Bezkontextové gramatiky, derivačné stromy, úpravy gramatík

Ing. Viliam Hromada, PhD.

C-510
Ústav informatiky a matematiky
FEI STU

viliam.hromada@stuba.sk

Pre pripomenutie:

- **Regulárne jazyky** - jazyky generovateľné pomocou tzv. regulárnych gramatík.
- **Bezkontextové jazyky** - jazyky generovateľné pomocou tzv. bezkontextových gramatík.

Bezkontextové jazyky

- Regulárne gramatiky (jazyky) sa ľahko spracujú z výpočtového hľadiska, ich použitie je však obmedzené - napr. popis a identifikácia základných symbolov programovacích jazykov.
- Na špecifikáciu zložitejších konštrukcií programovacích jazykov (výrazy, deklarácie, podprogramy) sa používajú bezkontextové gramatiky.
- Pre pripomenutie, bezkontextová gramatika $G = (N, T, P, S)$ obsahuje pravidlá v tvare:

$$A \to \alpha,$$

kde $A \in N, \alpha \in (N \cup T)^*$.

Príklad bezkontextovej gramatiky

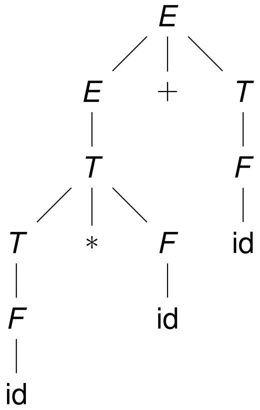
Nech $G = (\{E, T, F\}, \{(,), +, *, \mathrm{id}\}, P, E)$, kde pravidlá P:

$$E \rightarrow E + T \mid T$$
$$T \rightarrow T * F \mid F$$
$$F \rightarrow (E) \mid \mathrm{id}.$$

Zjednodušená gramatika na generovanie výrazov v programovacích jazykoch.
Existuje v nej napr. odvodenie:

$$E \Rightarrow E + T \Rightarrow T + T \Rightarrow T * F + T \Rightarrow F * F + T \Rightarrow \mathrm{id} * F + T$$
$$\Rightarrow \mathrm{id} * F + F \Rightarrow \mathrm{id} * F + \mathrm{id} \Rightarrow \mathrm{id} * \mathrm{id} + \mathrm{id}$$

V bezkontextovej gramatike vieme ku každému odvodeniu kresliť **derivačný strom** (strom odvodenia):



Strom odvodenia

Definícia

*Nech $G = (N, T, P, S)$ je bezkontextová gramatika. Ohodnotený strom sa nazýva **strom odvodenia** (derivačný strom) slova w v gramatike G, ak spĺňa nasledovné podmienky:*

- *Každý vrchol stromu je ohodnotený symbolom z množiny $N \cup T_\varepsilon$, kde $T_\varepsilon = T \cup \{\varepsilon\}$.*
- *Koreň stromu je ohodnotený začiatočným symbolom S.*
- *Ak má nejaký vrchol stromu potomkov, potom je ohodnotený symbolom z množiny N.*
- *Ak $X_1, X_2, ..., X_k$ sú ohodnotenia priamych potomkov vrcholu A, potom v P musí existovať pravidlo $A \rightarrow X_1 X_2 ... X_k$.*
- *Listy stromu sú ohodnotené symbolmi z množiny T_ε.*
- *Slovo $w \in T^*$ sa dá získať zreťazením ohodnotení listov stromu zľava doprava.*

Definícia

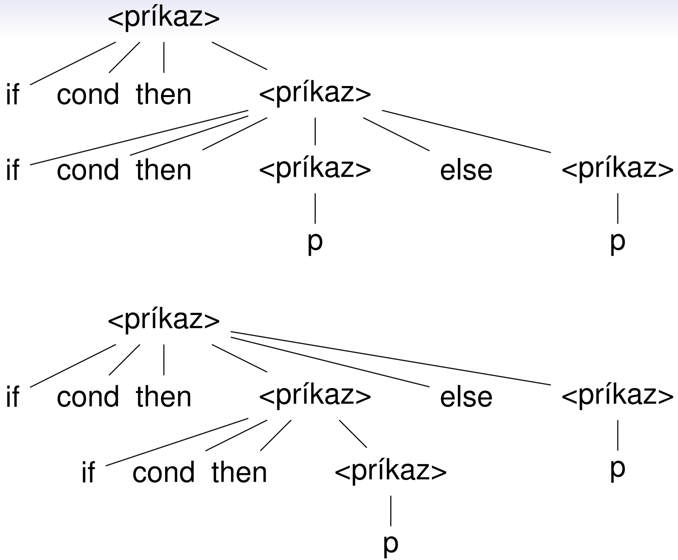
Bezkontextová gramatika G sa nazýva ***jednoznačná****, ak ku každému slovu $w \in L(G)$ existuje jediný strom odvodenia. V opačnom prípadne sa gramatika nazýva* ***nejednoznačná*** *(viacznačná, ambiguous).*

Nejednoznačná gramatika - príklad

Gramatika $G =$ ({<príkaz>}, {**cond, p, if, then, else**}, P, <príkaz>), pravidlá P:

1. <príkaz> $\rightarrow$ **if cond then** <príkaz>
2. <príkaz> $\rightarrow$ **if cond then** <príkaz> **else** <príkaz>
3. <príkaz> $\rightarrow$ **p**

Zjednodušená gramatika reprezentujúca podmienené príkazy, **cond** zastupuje podmienku, **p** zastupuje príkaz. Potom pre slovo (reťazec terminálov):
if cond then if cond then p else p existujú 2 rôzne stromy odvodenia:


<príkaz>
if cond then <príkaz>
if cond then <príkaz> else <príkaz>
p p
<príkaz>
if cond then <príkaz> else <príkaz>
if cond then <príkaz> p
p

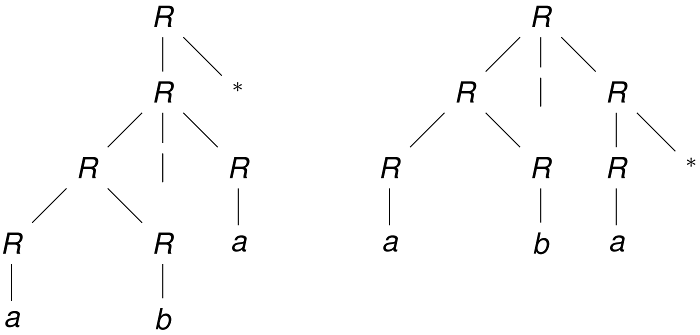
Gramatika pre regulárne výrazy nad $\Sigma = \{a, b\}$

Gramatika $G = \{\{R\}, \{^*, |, a, b\}, P, R\}$, kde pravidlá:

$R \to R \vert R$	(zjednotenie)
$R \to RR$	(zreťazenie)
$R \to R^*$	(iterácia)
$R \to a$	
$R \to b$	

Gramatika pre regulárne výrazy nad $\Sigma = \{a, b\}$

V takejto gramatike by mal regulárny výraz $ab|a^*$ viacero derivačných stromov, napr.



Čo by teoreticky mohlo znamenať nejednoznačnosť príslušného regulárneho výrazu pri jeho počítačovom spracovaní, pretože prvý regulárny výraz vyzerá ako $((ab)|a)^*$ a druhý ako $(ab)|(a^*)$.

Jednoznačné regexy

Našťastie, v tomto prípade existuje aj jednoznačná gramatika pre regulárne výrazy. Uvedieme príklad gramatiky pre regexy, ktoré budú popisovať reťazce nad abecedou $\{a, b\}$. **POZOR!!! Všimnite si, že v tejto gramatike sú zvislá čiara $|$, $\emptyset$, hviezdička či ε terminálne symboly - čo je v protiklade s tým, ako sme ich doteraz používali!).** Je to dané tým, že chceme popísať gramatiku, ktorá generuje reťazce predstavujúce regulárne výrazy a v nich majú tieto symboly svoj špecifický význam.

Gramatika $G =$
$(\{<\text{Regex}>, <\text{Term}>, <\text{Factor}>, <\text{Base}>, <\text{Char}>\}, \{^*, |, a, b, \varepsilon, \emptyset, (,)\}, P, <\text{Regex}>)$,
kde pravidlá sú na ďalšom slajde:

Jednoznačné regexy

<Regex> $\to$ <Term>

<Regex> $\to$ <Term> | <Regex>

<Term> $\to$ <Factor>

<Term> $\to$ <Factor><Term>

<Factor> $\to$ <Base>

<Factor> $\to$ <Base>*

<Base> $\to$ (<Regex>)

<Base> $\to$ <Char>

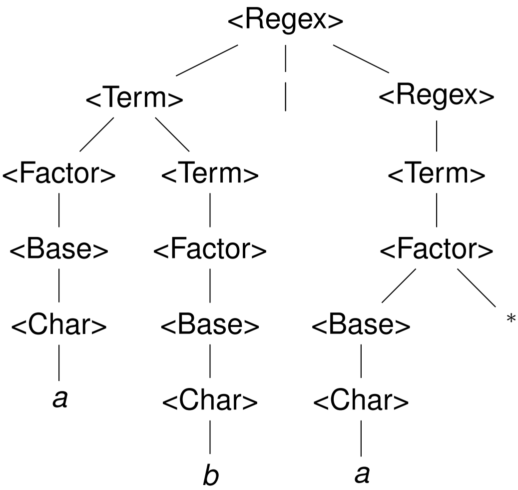
<Char> $\to$ a

<Char> $\to$ b

<Char> $\to$ ε

<Char> $\to$ $\emptyset$

Derivačný strom reťazca $ab \mid a^*$ v tejto gramatike:



Problém určenia nejednoznačnosti gramatiky

- Ak máme zadanú nejakú bezkontextovú gramatiku $G = (N, T, P, S)$, tak problém zistenia, či je gramatika **jednoznačná/nejednoznačná** je **vo všeobecnosti nerozhodnuteľný**!
- To znamená, že **neexistuje** algoritmus, ktorý by pre ľubovoľnú bezkontextovú gramatiku vedel v konečnom čase povedať, či je jednoznačná alebo nejednoznačná!

- Nejednoznačné gramatiky predstavujú problém - ak existujú pre 1 slovo viaceré stromy odvodenia, asi má slovo viacero sémantických interpretácií.
- Je zrejmé, že odvodenie slova a derivačný strom sú úzko prepojené - ak existuje odvodenie, existuje i derivačný strom (a naopak).
- Pri bezkontextových gramatikách si môžeme vybrať, na ktorý neterminál vo vetnej forme aplikujeme príslušné prepisovacie pravidlo v danom kroku odvodenia - bez rizika, že tým ovplyvníme výsledok.

2 špeciálne odvodenia

Príklad: Všimnite si tieto 2 odvodenia:

1. $E \Rightarrow E + T \Rightarrow T + T \Rightarrow T * F + T \Rightarrow F * F + T \Rightarrow \text{id} * F + T \Rightarrow \text{id} * \text{id} + T \Rightarrow \text{id} * \text{id} + F \Rightarrow \text{id} * \text{id} + \text{id}$
2. $E \Rightarrow E + T \Rightarrow E + F \Rightarrow E + \text{id} \Rightarrow T + \text{id} \Rightarrow T * F + \text{id} \Rightarrow T * \text{id} + \text{id} \Rightarrow F * \text{id} + \text{id} \Rightarrow \text{id} * \text{id} + \text{id}$

V odvodení č. 1 sme nahradili vždy prvý neterminál zľava, v odvodení č. 2 sme nahradili vždy prvý neterminál sprava. Oba derivačné stromy by však boli rovnaké (a také isté, ako derivačný strom na slajde č. 5).

Ľavé (pravé) odvodenie

Definícia

Nech G je bezkontextová gramatika. Potom odvodenie $\alpha_0 \Rightarrow \alpha_1 \Rightarrow \ldots \Rightarrow \alpha_k$ v G, kde každý krok odvodenia $\alpha_i \Rightarrow \alpha_{i+1}$ pre $0 \leq i \leq k$ realizujeme tak, že v reťazci α_i nahradíme prvý neterminál zľava, nazývame ***ľavé odvodenie (ľavá derivácia)****; ak nahrádzame prvý neterminál sprava, nazývame* ***pravé odvodenie (pravá derivácia)****. Príslušné vznikajúce vetné formy nazývame* ***ľavé (pravé) vetné formy****. Krok odvodenia v ľavom (pravom) odvodení budeme označovať symbolom $\Rightarrow_l$ ($\Rightarrow_r$).*

Ľavé (pravé) odvodenie

Teda podľa uvedeného značenia:

1. $E \Rightarrow_l E + T \Rightarrow_l T + T \Rightarrow_l T * F + T \Rightarrow_l F * F + T \Rightarrow_l \mathrm{id} * F + T \Rightarrow_l \mathrm{id} * \mathrm{id} + T \Rightarrow_l \mathrm{id} * \mathrm{id} + F \Rightarrow_l \mathrm{id} * \mathrm{id} + \mathrm{id}$
2. $E \Rightarrow_r E + T \Rightarrow_r E + F \Rightarrow_r E + \mathrm{id} \Rightarrow_r T + \mathrm{id} \Rightarrow_r T * F + \mathrm{id} \Rightarrow_r T * \mathrm{id} + \mathrm{id} \Rightarrow_r F * \mathrm{id} + \mathrm{id} \Rightarrow_r \mathrm{id} * \mathrm{id} + \mathrm{id}$

Ak G je jednoznačná gramatika, potom ľavé (pravé) odvodenie každého slova je určené jednoznačne. Túto vlastnosť využívajú **algoritmy hľadajúce deriváciu slova v gramatike - tzv. syntaktické analyzátory.**

Transformácie gramatík

- Gramatiky možno **transformovať** na **ekvivalentné** gramatiky (t.j. generujúce ten istý jazyk), ktoré budú navyše **spĺňať** dodatočné **požiadavky** (napr. na tvar pravidiel).
- Základom pri transformácii gramatík je, aby sa po transformácii **nezmenil jazyk generovaný gramatikou**.

3 základné úpravy

Existujú 3 základné úpravy bezkontextových gramatík:

1. Odstránenie ε-pravidiel
2. Odstránenie jednoduchých pravidiel
3. Odstránenie nadbytočných symbolov

Odstránenie ε-pravidiel

Definícia

Gramatika $G = (N, T, P, S)$ sa nazýva ***gramatika bez ε-pravidiel****, ak množina pravidiel P neobsahuje žiadne ε-pravidlo (také, kde na pravej strane je len prázdny reťazec ε), alebo v P existuje jediné ε-pravidlo $S \to \varepsilon$, pričom začiatočný symbol gramatiky sa nevyskytuje na pravej strane žiadneho pravidla.*

Odstránenie ε-pravidiel

- Každé ε-pravidlo odstránime a zároveň do gramatiky doplníme ďalšie pravidlá, aby sa nezmenil výsledný jazyk generovaný gramatikou.
- Pri dopĺňaní pravidiel postupujeme podľa toho, ktoré neterminály sa mohli počas derivácie prepísať na ε a teda sa mohli „stratiť" z vetnej formy.
- Jediný prípad, kedy akceptujeme ε-pravidlo je v prípade, že $\varepsilon \in L(G)$. V takom prípade doplníme nový počiatočný symbol S', doplníme pravidlá $S' \to S \mid \varepsilon$ a ostatné ε-pravidlá príslušným spôsobom odstránime.

Ilustračný príklad

Gramatiku s ε-pravidlami

$$S \to aSb \mid \varepsilon \mid A$$
$$A \to bAa \mid \varepsilon$$

Upravíme na:

$$S' \to S \mid \varepsilon$$
$$S \to aSb \mid A \mid ab$$
$$A \to bAa \mid ba$$

Odstránenie ε-pravidiel

Uvažujme o gramatike $G = (N, T, P, S)$. Hľadáme množinu neterminálov N_ε, ktoré sa môžu počas odvodzovania „stratiť“, t.j. prepísať na prázdne slovo ε.

$$N_\varepsilon = \{A \mid A \Rightarrow^* \varepsilon, A \in N\} \tag{1}$$

Množinu iteratívne tvoria tie neterminály, z ktorých možno odvodiť len ε.

Odstránenie ε-pravidiel

Vstup: bezkontextová gramatika $G = (N, T, P, S)$
Výstup: množina N_ε (1)

$N_\varepsilon \leftarrow \emptyset$.
opakuj
  $\acute{N}_\varepsilon \leftarrow N_\varepsilon$
  $N_\varepsilon \leftarrow \acute{N}_\varepsilon \cup \{A \mid A \to \alpha \in P \wedge \alpha \in \acute{N}_\varepsilon{}^*\}$
pokiaľ $N_\varepsilon \neq \acute{N}_\varepsilon$

Odstránenie ε-pravidiel

- Po určení N_ε je potrebné vytvoriť množinu pravidiel $\acute{P}$ z množiny P tak, že sa v $\acute{P}$ nebudú vyskytovať ε-pravidlá, ale nezmení sa generovaný jazyk.
- Do $\acute{P}$ môžeme dať tie pravidlá z P, ktoré na pravej strane neobsahujú neterminály z N_ε.
- Pre tie pravidlá z P, ktoré na pravej strane obsahujú neterminály z N_ε, t.j. tvaru:

$$A \to \alpha_0 B_1 \alpha_1 \ldots B_k \alpha_k,$$

kde $\alpha_i \in ((N - N_\varepsilon) \cup T)^*, B_i \in N_\varepsilon$, platí, že každé takéto pravidlo nahradíme v $\acute{P}$ množinou pravidiel tvaru

$$A \to \alpha_0 X_1 \alpha_1 \ldots X_k \alpha_k,$$

kde $X_i = B_i$ alebo $X_i = \varepsilon$ pre $i = 1, \ldots, k$.

Odstránenie ε-pravidiel

- POZOR! Ak by malo vzniknúť pravidlo $A \to \varepsilon$ alebo $A \to A$, tak ho do $\acute{P}$ nepridáme.
- Naviac, ak $\varepsilon \in L(G)$, potom sa štandardne do množiny neterminálov pridáva nový začiatočný neterminál $\acute{S}$ a do pravidiel $\acute{P}$ sa pridajú 2 pravidlá:
    - $\acute{S} \to S$
    - $\acute{S} \to \varepsilon$
- To zistíme o.i. tak, že $S \in N_\varepsilon$.

Odstránenie ε-pravidiel - príklad

Príklad: Odstráňte ε-pravidlá z gramatiky $G = (\{S, A, B, C, D\}, \{a, b, d\}, P, S)$, kde pravidlá P:

$$S \rightarrow aAbB \mid AC \mid BD$$
$$A \rightarrow Aa \mid \varepsilon$$
$$B \rightarrow bBb$$
$$C \rightarrow aCA \mid \varepsilon \mid A$$
$$D \rightarrow d$$

Množina N_ε postupne: $\emptyset, \{A, C\}, \{A, C, S\}$.

Odstránenie ε-pravidiel - príklad (pokr.)

Výsledná gramatika bude $\acute{G} = (\{\acute{S}, S, A, B, C, D\}, \{a, b, d\}, \acute{P}, \acute{S})$, kde $\acute{P}$:

$$\acute{S} \to S \mid \varepsilon$$
$$S \to aAbB \mid AC \mid BD \mid abB \mid A \mid C$$
$$A \to Aa \mid a$$
$$B \to bBb$$
$$C \to aCA \mid A \mid aC \mid aA \mid a$$
$$D \to d$$

pričom boli odstránené pravidlá $A \to \varepsilon$, $C \to \varepsilon$

Odstránenie jednoduchých pravidiel

Definícia

Nech $G = (N, T, P, S)$ je bezkontextová gramatika. Potom pravidlá tvaru $A \rightarrow B$, kde $A, B \in N$ sa nazývajú ***jednoduché pravidlá****.*

Odstránenie jednoduchých pravidiel

- Každé jednoduché pravidlo typu $A \to B$ odstránime a zároveň do gramatiky doplníme ďalšie pravidlá, aby sa nezmenil výsledný jazyk generovaný gramatikou.
- Zjednodušene, ak sa z A dalo odvodiť B, tak po odstránení pravidla $A \to B$ musíme doplniť pravidlá, ktoré majú na ľavej strane A a na pravej strane všetky možné pravé strany pravidiel typu $B \to$

Príklad

Gramatiku s jednoduchými pravidlami

$$S' \to S \mid \varepsilon$$
$$S \to aSb \mid A \mid ab$$
$$A \to bAa \mid ba$$

Upravíme na:

$$S' \to aSb \mid bAa \mid ba \mid ab \mid \varepsilon$$
$$S \to aSb \mid bAa \mid ba \mid ab$$
$$A \to bAa \mid ba$$

Odstránenie jednoduchých pravidiel

Pre každý neterminál A vytvoríme množinu N_A neterminálov, ktoré sa z A dajú odvodiť, t.j.:

$$N_A = \{B \mid A \Rightarrow^* B, B \in N\} \tag{2}$$

Vytvorenie množiny N_A

Vstup: bezkontextová gramatika $G = (N, T, P, S)$ bez ε-pravidiel, neterminál A
Výstup: množina N_A (4)

1: $N_A \leftarrow \{A\}$
2: **opakuj**
3: $\quad \acute{N}_A \leftarrow N_A$;
4: $\quad N_A \leftarrow \acute{N}_A \cup \{C \mid B \rightarrow C \in P \wedge B \in \acute{N}_A\}$
5: **pokiaľ** $N_A \neq \acute{N}_A$

Odstránenie jednoduchých pravidiel

- Na základe množín N_A teraz môžeme odstrániť jednoduché pravidlá.
- Pre každý neterminál vieme, na aké rôzne iné neterminály sa dá prepísať.
- Ak chceme odstrániť jednoduché pravidlá, musíme znovu zabezpečiť, aby to nezmenilo výsledný jazyk, ktorý gramatika generuje.
- To docielime tak, že pre všetky $B \in N_A$ a všetky pravidlá $B \to \alpha$, ktoré nie sú jednoduché, pridáme do upravenej množiny pravidiel aj pravidlá $A \to \alpha$.

Odstránenie jednoduchých pravidiel - príklad

Nech je daná gramatika (bola výsledkom úpravy odstránenia ε-pravidiel):

$$\acute{S} \rightarrow S \mid \varepsilon$$
$$S \rightarrow aAbB \mid AC \mid BD \mid abB \mid A \mid C$$
$$A \rightarrow Aa \mid a$$
$$B \rightarrow bBb$$
$$C \rightarrow aCA \mid A \mid aC \mid aA \mid a$$
$$D \rightarrow d$$

Pre jednotlivé neterminály sú množiny N_A (postupne):

$$\acute{S} : \{\acute{S}\}, \{\acute{S}, S\}, \{\acute{S}, S, A, C\} = N_{\acute{S}}$$
$$S : \{S\}, \{S, A, C\} = N_S$$
$$A : \{A\} = N_A$$
$$B : \{B\} = N_B$$
$$C : \{C\} = \{C, A\} = N_C$$
$$D : \{D\} = N_D$$

Odstránenie jednoduchých pravidiel - príklad

Po odstránení jednoduchých pravidiel dostávame pravidlá:

$$\acute{S} \to \varepsilon \mid aAbB \mid AC \mid BD \mid abB \mid Aa \mid a \mid aCA \mid aC \mid aA$$
$$S \to aAbB \mid AC \mid BD \mid abB \mid Aa \mid a \mid aCA \mid aC \mid aA$$
$$A \to Aa \mid a$$
$$B \to bBb$$
$$C \to aCA \mid aC \mid aA \mid a \mid Aa$$
$$D \to d$$

T.j. gramatiku $(\{\acute{S}, S, A, B, C, D\}, \{a, b, d\}, P, \acute{S})$ bez jednoduchých pravidiel, kde pravidlá P sú uvedené vyššie.

Odstránenie nadbytočných symbolov

V gramatike sa môžu vyskytovať symboly (N, T) ktoré sa alebo nikdy počas derivácie nepoužijú, alebo ak sa použijú, nikdy sa nepodarí odvodiť reťazec terminálov. Ich prítomnosť v gramatike potom zbytočne zväčšuje jej veľkosť a navyše môžu viesť k zlým výsledkom rôznych algoritmov, pracujúcich s gramatikami.

Definícia

Nech $G = (N, T, P, S)$ je bezkontextová gramatika. Symbol $X \in N \cup T$ sa nazýva:

- ***nedostupný**, ak v G neexistuje odvodenie $S \Rightarrow^* \alpha X \beta$ pre nejaké $\alpha, \beta \in (N \cup T)^*$,*
- ***nadbytočný**, ak v G neexistuje odvodenie $S \Rightarrow^* xXz \Rightarrow^* xyz$ pre nejaké $x, y, z \in T^*$.*

Odstránenie nadbytočných symbolov

Definícia

Gramatika bez nadbytočných symbolov sa nazýva ***redukovaná*** *gramatika.*

Každý nedostupný symbol je zároveň aj nadbytočný. Vytvorenie redukovanej gramatiky pozostáva z 2 hlavných fáz:

1. Odstránenie neterminálov, z ktorých nie je možné odvodiť reťazec terminálov alebo ε.
2. Odstránenie neterminálov a terminálov, ktoré sa nemôžu vyskytnúť vo vetných formách gramatiky.

Symboly sa odstránia spolu s príslušnými pravidlami, v ktorých vystupujú.

Príklad

Nech gramatika G má pravidlá:

$$S \to AB \mid A$$

$$A \to AA \mid a$$

$$B \to bB$$

$$C \to c$$

Neterminál B je nadbytočný symbol. Neterminál C a terminál c sú nedostupné symboly. Po ich odstránení dostávame ekvivalentnú redukovanú gramatiku:

$$S \to A$$

$$A \to AA \mid a$$

Odstránenie nadbytočných symbolov - množina N_T

V prvej fáze hľadáme neterminály, z ktorých je možné odvodiť reťazec terminálov (prípadne ε). Hľadáme množinu:

$$N_T = \{A \mid A \Rightarrow^* w, w \in T^*\} \tag{3}$$

Odstránenie nadbytočných symbolov - množina N_T

Vstup: bezkontextová gramatika $G = (N, T, P, S)$
Výstup: množina N_T podľa (1)

$N_T \leftarrow \emptyset$
opakuj
    $\acute{N}_T \leftarrow N_T$
    $N_T \leftarrow \acute{N}_T \cup \{A \mid A \rightarrow \alpha \in P \wedge \alpha \in (\acute{N}_T \cup T)^*\}$
pokiaľ $N_T \neq \acute{N}_T$

Odstránenie nadbytočných symbolov - množina N_T

Príklad: Nech $G = (\{\acute{S}, S, A, B, C, D\}, \{a, b, d\}, P, \acute{S})$, kde P obsahuje pravidlá:

$$\acute{S} \rightarrow \varepsilon \mid aAbB \mid AC \mid BD \mid abB \mid Aa \mid a \mid aCA \mid aC \mid aA$$
$$S \rightarrow aAbB \mid AC \mid BD \mid abB \mid Aa \mid a \mid aCA \mid aC \mid aA$$
$$A \rightarrow Aa \mid a$$
$$B \rightarrow bBb$$
$$C \rightarrow aCA \mid aC \mid aA \mid a \mid Aa$$
$$D \rightarrow d$$

Podľa algoritmu by sa premenná N_T postupne menila: $\emptyset, \{A, C, D, S, \acute{S}\}$. Neterminál B je teda taký, že sa z neho nedá odvodiť terminálny reťazec. Jeho výskyt v gramatike je teda **nadbytočný**, pretože jeho odstránením sa **nijako nezmení jazyk** $L(G)$, ktorý gramatika generuje.

Odstránením neterminálu B a pravidiel s jeho výskytom by sme dostali gramatiku $(\{\acute{S}, S, A, C, D\}, \{a, b, d\}, P, \acute{S})$ s pravidlami P:

$$\acute{S} \to \varepsilon \mid AC \mid Aa \mid a \mid aCA \mid aC \mid aA$$

$$S \to AC \mid Aa \mid a \mid aCA \mid aC \mid aA$$

$$A \to Aa \mid a$$

$$C \to aCA \mid aC \mid aA \mid a \mid Aa$$

$$D \to d$$

Odstránenie nadbytočných symbolov - množina V_D

V druhej fáze hľadáme neterminály a terminály, ktoré sa môžu vyskytnúť v nejakej vetnej forme danej gramatiky, t.j. sú to **dostupné symboly gramatiky**. Hľadáme množinu:

$$V_D = \{X \mid S \Rightarrow^* \alpha X \beta, \alpha, \beta \in (N \cup T)^*\}. \tag{4}$$

Odstránenie nadbytočných symbolov - množina V_D

Vstup: bezkontextová gramatika $G = (N, T, P, S)$
Výstup: množina V_D podľa (2)

1: $V_D \leftarrow \{S\}$
2: **opakuj**
3: $\quad \acute{V}_D \leftarrow V_D$
4: $\quad V_D \leftarrow \acute{V}_D \cup \{X \mid A \rightarrow \alpha X \beta \in P \wedge A \in \acute{V}_D\}$
5: **pokiaľ** $V_D \neq \acute{V}_D$

Odstránenie nadbytočných symbolov - množina V_D

Príklad: Pokračujme v redukcii gramatiky, z ktorej sme naposledy odstránili nadbytočné neterminály, t.j. máme gramatiku $(\{\acute{S}, S, A, C, D\}, \{a, b, d\}, P, \acute{S})$ s počiatočným neterminálom $\acute{S}$ a pravidlami:

$$\acute{S} \to \varepsilon \mid AC \mid Aa \mid a \mid aCA \mid aC \mid aA$$
$$S \to AC \mid Aa \mid a \mid aCA \mid aC \mid aA$$
$$A \to Aa \mid a$$
$$C \to aCA \mid aC \mid aA \mid a \mid Aa$$
$$D \to d$$

Podľa algoritmu by sa premenná V_D postupne menila: $\{\acute{S}\}, \{\acute{S}, A, C, a\}$. Nedostupné symboly sú teda neterminály S, D a terminály b, d (terminál b bol stále formálne uvedený medzi terminálmi, hoci sa už nevyskytoval v žiadnom pravidle).

Odstránenie nadbytočných symbolov - výsledný algoritmus

Vstup: bezkontextová gramatika $G = (N, T, P, S)$
Výstup: gramatika bez nadbytočných symbolov

1: Vytvor množinu N_T.
2: **ak** $S \in N_T$ **potom**
3: odstráň z gramatiky G všetky neterminály, ktoré nepatria do N_T;
4: vytvor množinu V_D
5: odstráň z gramatiky G všetky symboly, ktoré nepatria do V_D;
6: **koniec ak**

Poradie krokov je **dôležité**!. Najprv sa musia odstrániť neterminály, ktoré nevedú na terminálne reťazce (N_T) a až potom sa odstránia nedostupné symboly (V_D).

Redukcia gramatiky - výsledok

Príklad: Redukovanú gramatiku teda získame tak, že najprv odstránime nadbytočné neterminály a potom nedostupné symboly gramatiky. V prípade nami upravovanej gramatiky zo slajdu č. 45 má teda výsledná redukovaná gramatika pravidlá:

$$\acute{S} \rightarrow \varepsilon \mid AC \mid Aa \mid a \mid aCA \mid aC \mid aA$$

$$A \rightarrow Aa \mid a$$

$$C \rightarrow aCA \mid aC \mid aA \mid a \mid Aa$$

pričom neterminály $N = \{\acute{S}, A, C\}$,terminály $T = \{a\}$ a $\acute{S}$ je počiatočný neterminál.

Ak by sme gramatiku zo slajdu č. 45 upravovali **v opačnom poradí**, potom:

1. Množina $V_D = \{\acute{S}, S, A, B, C, D, a, b, d\}$, t.j. všetky symboly sú dostupné a nič sa neodstráni.
2. Množina $N_T = \{A, C, D, S, \acute{S}\}$, t.j. odstránime len neterminál B a výsledná gramatika:

$$
\begin{aligned}
&\acute{S} \to \varepsilon \mid AC \mid Aa \mid a \mid aCA \mid aC \mid aA \\
&S \to AC \mid Aa \mid a \mid aCA \mid aC \mid aA \\
&A \to Aa \mid a \\
&C \to aCA \mid aC \mid aA \mid a \mid Aa \\
&D \to d
\end{aligned}
$$

pričom vidíme, že neterminály S, D a terminál d nie sú dostupné a terminál b nevystupuje v žiadnom pravidle.

Použitá literatúra

Dedera, Ľ: Počítačové jazyky a ich spracovanie.

Linz, P.: An Introduction to Formal Languages and Automata.

Molnár, Ľ.: Gramatiky a jazyky.