AUTOMATY A GRAMATIKY

7

Pavel Surynek

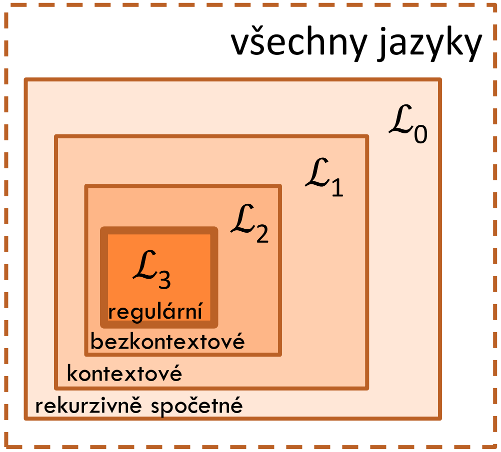
Univerzita Karlova v Praze

Matematicko-fyzikální fakulta
Katedra teoretické informatiky a matematické logiky

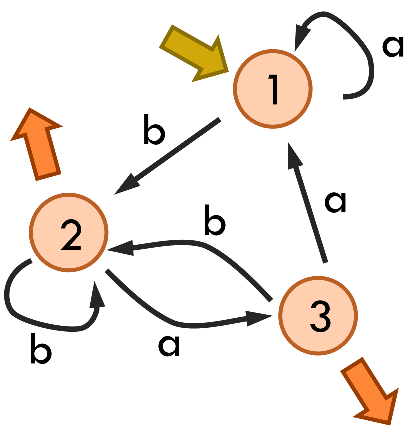
Gramatiky typu 3
Bezkontextové gramatiky
Backus-Naurova forma
Redukce
Derivace - pravé, levé
Derivační stromy

$\mathcal{L}_3$

Konečný **automat** ⇒ gramatika typu 3



- KA $A=(Q, X, \delta, q_0, F)$
  - definujeme gramatiku $G = (Q, X, q_0, P)$, kde
    - $p \rightarrow xq \in P$, kdykoli $\delta(p, x) = q$
    - $p \rightarrow \lambda \in P$, kdykoli $p \in F$
- $w \in L(A) \Leftrightarrow w \in L(G)$
  - $x_1x_2\ldots x_n \in L(A) \Leftrightarrow (\exists q_0q_1q_2\ldots q_n)\ \delta(q_{i-1},x_i)=q_i$ a $q_n \in F$, právě když
  - $q_0 \Rightarrow_G x_1q_1 \Rightarrow_G x_1x_2q_2 \Rightarrow_G \ldots \Rightarrow_G x_1x_2\ldots x_nq_n \Rightarrow_G x_1x_2\ldots x_n$



Př.: $G = (V_N, V_T, 1, P)$, kde
$V_N = \{ 1, 2, 3 \}$
$V_T = \{ a, b \}$
$P = \{ 1 \rightarrow a1 \mid b2$
$2 \rightarrow a3 \mid b2 \mid \lambda$
$3 \rightarrow a1 \mid b2 \mid \lambda \}$

$\mathcal{L}_3$

Gramatika **typu 3** ⇒ konečný automat (1)

- gramatika $G = (V_N, V_T, S, P)$ typu 3
  - zkonstruujeme gramatiku $G' = (V_N', V_T, S, P')$ typu 3, že $L(G') = L(G)$, kde
    - pravidla mají v P' mají tvar $X \to xY$ nebo $X \to \lambda$, kde $X, Y \in V_N$ a $x \in V_T$
      - pro pravidlo $\mathbf{X \to x_1 x_2 \ldots x_n Y} \in P$ s $x_i \in V_T$ pro $i=1,2,\ldots,n$ a $X,Y \in V_N$ dáme do P' pravidla
        - $X \to x_1 Y_1$, $Y_1 \to x_2 Y_2$, $Y_2 \to x_3 Y_3$, ..., $Y_{n-1} \to x_n Y$, kde $Y_1, Y_2, \ldots, Y_{n-1}$ jsou nové neterminály do V_N'
      - podobně pro pravidlo $\mathbf{X \to x_1 x_2 \ldots x_n} \in P$ s $x_i \in V_T$ pro $i=1,2,\ldots,n$ a $X,Y \in V_N$ dáme do P' pravidla
        - $X \to x_1 Z_1$, $Z_1 \to x_2 Z_2$, $Z_2 \to x_3 Z_3$, ..., $Z_{n-1} \to x_n Z_n$, $Z_n \to \lambda$, kde $Z_1, Z_2, \ldots, Z_n$ jsou nové neterminály do V_N'
    - ošetření pravidel $\mathbf{X \to Y} \in P$ s $X, Y \in V_N$
      - zkonstruujeme $\Phi(X) = \{Y \mid Y \in V_N \wedge X \Rightarrow_G^* Y\}$
        - postupná konstrukce $\Phi_1(X) = \{Y \mid Y \in V_N \wedge X \Rightarrow_G Y\}$
        - $\Phi_{i+1}(X) = \Phi_i(X) \cup \{Y \mid Y \in V_N \wedge (\exists Z)[Z \in \Phi_i(X) \wedge Z \Rightarrow_G Y]\}$
      - do P' přidáme pravidla $X \to w$, kdykoli $Y \to w \in P'$ pro $Y \in \Phi(X)$

$\mathcal{L}_3$

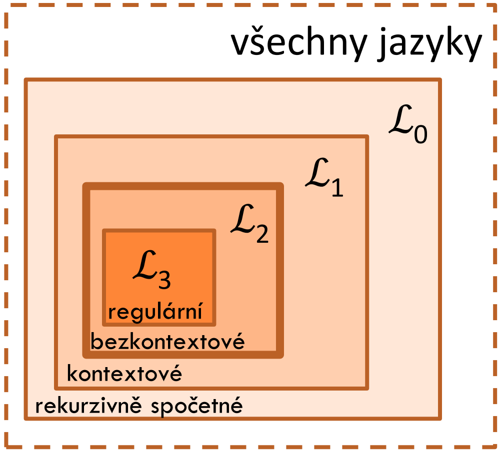
Gramatika **typu 3** ⇒ konečný automat (2)

- gramatika $G' = (V_N', V_T, S, P')$ typu 3, kde
  - pravidla mají v P' mají tvar $X \rightarrow xY$ nebo $X \rightarrow \lambda$, kde $X, Y \in V_N'$ a $x \in V_T$
    - definujeme NKA $A = (V_N', V_T, \delta, \{S\}, F)$, kde
      - $F = \{ X \mid X \in V_N' \wedge X \rightarrow \lambda \in P' \}$
      - $\delta(X, x) = \{ Y \mid Y \in V_N' \wedge X \rightarrow xY \in P' \}$ pro $X \in V_N'$ a $x \in V_T$
- $w \in L(G') \Leftrightarrow w \in L(A)$
  - $x_1 x_2 \ldots x_n \in L(G') \Leftrightarrow$ existuje odvození $S \Rightarrow_{G'} x_1 Y_1 \Rightarrow_{G'} x_1 x_2 Y_2 \Rightarrow_{G'} \ldots \Rightarrow_{G'} x_1 x_2 \ldots x_n Y_n \Rightarrow_{G'} x_1 x_2 \ldots x_n$, kde $Y_1, Y_2, \ldots, Y_n \in V_N'$, právě když
  - $(\exists Y_1, Y_2, \ldots, Y_n \in V_N')\ Y_{i+1} \in \delta(Y_i, x_{i+1})$ pro $i = 1, 2, \ldots, n-1$, $Y_1 \in \delta(S, x_1)$ a $Y_n \in F$

Pavel Surynek, 2015

$\mathcal{L}_2$

Bezkontextové gramatiky



- **syntaxe** jazyků
  - programovacích (Algol, Pascal, 60. léta)
  - značkovacích (xml, html, ...)
- Backus-Naurova forma
  - forma zápisu bezkontextových gramatik
    - pravidla tvaru: *<identifikátor> ::=* $výraz_1 | výraz_2 | \ldots | výraz_n$
      - *<identifikátor>* odpovídá neterminálu
        - $výraz_i$ odpovídá pravé straně pravidla
        - pomocí [ ] lze ve výrazu vyznačit volitelnou část
        - pomocí { } lze vyznačit skupinu, pomocí ... opakování skupiny

Pozn.: moderní jazyky C++, Java, C# mají složitější syntaxi

Př.: <number> ::= <digit> | <number> <digit>
<digit> ::= **0** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9**

Př.: <statement> ::= **if** <condition> **then** <statement> [**else** <statement>]

Př.: <program> ::= <statement> [{; <statement>} ...]
<statement> ::= <if-statement> | <while-statement> | <assignment>
<if-statement> ::= **if** <condition> **then** <statement> [**else** <statement>]
<while-statement> ::= **while** <condition> **do** <statement>
<assignment> ::= <variable> **:=** <expression>

Redukce bezkontextových gramatik (1)

- bezkontextová gramatika $G = (V_N, V_T, S, P)$, kde $L(G) \neq \emptyset$, je **redukovaná**, jestliže
  - (i) pro každý neterminál $X \in V_N$ existuje slovo $w \in V_T^*$, že $X \Rightarrow_G^* w$ a
    - ukončitelné
  - (ii) pro každý neterminál $X \in V_N - \{S\}$ existují slova $u, v \in (V_T \cup V_N)^*$, že $S \Rightarrow_G^* uXv$
    - dosažitelné
- pro bezkontextovou gramatiku G, kde $L(G) \neq \emptyset$, existuje redukovaná bezkontextová gramatika G'', že $L(G'') = L(G)$
  - 1. odstranit z G neterminály nesplňující (i)
  - 2. odstranit z G neterminály nesplňující (ii)
    - odstranění neterminálu = odstranění všech pravidel, které jej obsahují

Př.: $G = (V_N, V_T, S, P)$, kde
$V_N = \{ S, A, B, C, D \}$
$V_T = \{ a, b \}$
$P = \{ S \to aA \mid ab$
$A \to BC$
$B \to ba$
$D \to ab \mid \lambda \}$

G není redukovaná

C nesplňuje (i)
D nesplňuje (ii)

Redukce bezkontextových gramatik (2)

- 1. odstranění neterminálů **nesplňujících (i)**
  - tedy **neukončitelných**
  - hledáme $T = \{ X \in V_N \mid (\exists w \in V_T^*)\ X \Rightarrow_G^* w \}$
    - $T_0 = V_T$
    - $T_{i+1} = T_i \cup \{ X \in V_N \mid (\exists w \in T_i^*)\ X \Rightarrow_G w \}$
    - $T_0 \subseteq T_1 \subseteq \ldots \subseteq (V_T \cup V_N)$, $(\exists k)\ T_{k+1} = T_k$
      - platí $T = T_k \cap V_N$
      - mimochodem $L(G) \neq \emptyset \Leftrightarrow S \in T$
  - odstraníme z G pravidla obsahující neterminál z V_N-T (v pravidle vlevo či vpravo)
    - výsledná gramatika splňuje (i)
      - pro $X \in T$ existuje $w \in V_T^*$, že $X \Rightarrow_{G'}^* w$
        - derivace pro $X \Rightarrow_G^* w$, používá neodstraněná pravidla (indukcí)
      - $w \in L(G') \Rightarrow w \in L(G)$
        - G’ má méně pravidel
      - $w \in L(G) \Rightarrow w \in L(G')$
        - použití odstraněného pravidla by zabránilo ukončení derivace

Př.: $G = (V_N, V_T, S, P)$, kde
$V_N = \{ S, A, B, C, D \}$
$V_T = \{ a, b \}$
$P = \{ S \to aA \mid ab$
$A \to BC$
$B \to ba$
$D \to ab \mid \lambda \}$

$T_0 = \{ a, b \}$
$T_1 = \{ a, b, S, B, D \}$
$T_2 = \{ a, b, S, B, D \}$
$T = \{ S, B, D \}$

$G' = (T, V_T, S, P')$, kde
$V_T = \{ a, b \}$
$P' = \{ S \to ab$
$B \to ba$
$D \to ab \mid \lambda \}$

Pavel Surynek, 2015

Redukce bezkontextových gramatik (3)

- 2. odstranění neterminálů **nesplňujících (ii)**
  - tedy **nedosažitelných**
  - hledáme $R = \{ X \in V_N \mid (\exists u,v \in (V_T \cup V_N)^*)\ S \Rightarrow_{G'}^* uXv \}$
    - $R_0=\{S\}$
    - $R_{i+1}=R_i \cup \{ X \in V_N \mid (\exists Y \in R_i, \exists u,v \in (V_T \cup V_N)^*)\ Y \Rightarrow_{G'} uXv \}$
    - $R_0 \subseteq R_1 \subseteq \ldots \subseteq V_N$, $(\exists k)\ R_{k+1}=R_k$
      - platí $R = R_k$
    - odstraníme z G’ pravidla obsahující neterminál z T -R (v pravidle vlevo či vpravo)
    - výsledná gramatika G’’ splňuje (i) a (ii)
      - pro $X \in T$ existují $\exists u,v \in (V_T \cup V_N)^*$, že $S \Rightarrow_{G''}^* uXv$
        - derivace $S \Rightarrow_{G'}^* uXv$ používá neodstraněná pravidla (indukcí)
      - (i) zůstává splněno
        - pokud všechny derivace ukazující, že $X \Rightarrow_{G'}^* w$ pro nějaké $w \in V_T^*$ vytváří neterminál $Y \notin R$ (tj. X by se stal nově neukončitelným), pak $X \notin R$ (tj. X je nedosažitelný v G’ a bude odstraněn nyní)
      - $w \in L(G'') \Rightarrow w \in L(G')$
        - G’’ má méně pravidel
      - $w \in L(G') \Rightarrow w \in L(G'')$
        - v derivaci $S \Rightarrow_{G'}^* w$, kde $w \in V_T^*$ jsou všechny neterminály dosažitelné

Př.: $G' = (T, V_T, S, P')$, kde

$V_T = \{ a, b \}$

$P' = \{ S \rightarrow ab$

$B \rightarrow ba$

$D \rightarrow ab \mid \lambda \}$

$R_0 = \{ S \}$

$R_1 = \{ S \}$

$R = \{ S \}$

$G'' = (R, V_T, S, P'')$, kde

$V_T = \{ a, b \}$

$P'' = \{ S \rightarrow ab \}$

Bezkontextové derivace

Př.: G = (V_N, V_T, S, P), kde
V_N = { S }
V_T = { (,) }
P= { S → SS
S → (S)
S → () }

- derivace (**obecná**)
  - $\underline{S} \Rightarrow_G S\underline{S} \Rightarrow_G \underline{S}(S) \Rightarrow_G ()(\underline{S}) \Rightarrow_G ()((\underline{S})) \Rightarrow_G ()((()))$
- **levá** derivace (*left most*)
  - $\underline{S} \Rightarrow_G^{lm} \underline{S}S \Rightarrow_G^{lm} ()\underline{S} \Rightarrow_G^{lm} ()(\underline{S}) \Rightarrow_G^{lm} ()((\underline{S})) \Rightarrow_G^{lm} ()((()))$
    - přepisujeme vždy nejlevější literál
- **pravá** derivace (*right most*)
  - $\underline{S} \Rightarrow_G^{rm} S\underline{S} \Rightarrow_G^{rm} S(\underline{S}) \Rightarrow_G^{rm} S((\underline{S})) \Rightarrow_G^{rm} \underline{S}((())) \Rightarrow_G^{rm} ()((()))$
    - přepisujeme vždy nejpravější literál
- **poznatky**
  - všechny derivace mají stejnou délku
  - používají se stejná pravidla (na stejných místech v odvozovaném slově)
    - liší se pouze pořadí aplikace pravidel

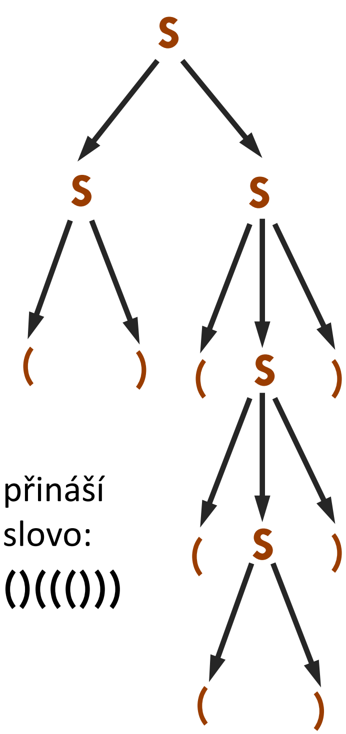
Levá/pravá derivace

- mějme bezkontextovou gramatiku $G=(V_N, V_T, S, P)$
  - jestliže pro $w \in V_T^*$ je $S \Rightarrow_G^* w$, pak $S \Rightarrow_G^{lm*} w$ a $S \Rightarrow_G^{rm*} w$
    - v bezkontextových gramatikách se lze omezit na levé resp. pravé derivace
    - indukcí podle délky pod-derivace, kterou je třeba upravit
      - $S \Rightarrow_G \dots \Rightarrow_G uXv_1 \Rightarrow_G uXv_2 \Rightarrow_G \dots \Rightarrow_G uXv_n \Rightarrow_G uzv_n \Rightarrow_G^* w$ pro $u \in V_T^*$ a $v_i \in (V_T \cup V_N)^*$ pro $i = 1,2,\dots,n$ a $z \in (V_T \cup V_N)^*$, kde $z \neq X$
        - $uXv_1 \Rightarrow_G uXv_2$ je první okamžik, kdy nebyl přepsán **nejlevější** neterminál
      - část derivace $uXv_1 \Rightarrow_G uXv_2 \Rightarrow_G \dots \Rightarrow_G uXv_n \Rightarrow_G uzv_n$ nahradíme $uXv_1 \Rightarrow_G uzv_1 \Rightarrow_G uzv_2 \Rightarrow_G \dots \Rightarrow_G uzv_n$
        - upravovaná pod-derivace je nyní $uzv_1 \Rightarrow_G uzv_2 \Rightarrow_G \dots \Rightarrow_G uzv_n \Rightarrow_G uzv_n \Rightarrow_G^* w$
    - pravá derivace analogicky

Derivační **strom**

Př.: $G = (V_N, V_T, S, P)$, kde
$V_N = \{ S \}$
$V_T = \{ (,) \}$
$P = \{ S \rightarrow SS$
$S \rightarrow (S)$
$S \rightarrow () \}$

- $G=(V_N, V_T, S, P)$ bezkontextová gramatika
  - určité přeuspořádání použití pravidel v derivaci nemá vliv na generované slovo
    - **derivační strom** (*parse tree*) je orientovaný pěstovaný strom (hledíme na pořadí následníků), kde:
      - kořen je ohodnocen S
      - každý vrchol je ohodnocen symbolem z $V_N \cup V_T \{\lambda\}$
      - jestliže je vrchol ohodnocen neterminálem $X \in V_N$, pak má následníky ohodnocené $x_1,x_2,\ldots,x_n$, kde
        - $X \rightarrow x_1x_2\ldots x_n$ s $x_i \in (V_T \cup V_N)$ pro $i=1,2,\ldots,n$ a $X \in V_N$ je pravidlo gramatiky
      - jestliže je vrchol ohodnocen terminálem nebo λ, je to list
  - derivační strom **přináší** slovo $w \in V_T^*$, jestliže w dostaneme konkatenací symbolů v listech postupně zleva doprava
    - v pěstovaném stromu je pořadí listů definováno

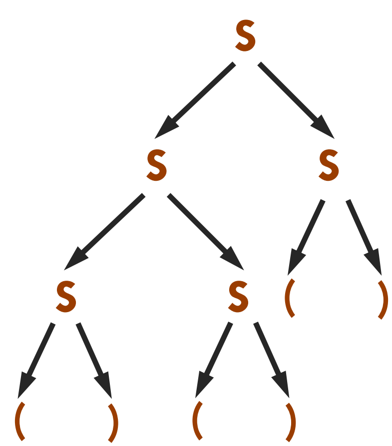


přináší slovo:
()((()))

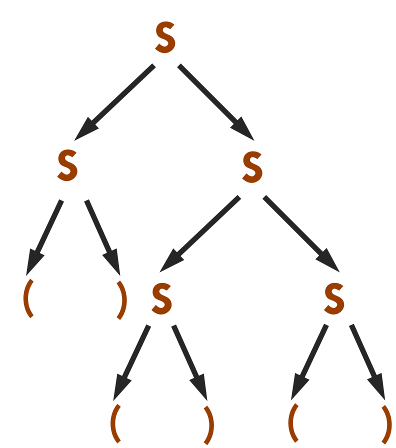
Vlastnosti derivačních stromů

- $G=(V_N, V_T, S, P)$ bezkontextová gramatika
  - $S \Rightarrow_G^* w$ pro $w \in V_T^*$, právě když existuje derivační strom pro G, který přináší w
    - z derivace dostaneme strom jednoznačně
    - strom neurčuje derivaci jednoznačně
      - strom určuje mnohé derivace
    - levá resp. pravá derivace je stromem určena jednoznačně
      - různé levé (pravé derivace) vedou na různé stromy
  - nejednoznačnost derivačních stromů

Př.: $G = (V_N, V_T, S, P)$, kde
$V_N = \{ S \}$
$V_T = \{ (,) \}$
$P = \{ S \rightarrow SS$
$S \rightarrow (S)$
$S \rightarrow () \}$



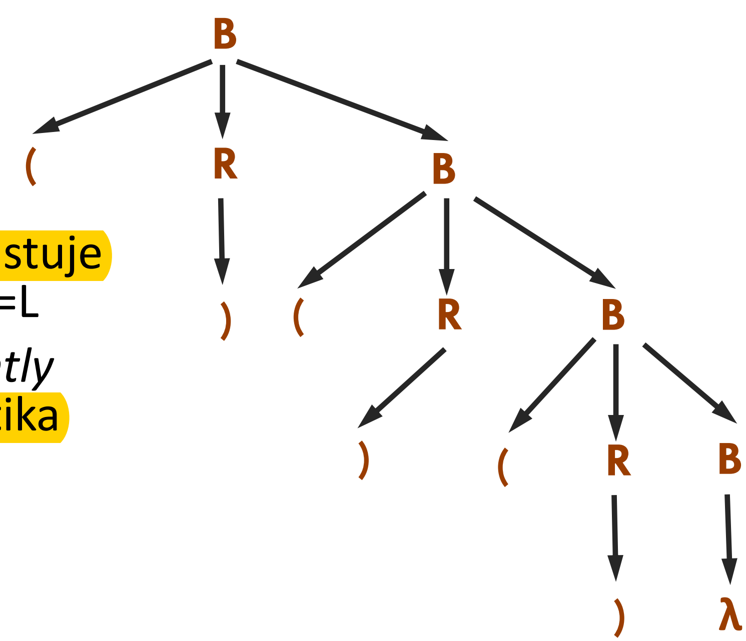
různé derivační stromy přinášející stejné slovo
()()()



Jednoznačnost gramatiky

- $G=(V_N, V_T, S, P)$ bezkontextová gramatika je **nejednoznačná** (*ambiguous*), jestliže
  - existuje slovo $w \in L(G)$, které má dvě různé levé derivace
  - alternativně: existuje slovo $w \in L(G)$, které má dvě různé pravé derivace
  - jinak říkáme, že gramatika je **jednoznačná** (unambiguous)
- tato nejednoznačnost je vlastnost gramatiky
  - bezkontextový jazyk L je **jednoznačný**, jestliže existuje jednoznačná bezkontextová gramatika G, že $L(G)=L$
  - bezkontextový jazyk L je **nejednoznačný** (*inherently ambiguous*), jestliže každá bezkontextová gramatika G, že $L(G)=L$, je nejednoznačná

Př.: $G = (V_N, V_T, B, P)$, kde
$V_N = \{ R, B \}$
$V_T = \{ (,) \}$
$P= \{ B \rightarrow (RB \mid \lambda$
$R \rightarrow) \mid (RR \}$



Př.: $L = \{ 0^i 1^j 2^k \mid i,j,k \in \mathbb{N}_0 \wedge (i=j \vee j=k) \}$
je nejednoznačný

Využití jednoznačnosti - LL(1)

- jednoznačnost gramatiky umožňuje snadné rozpoznávání přijímaných slov
  - pro dané slovo konstruujeme levou derivaci
    - použité pravidlo je **jednoznačně** určeno následujícími symboly
      - speciálně LL(1) gramatika
        - **L**L(1) - levá derivace
        - L**L**(1) - čtení vstupu zleva
        - LL(**1**) - výhled na 1 symbol (1 následující symbol stačí na určení pravidla)
        - programovací jazyky mají většinou LL(1) gramatiku

Př.: $G = (V_N, V_T, B, P)$, kde
$V_N = \{ R, B \}$
$V_T = \{ (,) \}$
$P = \{$ **B → (RB | λ**
R →) | (RR $\}$

zbývající vstup	**levá derivace**
(())()	B
**(**))()	(RB
))()	((RRB
)()	(()RB
**(**)	(())B
)	(())(RB
λ	(())()B
	(())()