

# Prednáška 4 - Bezkontextové jazyky, derivačné stromy, redukovaná gramatika

Ing. Viliam Hromada, PhD.

C-510  
Ústav informatiky a matematiky  
FEI STU

`viliam.hromada@stuba.sk`



## Bezkontextové jazyky

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

$$A \rightarrow \alpha,$$

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



## Príklad bezkontextovej gramatiky

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

$$E \rightarrow E + T \mid T$$

$$T \rightarrow T * F \mid F$$

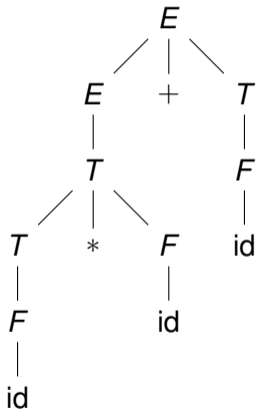
$$F \rightarrow (E) \mid \text{id}.$$

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

$$\begin{aligned} E &\Rightarrow E + T \Rightarrow T + T \Rightarrow T * F + T \Rightarrow F * F + T \Rightarrow \text{id} * F + T \\ &\Rightarrow \text{id} * F + F \Rightarrow \text{id} * F + \text{id} \Rightarrow \text{id} * \text{id} + \text{id} \end{aligned}$$



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



# 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  $N \cup T_\epsilon$ .
- Koreň stromu je ohodnotený začiatočným symbolom  $S$ .
- Ak má nejaký vrchol stromu potomkov, potom je ohodnotený symbolom z množiny  $N$ .
- Ak  $X_1, X_2, \dots, X_k$  sú ohodnotenia priamych potomkov vrcholu  $A$ , potom v  $P$  musí existovať pravidlo  $A \rightarrow X_1 X_2 \dots X_k$ .
- Listy stromu sú ohodnotené symbolmi z množiny  $T_\epsilon$ .
- Slovo  $w \in T^*$  sa dá získať zret'azením ohodnotení listov stromu zľava doprava.





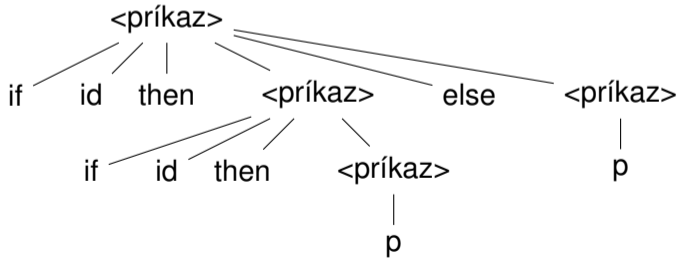
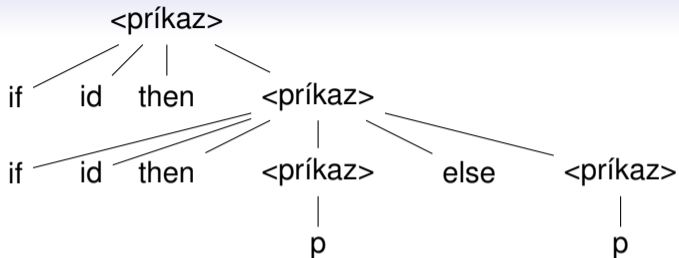
## Nejednoznačná gramatika - príklad

Gramatika  $G = (\{ \langle \text{príkaz} \rangle \}, \{ \text{id}, \text{p}, \text{if}, \text{then}, \text{else} \}, P, \langle \text{príkaz} \rangle)$ , pravidlá  $P$ :

1.  $\langle \text{príkaz} \rangle \rightarrow \text{if id then } \langle \text{príkaz} \rangle$
2.  $\langle \text{príkaz} \rangle \rightarrow \text{if id then } \langle \text{príkaz} \rangle \text{ else } \langle \text{príkaz} \rangle$
3.  $\langle \text{príkaz} \rangle \rightarrow \text{p}$

Zjednodušená gramatika reprezentujúca podmienené príkazy, **id** zastupuje identifikátor (premennú) predstavujúci podmienku, **p** zastupuje nejaký príkaz. Potom pre slovo (reťazec terminálov): **if id then if id then p else p** existujú 2 rôzne stromy odvodzenia:







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

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

$R \rightarrow R|R$  (zjednotenie)

$R \rightarrow R \cdot R$  (zreťazenie)

$R \rightarrow R^*$  (iterácia)

$R \rightarrow a$

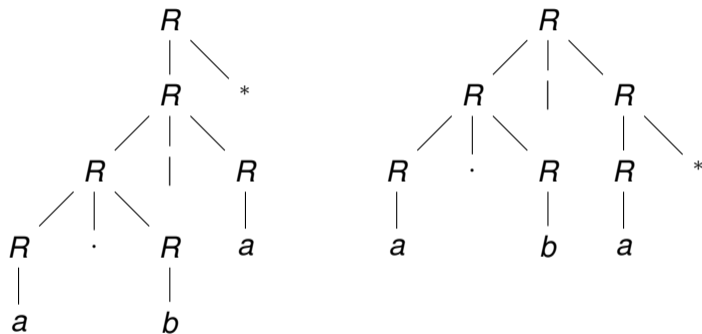
$R \rightarrow b$

$R \rightarrow \varepsilon$



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

V takejto gramatike by mal regulárny výraz  $a \cdot b|a^*$  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  $(ab)|(a^*)$ .



- 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. 4).



## Ľavé (pravé) odvodenie

### Definícia

Nech  $G$  je bezkontextová gramatika. Potom odvodenie  $\alpha_0 \Rightarrow \alpha_1 \Rightarrow \dots \Rightarrow \alpha_k$  v  $G$ , kde každý krok odvodenia  $\alpha_i \Rightarrow \alpha_{i+1}$  pre  $0 \leq i \leq k$  realizujeme tak, že v reťazci  $\alpha_i$  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  $\Rightarrow_l$  ( $\Rightarrow_p$ ).



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

Budeme uvažovať 3 základné úpravy bezkontextových gramatík

1. Odstránenie nadbytočných symbolov - dnes
2. Odstránenie  $\varepsilon$ -pravidiel - dnes
3. Odstránenie jednoduchých pravidiel - budúci týždeň





## Odstránenie nadbytočných symbolov

Môže sa stať, že pri definícii gramatiky sa použijú symboly (či už  $N$  alebo  $T$ ), ktoré sa alebo nikdy počas derivácii 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  $\epsilon$ .
2. Odstránenie neterminálov a terminálov, ktoré sa nemôžu vyskytnúť vo vetných formách gramatiky.

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



## Príklad

Nech gramatika  $G$  má pravidlá:

$$S \rightarrow AB \mid A$$

$$A \rightarrow AA \mid a$$

$$B \rightarrow bB$$

$$C \rightarrow c$$

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

$$S \rightarrow A$$

$$A \rightarrow AA \mid a$$



## Odstránenie nadbytočných symbolov - množina $N_T$

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

$$N_T = \{A \mid A \Rightarrow^* w, w \in T^*\} \quad (1)$$



## Odstránenie nadbytočných symbolov - množina $N_T$

**Vstup:** bezkontextová gramatika  $G = (N, T, P, S)$

**Výstup:** množina  $N_T$  podľa (1)

1:  $N_T \leftarrow \emptyset$

2: **opakuj**

3:  $N'_T \leftarrow N_T$

4:  $N_T \leftarrow N'_T \cup \{A \mid A \rightarrow \alpha \in P \wedge \alpha \in (N'_T \cup T)^*\}$

5: **pokiaľ**  $N_T \neq N'_T$



## Odstránenie nadbytočných symbolov - množina $N_T$

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

$$S \rightarrow ABc \mid abA$$

$$A \rightarrow aAa \mid bBb \mid b$$

$$B \rightarrow aBb \mid caBC$$

$$C \rightarrow \varepsilon \mid ab \mid BcCa$$

Podľa algoritmu by sa premenná  $N_T$  postupne menila:

$\emptyset, \{A, C\}, \{S, A, C\}$ . Neterminál  $B$  je teda taký, že sa z neho nedá odvodiť terminálny reťazec.



## 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:

$$V_D = \{X \mid S \Rightarrow^* \alpha X \beta, \alpha, \beta \in (N \cup T)^*\}. \quad (2)$$



## 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:  $V'_D \leftarrow V_D$

4:  $V_D \leftarrow V'_D \cup \{X \mid A \rightarrow \alpha X \beta \in P \wedge A \in V'_D\}$

5: **pokiaľ**  $V_D \neq V'_D$





## Odstránenie nadbytočných symbolov - množina $V_D$

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

$$S \rightarrow AB \mid abS \mid \varepsilon$$

$$A \rightarrow aAa \mid b$$

$$B \rightarrow aCb \mid cC$$

$$C \rightarrow cS$$

$$D \rightarrow abC \mid \varepsilon \mid dD$$

Podľa algoritmu by sa premenná  $V_D$  postupne menila:

$\{S\}, \{S, A, B, a, b\}, \{S, A, B, C, a, b, c\}$ . Nedostupné symboly sú teda  $D, d$ .



## 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 ( $V_D$ ).



## Odstránenie nadbytočných symbolov

**Príklad:** Pokračujme v gramatike, pre ktorú sme hľadali množinu  $N_T$  na slajde č. 22. Po odstránení neterminálu  $B$  (pretože nebol v množine  $N_T$ ) dostávame gramatiku  $G_1 = (\{S, A, C\}, \{a, b, c\}, P_1, S)$ , kde  $P_1$  obsahuje pravidlá:

$$S \rightarrow abA$$

$$A \rightarrow aAa \mid b$$

$$C \rightarrow ab \mid \varepsilon$$

Pre túto gramatiku je množina  $V_D = \{S, A, a, b\}$ , t.j. symboly  $C, c$  sú nedostupné. Po ich odstránení dostávame redukovanú gramatiku:

$$G_{red} = (\{S, A\}, \{a, b\}, \{S \rightarrow abA, A \rightarrow aAa \mid b\}, S).$$

T.j.  $L(G) = L(G_1) = L(G_{red}) = \{aba^nba^n \mid n \in \{0, 1, \dots\}\}$ .



Ak by sme gramatiku zo slajdu 22 upravovali **v opačnom poradí**, potom:

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

$$S \rightarrow abA$$

$$A \rightarrow aAa \mid b$$

$$C \rightarrow \varepsilon \mid ab$$

pričom vidíme, že neterminál  $C$  nie je dostupný a terminál  $c$  ani len nevystupuje v niektorom z pravidiel.



## Odstránenie $\varepsilon$ -pravidiel

### Definícia

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



## Odstránenie $\varepsilon$ -pravidiel

- Každé  $\varepsilon$ -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  $\varepsilon$  a teda sa mohli "*stratiť*" z vetnej formy.
- Jediný prípad, kedy akceptujeme  $\varepsilon$ -pravidlo je v prípade, že  $\varepsilon \in L(G)$ . V takom prípade doplníme nový počiatočný symbol  $S'$ , doplníme pravidlá  $S' \rightarrow S \mid \varepsilon$  a ostatné  $\varepsilon$ -pravidlá príslušným spôsobom odstránime.



## Príklad

Gramatiku s  $\varepsilon$ -pravidlami

$$S \rightarrow aSb \mid \varepsilon \mid A$$

$$A \rightarrow bAa \mid \varepsilon$$

Upravíme na:

$$S' \rightarrow S \mid \varepsilon$$

$$S \rightarrow aSb \mid A \mid ab$$

$$A \rightarrow bAa \mid ba$$



## Odstránenie $\varepsilon$ -pravidiel

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

$$N_\varepsilon = \{A \mid A \Rightarrow^* \varepsilon, A \in N\} \quad (3)$$

Množinu iteratívne tvoria tie neterminály, z ktorých možno odvodiť len  $\varepsilon$ .





## Odstránenie $\varepsilon$ -pravidiel

**Vstup:** bezkontextová gramatika  $G = (N, T, P, S)$

**Výstup:** množina  $N_\varepsilon$  (3)

1:  $N_\varepsilon \leftarrow \emptyset$ .

2: **opakuj**

3:  $\dot{N}_\varepsilon \leftarrow N_\varepsilon$

4:  $N_\varepsilon \leftarrow \dot{N}_\varepsilon \cup \{A \mid A \rightarrow \alpha \in P \wedge \alpha \in \dot{N}_\varepsilon^*\}$

5: **pokiaľ**  $N_\varepsilon \neq \dot{N}_\varepsilon$



## Odstránenie $\varepsilon$ -pravidiel

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

$$A \rightarrow \alpha_0 B_1 \alpha_1 \dots B_k \alpha_k,$$

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

$$A \rightarrow \alpha_0 X_1 \alpha_1 \dots X_k \alpha_k,$$

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



## Odstránenie $\varepsilon$ -pravidiel

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



## Odstránenie $\varepsilon$ -pravidliel - príklad

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

$$S \rightarrow aAbB \mid AC$$

$$A \rightarrow Aa \mid \varepsilon$$

$$B \rightarrow bBb \mid ab$$

$$C \rightarrow aCA \mid \varepsilon \mid A$$

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



## Odstránenie $\varepsilon$ -pravidliel - príklad (pokr.)

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

$$\hat{S} \rightarrow S \mid \varepsilon$$

$$S \rightarrow aAbB \mid AC \mid abB \mid A \mid C$$

$$A \rightarrow Aa \mid a$$

$$B \rightarrow bBb \mid ab$$

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

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



## Použitá literatúra

Dedera, L': Počítačové jazyky a ich spracovanie.

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

Molnár, L': Gramatiky a jazyky.