X_1 \ldots X_k)$ obsahuje $(r, Y_1 \ldots Y_l)$.
      i. Jestliže $l \geq k$, pak pro všechna $Z \in \Gamma_1$ a $\alpha \in \Gamma_1^*$ taková, že $|\alpha| = m - k$, pak $\delta_1([q, X_1 \ldots X_k \alpha], a, Z)$ obsahuje $([r, \beta], \gamma Z)$, kde $\beta\gamma = Y_1 \ldots Y_l \alpha$ a $|\beta| = m$.
      ii. Je-li $l < k$, pak pro všechna $Z \in \Gamma_1$ a $\alpha \in \Gamma_1^*$ taková, že $|\alpha| = m - k$, pak $\delta_1([q, X_1 \ldots X_k \alpha], a, Z)$ obsahuje $([r, Y_1 \ldots Y_l \alpha Z], \varepsilon)$.
   (b) Pro všechna $q \in Q, Z \in \Gamma_1$ a $\alpha \in \Gamma_1^*$ taková, že $|\alpha| < m$, platí $\delta_1([q, \alpha], \varepsilon, Z) = \{([q, \alpha Z], \varepsilon)\}$. Tato pravidla vedou k naplnění vyrovnávací paměti.

4. $q_1 = [q_0, Z_0, Z_1^{m-1}]$. Vyrovnávací paměť obsahuje na počátku symbol Z_0 na vrcholu a $m-1$ symbolů Z_1 na dalších místech. Symboly Z_1 jsou speciální znaky pro označení dna zásobníku.

5. $F_1 = \{[q, \alpha] \mid q \in F, \alpha \in \Gamma_1^*\}$
   Lze ukázat, že $(a, aw, X_1 \dots X_k X_{k+1} \dots X_n) \vdash_P (r, w, Y_1 \dots Y_l X_{k+1} \dots X_n)$ platí, právě když $([q, \alpha], aw, \beta) \vdash_{P_1}^+ ([r, \alpha'], w, \beta')$ kde
   $\alpha\beta = X_1 \dots X_n Z_1^m$
   $\alpha'\beta' = Y_1 \dots Y_l X_{k+1} \dots X_n Z_1^m$
   $|\alpha| = |\alpha'| = m$
   a mezi těmito dvěma konfiguracemi automatu P_1 není žádná konfigurace, ve které by druhý člen stavu (vyrovnávací paměť) měl délku m.

Tedy relace $(q_0, w, Z_0) \vdash_P (q, \varepsilon, \alpha)$ pro $q \in F, \alpha \in \Gamma^*$ platí, právě když $([q_0, Z_0, Z_1^{m-1}], w, Z_1) \vdash_{P_1}^* ([q, \beta], \varepsilon, \gamma)$, kde $|\beta| = m$ a $\beta\gamma = \alpha Z_1^m$. Tedy $L(P) = L(P_1)$. □

ZA přijímající s vyprázdněním zás.

Definice 4.8 Zásobníkový automat nebo rozšířený zásobníkový automat $P = (Q, \Sigma, \Gamma, \delta, q_0, Z, \emptyset)$ přijímá s vyprázdněním zásobníku, pokud

$$L(P) = \{w \mid (q_0, w, Z_0) \vdash^* (q, \varepsilon, \varepsilon), q \in Q\}$$

Věta 4.2 Ke každému ZA (resp. RZA) P existuje ZA (resp. RZA) P', který přijímá s vyprázdněním zásobníku, takový, že $L(P) = L(P')$.

Důkaz. (Hlavní myšlenka) Opět budeme konstruovat automat P' tak, aby simuloval automat P. Kdykoli automat P dospěje do koncového stavu, přejde automat P' do speciálního stavu q_ε, který způsobí vyprázdnění zásobníku. Musíme však uvážit situaci, kdy automat P je v konfiguraci s prázdným zásobníkem, nikoli však v koncovém stavu. Abychom zabránili případům, že automat P' přijímá řetězec, který nemá být přijat, přidáme k zásobníkové abecedě automatu P' znak, jenž bude označovat dno zásobníku a může být vybrán pouze tehdy, je-li automat P' ve stavu q_ε. □

Ekvivalence bezkontextových jazyků a jazyků přijímaných zásobníkovým automatem

Označme třídu všech jazyků přijímaných zásobníkovými automaty symbolem $\mathcal{L}_P$. Dokážeme, že $\mathcal{L}_2 = \mathcal{L}_P$ postupem analogickým s důkazem tvrzení $\mathcal{L}_3 = \mathcal{L}_M$. Ukážeme tedy, že

- ke každé bezkontextové gramatice existuje ekvivalentní zásobníkový automat, tj. $\mathcal{L}_2 \subseteq \mathcal{L}_P$
- a ke každému zásobníkovému automatu existuje ekvivalentní gramatika typu 2, tj. $\mathcal{L}_P \subseteq \mathcal{L}_2$

Pro důkaz inkluze $\mathcal{L}_2 \subseteq \mathcal{L}_P$ zkonstruujeme (redundantně) automaty modelující oba typy syntaktické analýzy příslušného bezkontextového jazyka.

$\mathcal{L}_2 \subseteq \mathcal{L}_P$ – Analýza shora dolů

Věta 4.3 Nechť $G = (N, \Sigma, P, S)$ je bezkontextová gramatika. Pak existuje zásobníkový automat P, který přijímá s vyprázdněním zásobníku takový, že $L(G) = L(P)$.

Důkaz. Zásobníkový automat P vytvoříme tak, aby vytvářel levou derivaci vstupního řetězce v gramatice G (modeloval syntaktickou analýzu shora dolů). Nechť P je ZA:

$P = (\{q\}, \Sigma, N \cup \Sigma, \delta, q, S, \emptyset)$, kde δ je určena takto:

- Je-li $A \to \alpha$ pravidlo z P, pak $(q, \alpha) \in \delta(q, \varepsilon, A)$
- $\delta(q, a, a) = \{(q, \varepsilon)\}$ pro všechna $a \in \Sigma$

Indukcí lze dokázat ekvivalenci

$$A \Rightarrow^m w \;\Leftrightarrow\; (q, w, A) \vdash^n (q, \varepsilon, \varepsilon),\ m, n \geq 1,\ w \in \Sigma^*$$

což pro případ $A = S$ znamená $L(G) = L(P)$.

□

Příklad 4.8 Ke gramatice

$$G = (\{S\}, \{0, 1\}, \{S \to 0S1, S \to 01\}, S),$$

sestrojíme zásobníkový automat P, který modeluje syntaktickou analýzu shora dolů:

$$P = (\{q\}, \{0, 1\}, \{S, 0, 1\}, \delta, q, S, 0), \text{ kde}$$

$$\begin{aligned}
&\delta(q, \varepsilon, S) = \{(q, 0S1), (q, 01)\} \\
&\delta(q, 0, 0) = \{(q, \varepsilon)\} \\
&\delta(q, 1, 1) = \{(q, \varepsilon)\}
\end{aligned}$$

Skutečně, např. derivaci

$$S \Rightarrow 0S1 \Rightarrow 00S11 \Rightarrow 000111$$

odpovídá posloupnost přechodů automatu P:

$(q, 000111, S) \vdash (q, 000111, 0S1) \vdash (q, 00111, S1) \vdash (q, 00111, 0S11) \vdash (q, 0111, S11) \vdash$
$(q, 0111, 0111) \vdash (q, 111, 111) \vdash (q, 11, 11) \vdash (q, 1, 1) \vdash (q, \varepsilon, \varepsilon)$

$\mathcal{L}_2 \subseteq \mathcal{L}_P$ – Analýza zdola nahoru

Věta 4.4 Nechť $G = (N, \Sigma, P, S)$ je bezkontextová gramatika. Pak lze ke gramatice G sestrojit RZA P takový, že $L(G) = L(P)$.

Důkaz. RZA P sestrojme tak, aby modeloval syntaktickou analýzu zdola nahoru. Nechť P je RZA

$$P = (\{q, r\}, \Sigma, N \cup \Sigma \cup \{\#\}, \delta, q, \#, \{r\})$$

kde δ je určena takto:

1. Je-li $A \to \alpha$ pravidlo z P, pak $\delta(q, \varepsilon, \alpha)$ obsahuje (q, A). - redukce
2. $\delta(q, a, \varepsilon) = \{(q, a)\}$ pro všechna $a \in \Sigma$ - shift
3. $\delta(q, \varepsilon, S\#) = \{(r, \varepsilon)\}$

Indukcí lze opět dokázat $L(G) = L(P)$.

□

Příklad 4.9 Na gramatiku

$$G = (\{S\}, \{0, 1\}, \{S \to 0S1, S \to 01\}, S)$$

aplikujeme nyní větu 5.4. Výsledný RZA bude mít tvar:

$$P = (\{q, r\}, \{0, 1\}, \{0, 1, S, \#\}, \delta, q, \#, \{r\})$$

kde δ je definována takto

$$\begin{array}{ll} \delta(q, \varepsilon, 0S1) = \{(q, S)\} & \text{redukce} \\ \delta(q, \varepsilon, 01) = \{(q, S)\} & \text{redukce} \\ \delta(q, 0, \varepsilon) = \{(q, 0)\} & \text{shift} \\ \delta(q, 1, \varepsilon) = \{(q, 1)\} & \text{shift} \\ \delta(q, \varepsilon, \#S) = \{(r, \varepsilon)\} & \end{array}$$

Derivaci $S \Rightarrow 0S1 \Rightarrow 0011$ odpovídá posloupnosti konfigurací $(q, 0011, \#) \vdash (q, 011, \#0) \vdash (q, 11, \#00) \vdash (q, 1, \#001) \vdash (q, 1, \#0S) \vdash (q, \varepsilon, \#0S1) \vdash (q, \varepsilon, \#S) \vdash (r, \varepsilon, \varepsilon)$

Poznámka 4.1 Vrchol zásobníku uvádíme, pro lepší čitelnost, vpravo

$*\mathcal{L}_P \subseteq \mathcal{L}_2*$

Věta 4.5 Nechť $P = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, \emptyset)$ je zásobníkový automat přijímající s vyprázdněním zásobníku. Pak existuje gramatika $G = (N, \Sigma, P, S)$ taková, že

$$L(P) = L(G).$$

Důkaz. Gramatiku G budeme definovat formálně takto:

- $N = \{[qZr] \mid q, r \in Q, Z \in \Gamma\} \cup \{S\}$
- Jestliže $(r, X_1 X_2 \dots X_k) \in \delta(q, a, Z)$, $k \geq 1$, pak k P přidej pravidla tvaru

  $$[qZs_k] \to a[rX_1s_1][s_1X_2s_2] \dots [s_{k-1}X_ks_k]$$

  pro každou posloupnost stavů $s_1, s_2, \dots, s_k$ z množiny Q
- Jestliže $(r, \varepsilon) \in \delta(q, a, Z)$, pak k P přidej pravidlo $[qZr] \to a$ (pro $a \in \Sigma \cup \{\varepsilon\}$)
- Pro každý stav $q \in Q$ přidej k P pravidlo $S \to [q_0 Z_0 q]$

Indukcí lze dokázat $S \Rightarrow [q_0 Z_0 q] \Rightarrow^+ w$ právě když $(q_0, w, Z_0) \vdash^* (q, \varepsilon, \varepsilon)$

□

Ekvivalence $\mathcal{L}_2 = \mathcal{L}_P$

Věta 4.6 **Třída bezkontextových jazyků a třída jazyků přijímaných zásobníkovými automaty jsou totožné.**

Důkaz. Přímý důsledek vět 4.4 a 4.5. □

Transformace bezkontextových gramatik

Ekvivalentní gramatiky

Definice 4.9 Nechť G_1 a G_2 jsou gramatiky libovolného typu Chomského klasifikace. G_1 a G_2 jsou ekvivalentní, pokud $L(G_1) = L(G_2)$.

Věta 4.7 Nechť $G = (N, \Sigma, P, S)$ je gramatika, $A \to \alpha B \beta$, $B \in N$, $\alpha, \beta \in (N \cup \Sigma)^*$ je pravidlo z P a nechť $B \to \gamma_1 \mid \gamma_2 \mid \ldots \mid \gamma_n$ jsou všechna B-pravidla z P. Pak gramatika $G' = (N, \Sigma, P', S)$ kde

$$P' = P \setminus \{A \to \alpha B \beta\} \cup \{A \to \alpha\gamma_1\beta, A \to \alpha\gamma_2\beta, \ldots, A \to \alpha\gamma_n\beta\}$$

je ekvivalentní s gramatikou G.

Důkaz. Na cvičení. □

Příklad 4.10

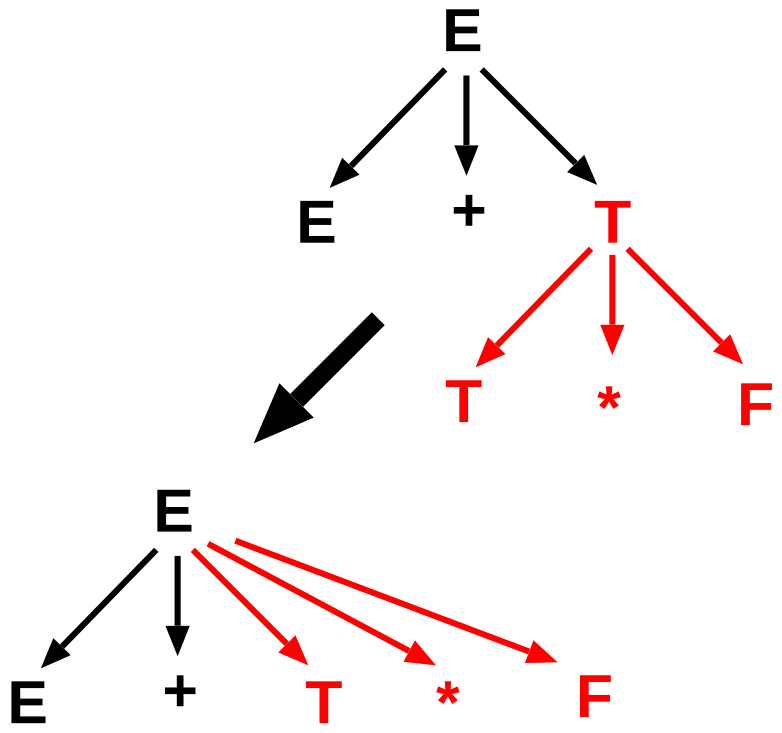
Gramatiky s pravidly

$$E \to E + T \mid T$$
$$T \to T * F \mid F$$
$$F \to (E) \mid i$$

resp.

$$E \to E + T * F \mid E + F \mid T$$
$$T \to T * F \mid F$$
$$F \to (E) \mid i$$

jsou ekvivalentní.



Nedostupné a zbytečné symboly

Definice 4.10 Nechť $G = (N, \Sigma, P, S)$ je gramatika a $X \in N \cup \Sigma$ symbol. Říkáme, že symbol X je nedostupný v G, jestliže v G neexistuje derivace $S \Rightarrow^* \alpha X \beta$ pro nějaké $\alpha, \beta \in (N \cup \Sigma)^*$. Symbol X nazýváme zbytečný v G, jestliže v G neexistuje derivace tvaru $S \Rightarrow^* \alpha X \beta \Rightarrow^* zxy$ pro nějaké $\alpha, \beta \in (N \cup \Sigma)^*$ a $zxy \in \Sigma^*$.

Příklad 4.11 Uvažujme gramatiku $G = (\{S, A, B\}, \{a, b\}, P, S)$ s pravidly:

$S \to SB \mid a$
$A \to b$
$B \to Ba$

Symboly A, B, b jsou zbytečné. Symboly A, b jsou nedostupné.

Poznámka 4.2 $G' = (\{S\}, \{a\}, \{S \to a\}, S)$ je ekvivalentní s G.

Nonterminály generující terminální řetězce

Algoritmus 4.1 Výpočet množiny nonterminálů generujících terminální řetězce
Vstup: Gramatika $G = (N, \Sigma, P, S)$.
Výstup: Množina $N_t = \{A \in N \mid A \Rightarrow^+ w, w \in \Sigma^*\}$.
Metoda: Počítáme množiny $N_0, N_1, N_2, \ldots$ rekurentně takto:

1. N_0 := $\emptyset, i = 1$
2. N_i := $\{A \mid A \to \alpha$ je v P a $\alpha \in (N_{i-1} \cup \Sigma)^*\}$
3. Je-li $N_i \neq N_{i-1}$, i := $i + 1$ a vrať se k (2). Je-li $N_i = N_{i-1}$, polož $N_t = N_i$ a skonči.

Dostupné symboly

Algoritmus 4.2 Výpočet množiny dostupných symbolů
Vstup: Gramatika $G = (N, \Sigma, P, S)$.
Výstup: Množina $V = \{X \in N \cup \Sigma \mid S \Rightarrow^* \alpha X \beta, \alpha, \beta \in (N \cup \Sigma)^*\}$.
Metoda:

1. V_0 := $\{S\}, i = 1$
2. V_i := $\{X \mid A \to \alpha X \beta$ je v P a $A \in V_{i-1}\} \cup V_{i-1}$
3. Je-li $V_i \neq V_{i-1}$, i := $i + 1$ a vrať se k (2). Je-li $V_i = V_{i-1}$, polož $V = V_i$ a skonči.

Odstranění zbytečných symbolů

Algoritmus 4.3 Odstranění zbytečných symbolů
Vstup: Gramatika $G = (N, \Sigma, P, S)$.
Výstup: Gramatika $G' = (N', \Sigma', P', S)$ bez zbytečných symbolů, $L(G) = L(G')$.
Metoda:

1. Aplikací algoritmu 4.1 na G vypočti množinu N_t.
2. Polož $\overline{G} = (N_t \cup \{S\}, \Sigma, \overline{P}, S)$, kde
   $\overline{P} = \{A \to \alpha | (A \to \alpha) \in P \wedge A \in N_t \wedge \alpha \in (N_t \cup \Sigma)^*\}$.
3. Aplikací algoritmu 4.2 na $\overline{G}$ vypočti množinu V.
4. Výslednou gramatiku G' sestroj takto:
   (a) $N' = N_t \cap V$
   (b) $\Sigma' = \Sigma \cap V$
   (c) $P' = \{A \to \alpha | (A \to \alpha) \in P \wedge A \in N' \wedge \alpha \in V^*\}$

Poznámka: Sjednocení $N_t \cup \{S\}$ v bodě 2 řeší případ, kdy $L(G) = \emptyset$ a $N_t = \emptyset$, ovšem gramatika musí mít svůj startovací symbol.

Příklad 4.12 Uvažujme gramatiku
$G = (\{S, A, B\}, \{a, b\}, \{S \to a, S \to A, A \to AB, B \to b\}, S)$.

1. $N_0 = \emptyset$, $N_1 = \{S, B\}$, $N_2 = N_1 = N_t = \{S, B\}$
2. $\overline{G} = (\{S, B\}, \{a, b\}, \{S \to a, B \to b\}, S)$
3. $V_0 = \{S\}$, $V_1 = \{S, a\}$, $V_2 = V_1 = V = \{S, a\}$
4. $G' = (\{S\}, \{a\}, \{S \to a\}, S)$.

Poznámka 4.3 Pořadí kroků 2. a 4. je významné.

Odstranění ε-pravidel

Definice 4.11 G je gramatikou bez ε-pravidel, jestliže neobsahuje žádné ε-pravidlo (pravidlo tvaru $A \to \varepsilon$), nebo, pokud $\varepsilon \in L(G)$, potom obsahuje jediné ε-pravidlo $S \to \varepsilon$ a S se pak nevyskytuje na pravé straně žádného přepisovacího pravidla.

Algoritmus 4.4 Transformace na gramatiku bez ε-pravidel
Vstup: Gramatika $G = (N, \Sigma, P, S)$.
Výstup: Gramatika $G' = (N', \Sigma, P', S')$ bez ε-pravidel ekvivalentní s G.
Metoda:

1. Vypočítej množinu $N_\varepsilon = \{A \in N \mid A \Rightarrow^+ \varepsilon\}$
2. Každé pravidlo z P, které není ε-pravidlem, uvažuj ve tvaru $A \to \alpha_0 B_1 \alpha_1 B_2 \dots B_k \alpha_k$, kde $B_i \in N_\varepsilon, \alpha_i \in (N \setminus N_\varepsilon \cup \Sigma)^*$ pro $i = 1, \dots, k$
   Toto pravidlo nahraď množinou pravidel, které vzniknou všemi možnými substitucemi $B_i \rightsquigarrow B_i$ a $B_i \rightsquigarrow \varepsilon$ pro $i = 1, \dots, k$ (to jest substitucemi, kdy nonterminály z N_ε jsou alternativně ponechávány a vypouštěny). Počet těchto substitucí (nových pravidel) je zřejmě 2^k.
3. Všechna ε-pravidla vypusť.
4. Pokud $S \in N_\varepsilon$, pak $N' = N \cup \{S'\}$, kde S' je nový nonterminální symbol, a přidej pravidla $S' \to \varepsilon \mid S$, v opačném případě $N' = N$, $S' = S$

Příklad 4.13 Uvažujme gramatiku $G = (\{A, B, C\}, \{a, b, c\}, P, A)$ s pravidly:
$A \to AbAcBC \mid \varepsilon \mid a$
$B \to b \mid \varepsilon$
$C \to c \mid \varepsilon$

1. $N_\varepsilon = \{A, B, C\}$
2. $A \to AbAcBC$
   $A \to \ bAcBC$
   $A \to Ab\ cBC$
   $A \to \ b\ cBC$
   $\vdots$
   $A \to \ bc$
   $A \to a$
   $B \to b$
   $C \to c$
3. $A' \to \varepsilon$
   $A' \to A$

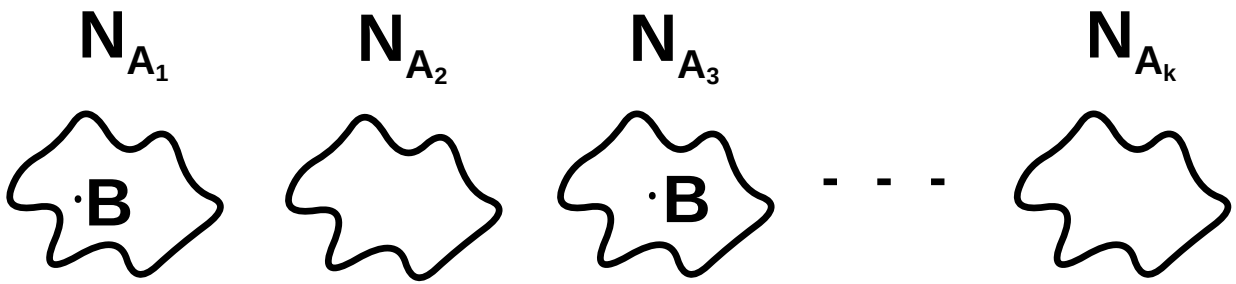
A' je nový startovací symbol.

Odstranění jednoduchých pravidel

Definice 4.12 Pravidlo tvaru $A \to B$, kde $A, B \in N$ nazýváme jednoduché pravidlo.

Algoritmus 4.5 Transformace na gramatiku bez jednoduchých pravidel
Vstup: Gramatika $G = (N, \Sigma, P, S)$ bez ε-pravidel.
Výstup: Gramatika $G' = (N, \Sigma, P', S)$ bez jednoduchých pravidel ekvivalentní s G.
Metoda:

1. Pro všechny $A \in N$ vypočítej množinu $N_A = \{B \in N \mid A \Rightarrow^* B\}$, polož $P' := \emptyset$.
2. Nechť $B \to \alpha$, $\alpha \notin N$ je pravidlo z P. Potom k P' přidej nová pravidla $A_i \to \alpha$ pro všechny A_i, kde $B \in N_{A_i}$.
3. Výsledná množina pravidel P' tvoří všechna pravidla gramatiky G' (neobsahuje jednoduchá pravidla).



Nová pravidla: $A_1 \to \alpha$ $A_3 \to \alpha$

Příklad 4.14 Uvažujme gramatiku $G = (\{E, T, F\}, \{i, +, *, (,)\}, P, E)$ s pravidly:
$E \to E + T \mid T$
$T \to T * F \mid F$
$F \to (E) \mid i$

1. Nalezneme množiny N_A pro všechny $A \in N$:
   $N_E = \{E, T, F\} \quad N_T = \{T, F\} \quad N_F = \{F\}$
2. Doplňujeme nová pravidla a vypouštíme jednoduchá pravidla:
   $E \to E + T \mid T * F \mid (E) \mid i$
   $T \to T * F \mid (E) \mid i$
   $F \to (E) \mid i$

Cyklus

Definice 4.13 Nechť $G = (N, \Sigma, P, S)$ je gramatika, $A \in N$. Gramatika G obsahuje cyklus, jestliže $A \Rightarrow^+ A$.

Věta 4.8 Jestliže gramatika $G = (N, \Sigma, P, S)$ obsahuje cyklus v nonterminálu A, $A \in N$ a jestliže existuje derivace

$$S \Rightarrow^* \alpha A \beta \Rightarrow^+ w, w \in \Sigma^*, \alpha, \beta \in (N \cup \Sigma)^*$$

pak G je víceznačná.

Důkaz. Existuje-li derivace

$$S \Rightarrow^* \alpha A \beta \Rightarrow^+ w$$

pak vzhledem k existenci cyklu $A \Rightarrow^+$ existuje i derivace

$$S \Rightarrow^* \alpha A \beta \Rightarrow \alpha \gamma_1 \beta \Rightarrow \alpha \gamma_2 \beta \Rightarrow \ldots \Rightarrow \alpha A \beta \Rightarrow^+ w$$

Těmto derivacím přísluší různé derivační stromy. □

Zdroje cyklu

- Jednoduchá pravidla (tvaru $A \to B$), např.

$$A \Rightarrow B \Rightarrow C \Rightarrow A$$

  v důsledku pravidel $A \to B$, $B \to C$, $C \to A$

- ε-pravidla, např.

$$A \Rightarrow AB \Rightarrow A$$

  v důsledku pravidla $B \to \varepsilon$

Vlastní gramatika

Definice 4.14 Gramatika bez zbytečných symbolů, ε-pravidel a bez cyklů se nazývá vlastní gramatikou.

Věta 4.9 Každá bezkontextová gramatika má ekvivalentní vlastní gramatiku.

Důkaz. Aplikací algoritmů 4.3 a 4.4 odstraníme zbytečné symboly a ε-pravidla. Jestliže po této transformaci existuje v G derivace $A \Rightarrow^+ A$, tj. cyklus, pak jeho příčinou mohou být pouze jednoduchá pravidla a ty lze odstranit aplikací algoritmu 4.5. Na závěr je nutné znovu aplikovat algoritmus 4.3, jelikož po aplikaci algoritmu 4.5 mohou znovu vzniknout nedostupné symboly. □

Odstranění levé rekurze

❖ Základem algoritmu je odstranění přímé levé rekurze, tj. levě rekurzivních pravidel, podle následující transformace:

Věta 4.10 Nechť gramatika G má levě rekurzivní pravidla v nonterminálu A a nechť

$$A \to A\alpha_1 \mid A\alpha_2 \mid \ldots \mid A\alpha_m \mid \beta_1 \mid \beta_2 \mid \ldots \mid \beta_n$$

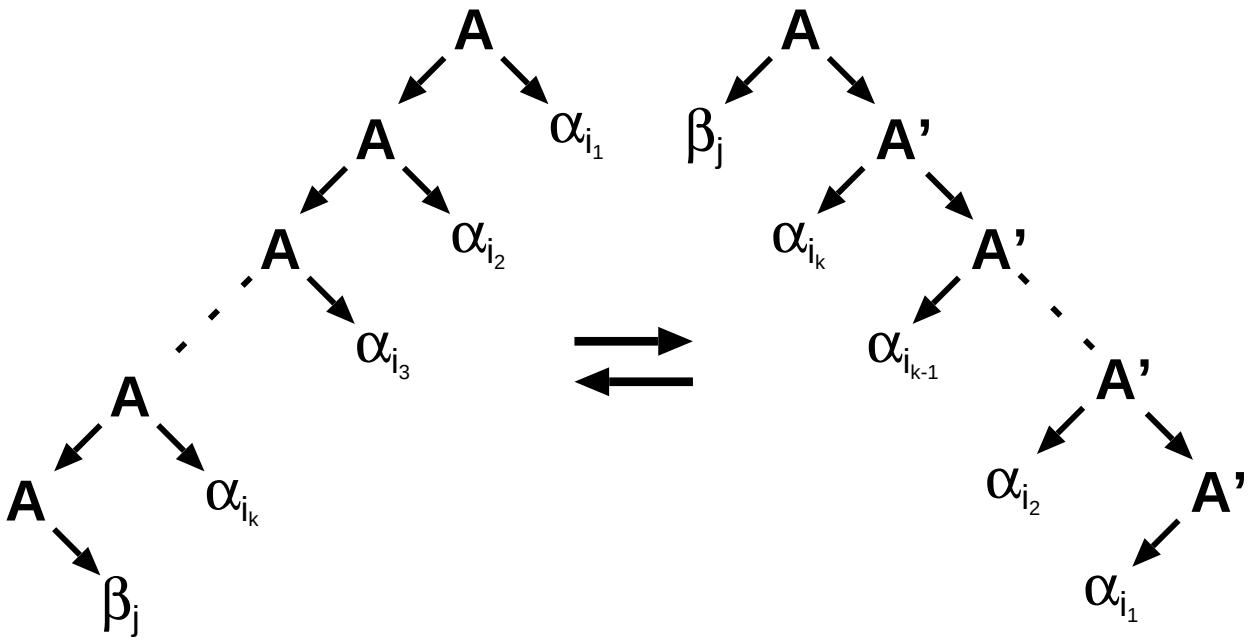
jsou všechna její A-pravidla, přičemž řetězce β_i nezačínají symbolem A. Pak gramatika G', ve které budou tato pravidla nahrazena pravidly:

$$A \to \beta_1 \mid \beta_2 \mid \ldots \mid \beta_n \mid \beta_1 A' \mid \beta_2 A' \mid \ldots \mid \beta_n A'$$

$$A' \to \alpha_1 \mid \alpha_2 \mid \ldots \mid \alpha_1 A' \mid \alpha_2 A' \mid \ldots \mid \alpha_m A'$$

kde A' je nový nonterminál, je ekvivalentní s G.

Důkaz. Uvedená transformace nahrazuje pravidla rekurzivní zleva pravidly, které jsou rekurzivní zprava. Označíme-li jazyky $L_1 = \{\beta_1, \beta_2, \ldots, \beta_n\}$ a $L_2 = \{\alpha_1, \alpha_2, \ldots, \alpha_m\}$, vidíme, že v G lze z nonterminálu A derivovat řetězce tvořící jazyk $L_1 L_2^*$. Právě tyto řetězce můžeme však derivovat z A také v gramatice G'. Efekt popisované transformace ilustruje následující obrázek. □



Příklad 4.15 Uvažujme gramatiku $G = (\{E, T, F\}, \{i, +, *, (,)\}, P, E)$ s pravidly:

$$E \to E + T \mid T$$
$$T \to T * F \mid F$$
$$F \to (\ E \ ) \mid i$$

$$\begin{array}{rcl} E \to E + T \mid T & \longrightarrow & E \to T \mid TE' \\ \alpha_1 = +T, \beta_1 = T & & E' \to +T \mid \ +TE' \\ T \to T * F \mid F & \longrightarrow & T \to F \mid FT' \\ \alpha_1 = *F, \beta_1 = F & & T' \to *F \mid \ *FT' \\ & & F \to (\ E \ ) \mid i \end{array}$$

Odstranění nepřímé levé rekurze

❖ Odstranění nepřímé levé rekurze spočívá v opakovaném aplikování transformace podle věty 4.7 (rozgenerování pravidla) a transformačních vzorců pro odstranění přímé levé rekurze (věta 4.10).

Příklad 4.16 Uvažujme gramatiku $G = (\{S, A, B\}, \{a, b\}, P, S)$ s pravidly:

$S \to AB$
$A \to BS \mid b$
$B \to SA \mid a$

Na pravidlo $B \to SA$ aplikujeme dvakrát větu 4.7:

$B \to ABA$
$B \to BSBA \mid bBA$

Na všechna B-pravidla

$B \to BSBA \mid bBA \mid a$

aplikujme transformaci věty 4.10:

$B \to bBA \mid a \mid bBAB' \mid aB'$
$B' \to SBA \mid SBAB'$

Normální formy bezkontextových gramatik

Chomského normální forma (CNF)

Definice 4.15 Bezkontextová gramatika $G = (N, \Sigma, P, S)$ je v Chomského normální formě, má-li každé pravidlo z P jeden z těchto tvarů:

1. $A \to BC$, kde $A, B, C \in N$
2. $A \to a$, kde $a \in \Sigma$
3. je-li $\varepsilon \in L(G)$, pak $S \to \varepsilon$ je jediné ε-pravidlo a S se nevyskytuje na pravé straně žádného přepisovacího pravidla.

Problém: Nechť $G = (N, \Sigma, P, S)$ je bezkontextová gramatika v CNF a nechť $w \in L(G)$ a $S \Rightarrow_G^p w$. Jaká je délka řetězce w?

Řešení: Označme $|w| = n$. Zřejmě platí

$$p = n + (n - 1) = 2n - 1$$

$$|w| = \frac{p + 1}{2}$$

Věta 4.11 Nechť G je bezkontextová gramatika. Pak existuje gramatika G' v Chomského normální formě taková, že $L(G') = L(G)$.

Důkaz. (Hlavní myšlenka) Gramatiku G převedeme na ekvivalentní vlastní gramatiku bez jednoduchých pravidel.

1. Pravidla tvaru (1), (2) a (3) ponecháme.
2. Pravidla tvaru $A \to X_1 X_2 \dots X_n$, kde $X_i \in (N \cup \Sigma)$ pro $i = 1, \dots, n$, $n > 2$, transformujeme na $A \to X_1' \langle X_2 X_3 \dots X_n \rangle$, kde $\langle X_2 X_3 \dots X_n \rangle$ je nový nonterminál a X_1' je nový nonterminál pokud $X_1 \in \Sigma$, nebo $X_1' = X_1$ v opačném případě.
3. Pravidla tvaru $A \to X_1 X_2$ transformujeme na pravidla $A \to X_1' X_2'$, kde X_i' je nový nonterminál pokud $X_i \in \Sigma$, nebo $X_i' = X_i$ v opačném případě pro $i \in \{1, 2\}$
4. Pro nové nonterminály tvaru $\langle X_1 X_2 \dots X_n \rangle$, $n \geq 2$, zavedeme pravidla $\langle X_1 X_2 \dots X_n \rangle \to X_1' \langle X_2 \dots X_n \rangle$ pro $n > 2$ a $\langle X_1 X_2 \rangle \to X_1' X_2'$ pro $n = 2$, kde $\langle X_2 \dots X_n \rangle$ je nový nonterminál a X_i' je nový nonterminál pokud $X_i \in \Sigma$, nebo $X_i' = X_i$ v opačném případě pro $i \in \{1, 2\}$.
5. Pro nové nonterminály tvaru X_i', kde $X_i \in \Sigma$ přidáme pravidla tvaru $X_i' \to X_i$.

□

Příklad 4.17 Uvažujme gramatiku $G = (\{A, B\}, \{a, b, c\}, P, A)$ s pravidly:

$A \to BAB \mid Ba \mid bc$
$B \to AB \mid a \mid BBB$

Po aplikaci transformací (1.)-(4.) získáme CNF ve tvaru:

$A \to B\langle AB\rangle | Ba' | b'c'$
$B \to AB \mid a \mid B\langle BB\rangle$
$\langle AB\rangle \to AB$
$\langle BB\rangle \to BB$
$a' \to a$
$b' \to b$
$c' \to c$

Greibachové normální forma (GNF)

Definice 4.16 Bezkontextová gramatika $G = (N, \Sigma, P, S)$ je v Greibachové normální formě, je-li G gramatikou bez ε-pravidel a každé pravidlo z P (vyjma případného pravidla $S \to \varepsilon$) má tvar:
$A \to a\alpha$, kde $a \in \Sigma, \alpha \in N^*$