Kapitola VI. Modely pro bezkontextové jazyky

Bezkontextová gramatika (BKG)

Myšlenka: *Gramatika* je založena na konečné množině gramatických pravidel, které generují řetězce daného jazyka.

Ilustrace:

Počáteční neterminál

Gramatika *G*:

Neterminály: $A, B, \boxed{S}$

Terminály: a, b, c, d

Pravidla:
$S \rightarrow AB,$
$A \rightarrow aAb,$
$A \rightarrow ab,$
$B \rightarrow bBa,$
$B \rightarrow ba$

S

Pravidlo: $S \rightarrow AB$

$\overbrace{AB}$

Pravidlo: $A \rightarrow ab$

$\overbrace{ab}B$

Pravidlo: $B \rightarrow bBa$

$ab\overbrace{bBa}$

Pravidlo: $B \rightarrow ba$

$abb\overbrace{ba}a \in L(G)$

Bezkontextová gramatika: Definice

Definice: Bezkontextová gramatika (BKG) je čtveřice $G = (N, T, P, S)$, kde

- N je abeceda *neterminálů*
- T je abeceda *terminálů*, přičemž $N \cap T = \emptyset$
- P je konečná množina *pravidel* tvaru $A \rightarrow x$, kde $A \in N$, $x \in (N \cup T)^*$
- $S \in N$ je *počáteční neterminál*

Matematická poznámka k pravidlům:

- Čistě matematicky, P je relace z N do $(N \cup T)^*$
- Místo relačního zápisu $(\boldsymbol{A}, \boldsymbol{x}) \in P$ zapisujeme pravidla $\boldsymbol{A} \rightarrow \boldsymbol{x} \in P$

- $\boldsymbol{A} \rightarrow \boldsymbol{x}$ znamená, že $\boldsymbol{A}$ má být přepsáno na $\boldsymbol{x}$
- $\boldsymbol{A} \rightarrow \varepsilon$ je nazýváno ***ε-pravidlo***

Konvence

- $A, \ldots, F, S$: neterminály
- S : počáteční neterminál
- $a, \ldots, d$: terminály
- $U, \ldots, Z$: prvky množiny $(N \cup T)$
- $u, \ldots, z$: prvky množiny $(N \cup T)^*$
- π : sekvence pravidel

Každá podmnožina pravidel tvaru:

$$A \rightarrow x_1, A \rightarrow x_2, \ldots, A \rightarrow x_n$$

může být zjednodušeně zapsána jako:

$$A \rightarrow x_1 \mid x_2 \mid \ldots \mid x_n$$

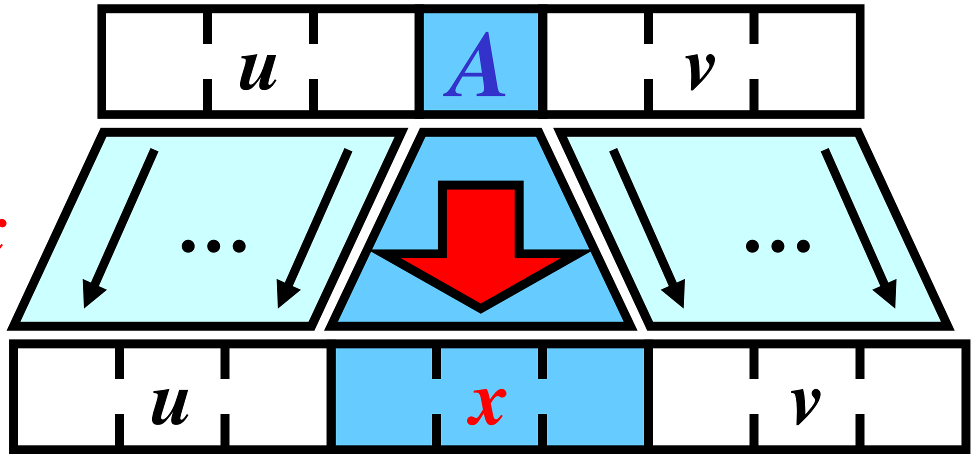
Derivační krok u BKG

Myšlenka: Změnění řetězce použitím pravidla

Definice: Nechť $G = (N, T, P, S)$ je BKG. Nechť $\boldsymbol{u}, \boldsymbol{v} \in (N \cup T)^*$ a $p = A \rightarrow x \in P$. Potom, $\boldsymbol{uAv}$ *přímo derivuje* $\boldsymbol{uxv}$ za použití p v G, zapsáno $\boldsymbol{uAv} \Rightarrow \boldsymbol{uxv}$ [p] nebo zjednodušeně $\boldsymbol{uAv} \Rightarrow \boldsymbol{uxv}$.

Pozn.: Pokud $\boldsymbol{uAv} \Rightarrow \boldsymbol{uxv}$ v G, můžeme říct, že G provádí derivační krok z $\boldsymbol{uAv}$ do $\boldsymbol{uxv}$.

Pravidlo: $A \rightarrow x$



Sekvence derivačních kroků 1/2

Myšlenka: Několik derivačních kroků po sobě

Definice: Nechť $u \in (N \cup T)^*$. G provede *nula derivačních kroků* z u do u; zapisujeme:

$$u \Rightarrow^0 u\ [\varepsilon] \text{ nebo zjednodušeně } u \Rightarrow^0 u$$

Definice: Nechť $u_0,\ldots,u_n \in (N \cup T)^*$, $n \geq 1$ a $u_{i-1} \Rightarrow u_i\ [p_i]$, $p_i \in P$ pro všechna $i = 1,\ldots, n$, což znamená:

$$u_0 \Rightarrow u_1\ [p_1] \Rightarrow u_2\ [p_2]\ \ldots \Rightarrow u_n\ [p_n]$$

Pak, G provede *n derivačních kroků* z u_0 do u_n; zapisujeme:

$$u_0 \Rightarrow^n u_n\ [p_1 \ldots p_n] \text{ nebo zjednodušeně } u_0 \Rightarrow^n u_n$$

Sekvence derivačních kroků 2/2

Pokud $u_0 \Rightarrow^n u_n$ [π] pro nějaké $n \geq 1$, pak u_0 *derivuje* u_n v G, zapisujeme: $u_0 \Rightarrow^+ u_n$ [π].

Pokud $u_0 \Rightarrow^n u_n$ [π] pro nějaké $n \geq 0$, pak u_0 *derivuje* u_n v G, zapisujeme: $u_0 \Rightarrow^* u_n$ [π].

Příklad: Uvažujme

$aAb \Rightarrow aaBbb$ [**1**: $A \rightarrow aBb$] a

$aaBbb \Rightarrow aacbb$ [**2**: $B \rightarrow c$].

Potom: $aAb \Rightarrow^2 aacbb$ [**1 2**],

$aAb \Rightarrow^+ aacbb$ [**1 2**],

$aAb \Rightarrow^* aacbb$ [**1 2**]

Generovaný jazyk

Myšlenka: *G generuje* řetězec terminálů *w* pomocí sekvence derivačních kroků z *S* do *w*

Definice: Nechť $G = (N, T, P, S)$ je BKG.
Jazyk generovaný BKG G, $L(G)$, je definován:

$$L(G) = \{w : w \in T^*, S \Rightarrow^* w\}$$

Ilustrace:

$G = (N, T, P, S)$, nechť $w = a_1 a_2 \ldots a_n$; $a_i \in T$ pro $i = 1..n$

pokud $S \Rightarrow \ldots \Rightarrow \ldots \Rightarrow \underbrace{a_1 a_2 \ldots a_n}_{w}$, **pak** $w \in L(G)$;

jinak $w \notin L(G)$

Bezkontextový jazyk (BKJ)

Myšlenka: Jazyk generovaný bezkontextovou gramatikou

Definice: Nechť L je jazyk. L je bezkontextový jazyk (BKJ), pokud existuje bezkontextová gramatika, která generuje tento jazyk L.

Příklad:

$G = (N, T, P, S)$, kde $N = \{S\}$, $T = \{a, b\}$,

$P = \{1: S \rightarrow aSb,\ 2: S \rightarrow \varepsilon\}$

$S \Rightarrow \varepsilon \quad [2]$

$S \Rightarrow aSb\ [1] \Rightarrow ab \quad [2]$

$S \Rightarrow aSb\ [1] \Rightarrow aaSbb\ [1] \Rightarrow aabb\ [2]$

$\vdots$

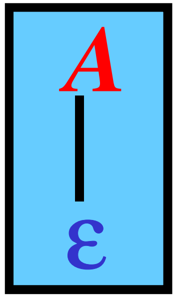
$L(G) = \{a^n b^n : n \geq 0\}$

$L = \{a^n b^n : n \geq 0\}$ je bezkontextový jazyk.

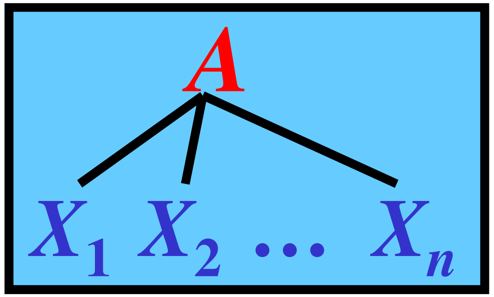
Pravidlový strom

- **Pravidlový strom graficky znázorňuje pravidlo**

1) $A \to \varepsilon$:

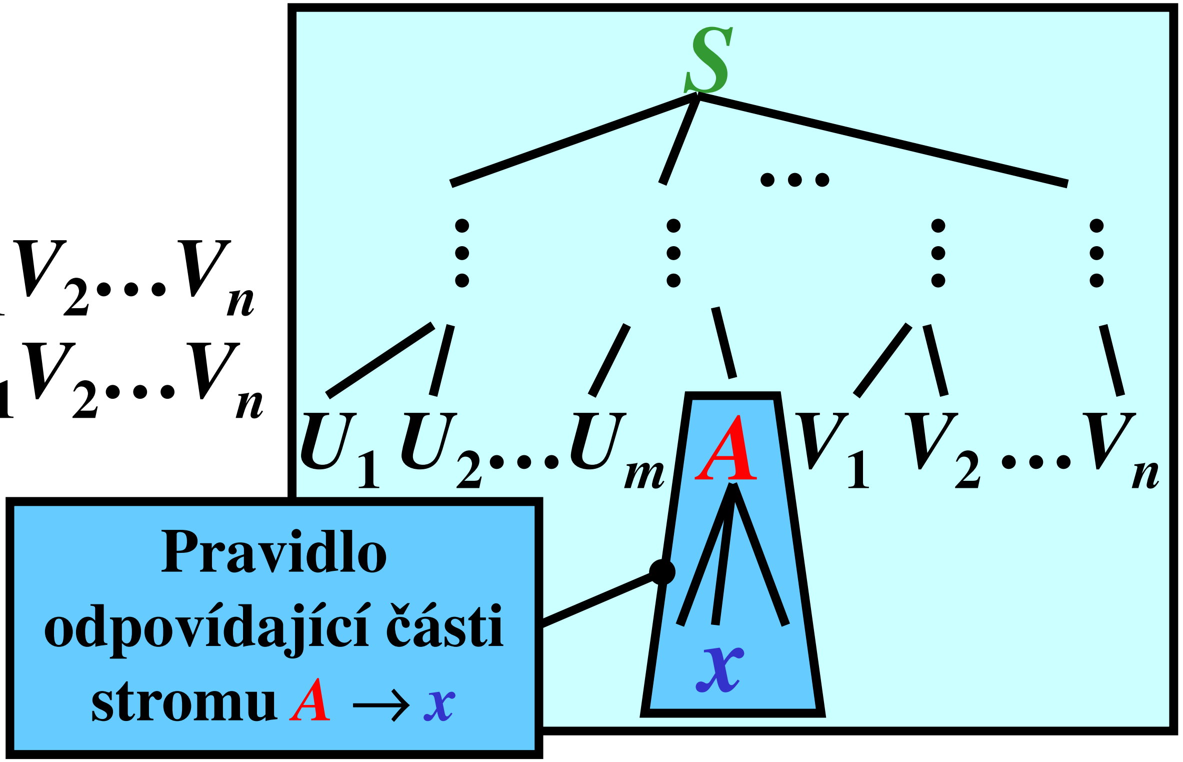


2) $A \to X_1X_2\ldots X_n$:



- **Derivační strom odpovídá použitým pravidlům**

$$S \Rightarrow \ldots$$
$$\vdots$$
$$\Rightarrow U_1U_2\ldots U_m A V_1V_2\ldots V_n$$
$$\Rightarrow U_1U_2\ldots U_m x\, V_1V_2\ldots V_n$$



Derivační strom: Příklad

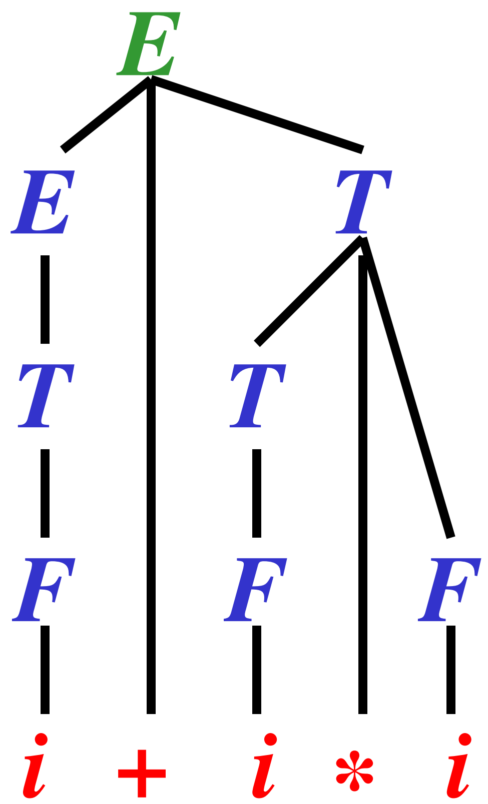
$G = (N, T, P, E)$, kde $N = \{E, F, T\}$, $T = \{i, +, *, (,)\}$,

$P = \{$ 1: $E \rightarrow E+T$, 2: $E \rightarrow T$, 3: $T \rightarrow T*F$, 4: $T \rightarrow F$, 5: $F \rightarrow (E)$, 6: $F \rightarrow i$ $\}$

Jednotlivé derivace:

$$
\begin{array}{lll}
\underline{E} & \Rightarrow E + \underline{T} & [1] \\
 & \Rightarrow E + \underline{T} * F & [3] \\
 & \Rightarrow E + \underline{F} * F & [4] \\
 & \Rightarrow \underline{E} + i * F & [6] \\
 & \Rightarrow T + i * \underline{F} & [2] \\
 & \Rightarrow \underline{T} + i * i & [6] \\
 & \Rightarrow \underline{F} + i * i & [4] \\
 & \Rightarrow i + i * i & [6]
\end{array}
$$

Derivační strom:



Nejlevější derivace

Myšlenka: Během nejlevějšího derivačního kroku je přepsán nejlevější neterminál.

Definice: Nechť $G = (N, T, P, S)$ je BKG, nechť $\boldsymbol{u} \in T^*$, $\boldsymbol{v} \in (N \cup T)^*$, $\boldsymbol{p} = A \rightarrow \boldsymbol{x} \in P$ je pravidlo. Pak $\boldsymbol{uAv}$ přímo derivuje $\boldsymbol{uxv}$ za pomocí *nejlevější derivace* užitím pravidla $\boldsymbol{p}$ v G, zapsáno jako: $\boldsymbol{uAv} \Rightarrow_{lm} \boldsymbol{uxv}\ [\boldsymbol{p}]$

Pozn.: $\Rightarrow_{lm}^+$ a $\Rightarrow_{lm}^*$ je definováno pomocí $\Rightarrow_{lm}$ stejně jako $\Rightarrow^+$ a $\Rightarrow^*$ je dříve definováno pomocí $\Rightarrow$.

Nejlevější derivace: Příklad

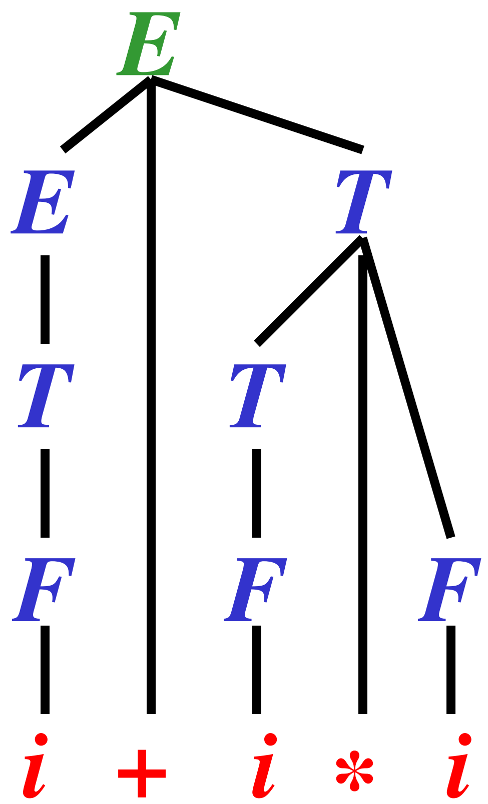
$G = (N, T, P, E)$, kde $N = \{E, F, T\}$, $T = \{i, +, *, (,)\}$,

$P = \{$ 1: $E \rightarrow E+T$, 2: $E \rightarrow T$, 3: $T \rightarrow T*F$,
4: $T \rightarrow F$, 5: $F \rightarrow (E)$, 6: $F \rightarrow i$ $\}$

Nejlevější derivace:

$$
\begin{aligned}
\underline{E} &\Rightarrow_{lm} \underline{E} + T && [1]\\
&\Rightarrow_{lm} \underline{T} + T && [2]\\
&\Rightarrow_{lm} \underline{F} + T && [4]\\
&\Rightarrow_{lm} i + \underline{T} && [6]\\
&\Rightarrow_{lm} i + \underline{T} * F && [3]\\
&\Rightarrow_{lm} i + \underline{F} * F && [4]\\
&\Rightarrow_{lm} i + i * \underline{F} && [6]\\
&\Rightarrow_{lm} i + i * i && [6]
\end{aligned}
$$

Derivační strom:



Nejpravější derivace

Myšlenka: Během nejpravějšího derivačního kroku je přepsán nejpravější neterminál.

Definice: Nechť $G = (N, T, P, S)$ je BKG, nechť $\boldsymbol{u} \in (N \cup T)^*$, $\boldsymbol{v} \in T^*$, $\boldsymbol{p} = A \rightarrow \boldsymbol{x} \in P$ je pravidlo. Pak $\boldsymbol{uAv}$ přímo derivuje $\boldsymbol{uxv}$ za pomocí *nejpravější derivace* užitím pravidla $\boldsymbol{p}$ v G, zapsáno jako: $\boldsymbol{uAv} \Rightarrow_{rm} \boldsymbol{uxv}\ [\boldsymbol{p}]$

Pozn.: $\Rightarrow_{rm}{}^+$ a $\Rightarrow_{rm}{}^*$ je definováno pomocí $\Rightarrow_{rm}$ stejně jako $\Rightarrow^+$ a $\Rightarrow^*$ je dříve definováno pomocí $\Rightarrow$.

Nejpravější derivace: Příklad

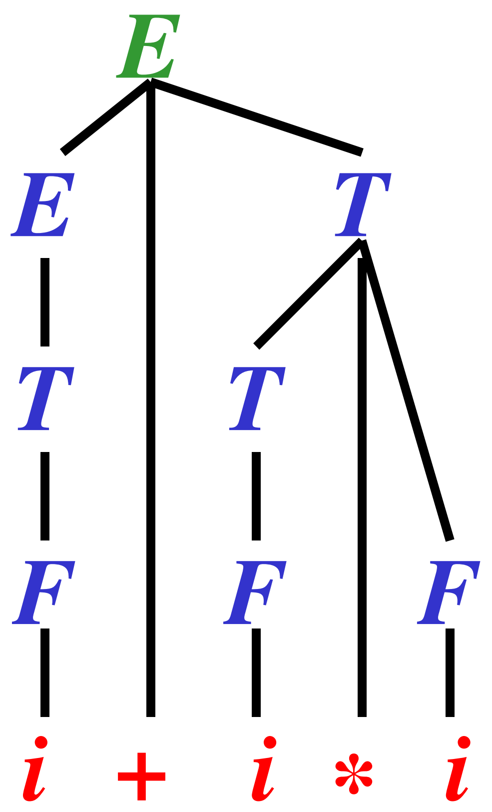
$G = (N, T, P, E)$, kde $N = \{E, F, T\}$, $T = \{i, +, *, (,)\}$,

$P = \{$ 1: $E \rightarrow E+T$, 2: $E \rightarrow T$, 3: $T \rightarrow T*F$, 4: $T \rightarrow F$, 5: $F \rightarrow (E)$, 6: $F \rightarrow i$ $\}$

Nejpravější derivace:

$$
\begin{aligned}
\underline{E} &\Rightarrow_{rm} E + \underline{T} && [1] \\
&\Rightarrow_{rm} E + T * \underline{F} && [3] \\
&\Rightarrow_{rm} E + \underline{T} * i && [6] \\
&\Rightarrow_{rm} E + \underline{F} * i && [4] \\
&\Rightarrow_{rm} \underline{E} + i * i && [6] \\
&\Rightarrow_{rm} \underline{T} + i * i && [2] \\
&\Rightarrow_{rm} \underline{F} + i * i && [4] \\
&\Rightarrow_{rm} i + i * i && [6]
\end{aligned}
$$

Derivační strom:



Derivace: Shrnutí

- Nechť $A \to x \in P$ je pravidlo.

---

1) Derivace:

Nechť $u, v \in (N \cup T)^*$: $uAv \Rightarrow uxv$

Pozn.: Přepsán je <u>libovolný</u> neterminál

---

2) Nejlevější derivace:

Nechť $u \in T^*$, $v \in (N \cup T)^*$: $uAv \Rightarrow_{lm} uxv$

Pozn.: Přepsán je <u>nejlevější</u> neterminál

---

3) Nejpravější derivace:

Nechť $u \in (N \cup T)^*$, $v \in T^*$: $uAv \Rightarrow_{rm} uxv$

Pozn.: Přepsán je <u>nejpravější</u> neterminál

Redukce počtu možných derivací

Myšlenka: Bez újmy na obecnosti můžeme uvažovat používání pouze nejlevějších nebo nejpravějších derivací.

Tvrzení: Nechť $G = (N, T, P, S)$ je BKG.
Následující 3 jazyky jsou totožné:

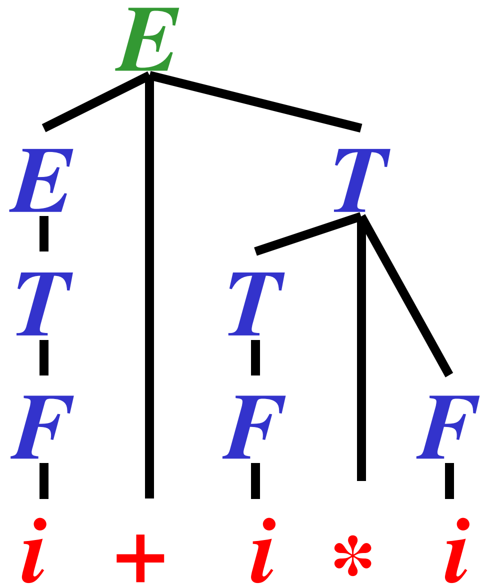
(1) $\{w: w \in T^*, S \Rightarrow_{lm}^* w\}$

(2) $\{w: w \in T^*, S \Rightarrow_{rm}^* w\}$

(3) $\{w: w \in T^*, S \Rightarrow^* w\} = L(G)$

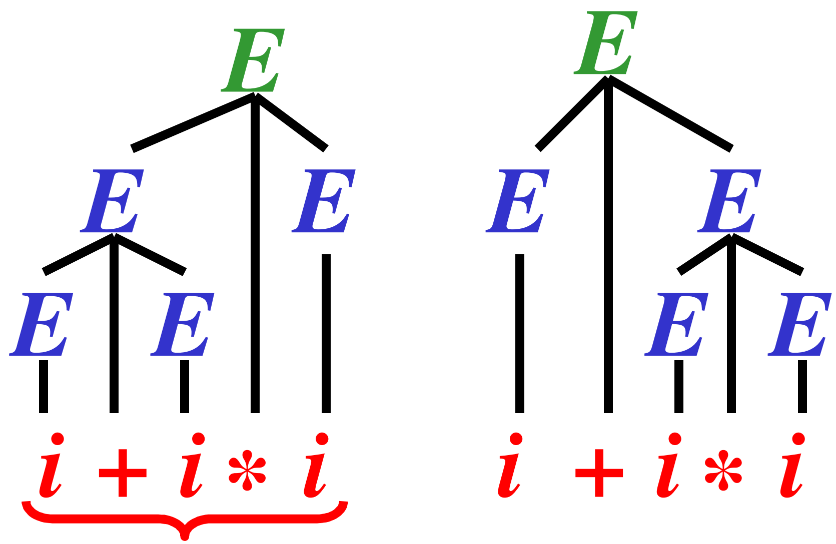
Úvod do nejednoznačnosti

$G_{expr1} = (N, T, P, E)$, kde
$N = \{E, F, T\}$, $T = \{i, +, *, (,)\}$,
$P = \{$ 1: $E \rightarrow E+T$, 2: $E \rightarrow T$,
3: $T \rightarrow T*F$, 4: $T \rightarrow F$,
5: $F \rightarrow (E)$, 6: $F \rightarrow i$ $\}$



Teorie: ☹ × Praxe: ☺

$G_{expr2} = (N, T, P, E)$, kde
$N = \{E\}$, $T = \{i, +, *, (,)\}$,
$P = \{$ 1: $E \rightarrow E+E$, 2: $E \rightarrow E*E$,
3: $E \rightarrow (E)$, 4: $E \rightarrow i$ $\}$



Teorie: ☺ × Praxe: ☹

Pozn.: $L(G_{expr1}) = L(G_{expr2})$

Odstranit v průběhu kompilace!

Gramatická nejednoznačnost

Definice: Nechť $G = (N, T, P, S)$ je BKG. Pokud existuje řetězec $x \in L(G)$ s více jak jedním derivačním stromem, potom G je *nejednoznačná*. Jinak G je *jednoznačná*.

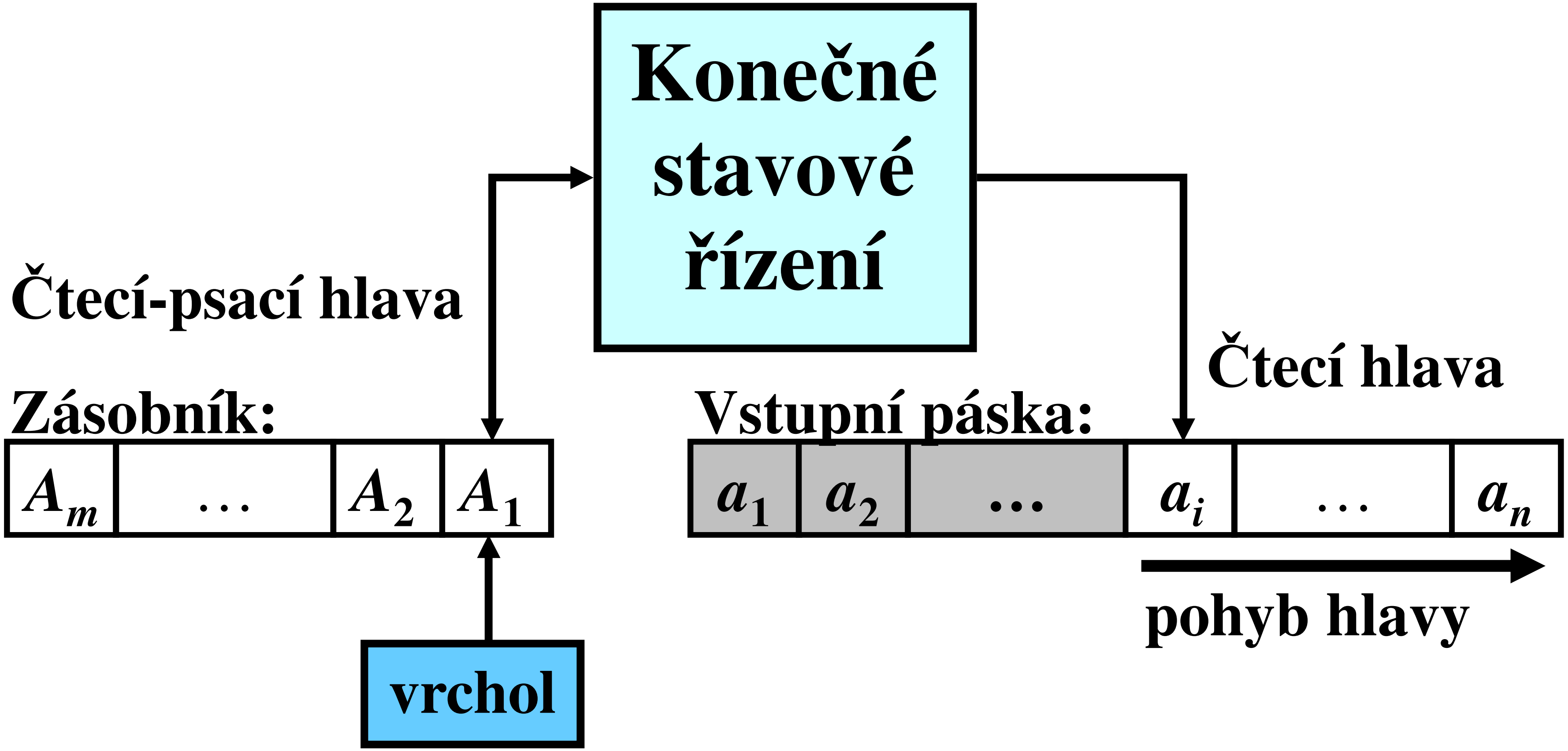
Definice: BKJ L je *vnitřně nejednoznačný*, pokud L není generován žádnou jednoznačnou BKG.

Příklad:

- G_{expr1} je **jednoznačná**, protože pro každé $x \in L(G_{expr1})$ existuje **jeden derivační strom**
- G_{expr2} je **nejednoznačná**, protože pro $i+i*i \in L(G_{expr2})$ existují **dva derivační stromy**
- $L_{expr} = L(G_{expr1}) = L(G_{expr2})$ **není vnitřně nejednoznačný**, protože G_{expr1} **je jednoznačná**

Zásobníkové automaty (ZA)

Myšlenka: Je to KA rozšířený o zásobník



Zásobníkové automaty: Definice

Definice: *Zásobníkový automat* (ZA) je sedmice:

$$M = (Q, \Sigma, \Gamma, R, s, S, F)\text{, kde}$$

- Q je *konečná množina stavů*
- Σ je *vstupní abeceda*
- Γ je *zásobníková abeceda*
- R je *konečná množina pravidel* tvaru $Apa \rightarrow wq$, kde $A \in \Gamma$, $p, q \in Q$, $a \in \Sigma \cup \{\varepsilon\}$, $w \in \Gamma^*$
- $s \in Q$ je *počáteční stav*
- $S \in \Gamma$ je *počáteční symbol na zásobníku*
- $F \subseteq Q$ je množina *koncových stavů*

Poznámky k pravidlům

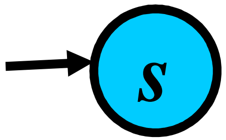
Matematická poznámka k pravidlům:

- Čistě matematicky, R je konečná relace z $\Gamma \times Q \times (\Sigma \cup \{\varepsilon\})$ do $\Gamma^* \times Q$
- Místo relačního zápisu $(Apa, wq) \in R$ zapisujeme $Apa \rightarrow wq \in R$

- **Interpretace pravidel:** $Apa \rightarrow wq$ znamená, že pokud je aktuální stav p, aktuální symbol na vstupní pásce a a symbol na vrcholu zásobníku A, potom M může přečíst a a na zásobníku nahradit A za w a přejít ze stavu p do q.
- **Pozn.**: pokud $a = \varepsilon$, symbol z pásky není přečten

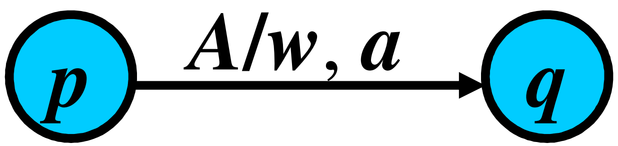
Grafická reprezentace

označuje stav $q \in Q$



označuje počáteční stav $s \in Q$

označuje koncový stav $f \in F$



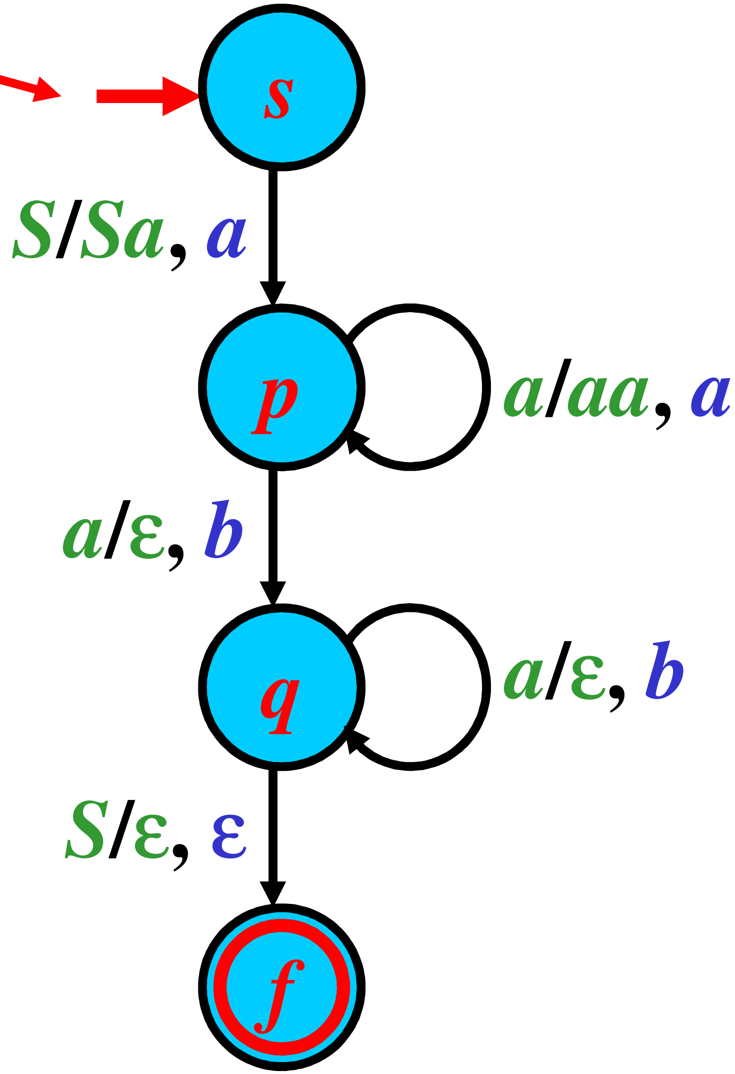
označuje $Apa \to wq \in R$

Grafická reprezentace: Příklad

$M = (Q, \Sigma, \Gamma, R, s, S, F)$

kde:

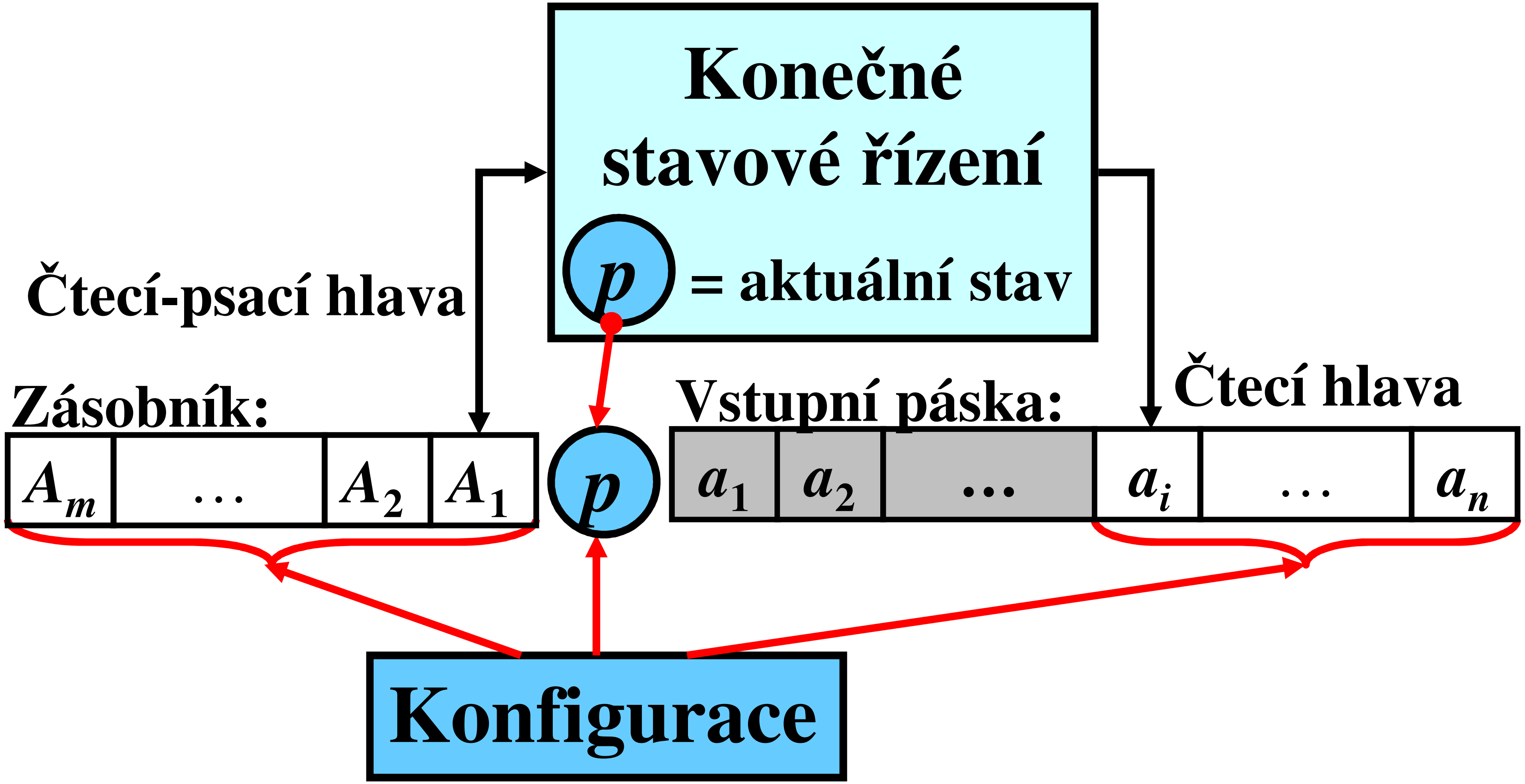
- $Q = \{s, p, q, f\}$;
- $\Sigma = \{a, b\}$;
- $\Gamma = \{a, S\}$;
- $R = \{Ssa \rightarrow Sap,$
  $apa \rightarrow aap,$
  $apb \rightarrow q,$
  $aqb \rightarrow q,$
  $Sq \rightarrow f\}$
- $F = \{f\}$



Konfigurace u ZA

Myšlenka: Instance popisu ZA

Definice: Nechť $M = (Q, \Sigma, \Gamma, R, s, S, F)$ je ZA.
Konfigurace ZA M je řetězec $\chi \in \Gamma^*Q\Sigma^*$



Přechod u ZA

Myšlenka: Jeden výpočetní krok ZA

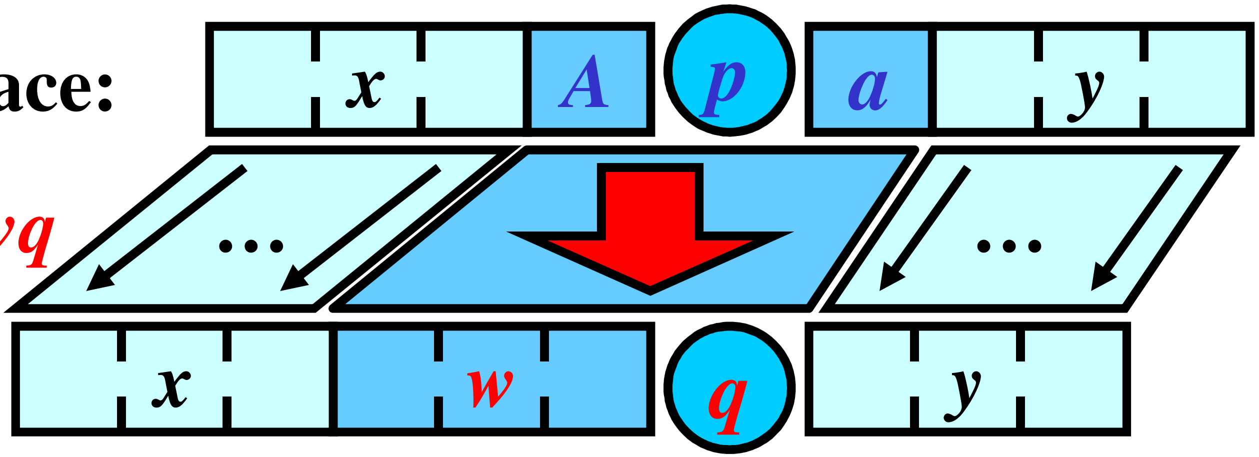
Definice: Nechť $xApay$ a $xwqy$ jsou dvě konfigurace ZA M, kde x, $w \in \Gamma^*$, $A \in \Gamma$, p, $q \in Q$, $a \in \Sigma \cup \{\varepsilon\}$ a $y \in \Sigma^*$. Nechť $r = Apa \rightarrow wq \in R$ je pravidlo. Potom M může provést *přechod* z $xApay$ do $xwqy$ za použití r, zapsáno $xApay \vdash xwqy$ [r] nebo zjednodušeně $xApay \vdash xwqy$.

Pozn.: pokud $a = \varepsilon$, není ze vstupu přečten žádný symbol

Konfigurace:

Pravidlo: $Apa \rightarrow wq$

Nová konfigurace:



Sekvence přechodů 1/2

Myšlenka: několik výpočetních kroků po sobě

Definice: Nechť χ je konfigurace. *M* provede *nula přechodů* z χ do χ; zapisujeme:

$$\chi \vdash^0 \chi \; [\varepsilon] \text{ nebo zjednodušeně } \chi \vdash^0 \chi$$

Definice: Nechť $\chi_0, \chi_1, \ldots, \chi_n$ je sekvence přechodů konfigurací pro $n \geq 1$ a $\chi_{i-1} \vdash \chi_i \; [r_i]$, $r_i \in R$ pro všechna $i = 1, \ldots, n$, což znamená:

$$\chi_0 \vdash \chi_1 \; [r_1] \vdash \chi_2 \; [r_2] \ldots \vdash \chi_n \; [r_n]$$

Pak *M* provede *n-přechodů* z χ_0 do χ_n; zapisujeme:

$\chi_0 \vdash^n \chi_n \; [r_1 \ldots r_n]$ nebo zjednodušeně $\chi_0 \vdash^n \chi_n$

Sekvence přechodů 2/2

Pokud $\chi_0 \vdash^n \chi_n$ [ρ] pro nějaké $n \geq 1$, pak

$$\chi_0 \vdash^+ \chi_n\,[\rho].$$

Pokud $\chi_0 \vdash^n \chi_n$ [ρ] pro nějaké $n \geq 0$, pak

$$\chi_0 \vdash^* \chi_n\,[\rho].$$

Příklad: Uvažujme

$AApabc \vdash ABqbc$ [**1**: $Apa \rightarrow Bq$] a

$ABqbc \vdash ABCrc$ [**2**: $Bqb \rightarrow BCr$].

Potom,

$AApabc \vdash^2 ABCrc$ [**1 2**],

$AApabc \vdash^+ ABCrc$ [**1 2**],

$AApabc \vdash^* ABCrc$ [**1 2**]

Přijímaný jazyk: Tři typy

Definice: Nechť $M = (Q, \Sigma, \Gamma, R, s, S, F)$ je ZA.

1) *Jazyk přijímaný ZA M* ***přechodem do koncového stavu***, značen jako $\boldsymbol{L(M)_f}$, je definován:
   $L(M)_f = \{w: w \in \Sigma^*, Ssw \vdash^* zf, z \in \Gamma^*, f \in F\}$
2) *Jazyk přijímaný ZA M* ***vyprázdněním zásobníku***, značen jako $\boldsymbol{L(M)_\varepsilon}$, je definován:
   $L(M)_\varepsilon = \{w: w \in \Sigma^*, Ssw \vdash^* zf, z = \varepsilon, f \in Q\}$
3) *Jazyk přijímaný ZA M* ***přechodem do koncového stavu a vyprázdněním zásobníku***, značen jako $\boldsymbol{L(M)_{f\varepsilon}}$, je definován:
   $L(M)_{f\varepsilon} = \{w: w \in \Sigma^*, Ssw \vdash^* zf, z = \varepsilon, f \in F\}$

ZA: Příklad

$M = (Q, \Sigma, \Gamma, R, s, S, F)$

kde:

- $Q = \{s, p, q, f\}$;
- $\Sigma = \{a, b\}$;
- $\Gamma = \{a, S\}$;
- $R = \{Ssa \rightarrow Sap,$
  $apa \rightarrow aap,$
  $apb \rightarrow q,$
  $aqb \rightarrow q,$
  $Sq \rightarrow f\}$
- $F = \{f\}$

Otázka: $aabb \in L(M)_{f\varepsilon}$**?**

$S \; s \; a \; a \; b \; b$

Prav.: $Ssa \rightarrow Sap$

$S \; a \; p \; a \; b \; b$

Prav.: $apa \rightarrow aap$

$S \; a \; a \; p \; b \; b$

Prav.: $apb \rightarrow q$

$S \; a \; q \; b$

Prav.: $aqb \rightarrow q$

$S \; q$

Prav.: $Sq \rightarrow f$

f

Prázdný zásobník

Koncový stav

Odpověď: ANO

$Ssaabb \vdash Sapabb \vdash Saapbb \vdash Saqb \vdash Sq \vdash f$

Pozn.: $L(M)_f = L(M)_\varepsilon = L(M)_{f\varepsilon} = \{a^n b^n : n \geq 1\}$

Tři typy přijímaných jazyků: Ekvivalence

Tvrzení:
- $L = L(M_f)_f$ pro ZA $M_f \Leftrightarrow L = L(M_{f\varepsilon})_{f\varepsilon}$ pro ZA $M_{f\varepsilon}$
- $L = L(M_\varepsilon)_\varepsilon$ pro ZA $M_\varepsilon \Leftrightarrow L = L(M_{f\varepsilon})_{f\varepsilon}$ pro ZA $M_{f\varepsilon}$
- $L = L(M_f)_f$ pro ZA $M_f \Leftrightarrow L = L(M_\varepsilon)_\varepsilon$ pro ZA M_ε

Pozn. Existují algoritmy pro následující převody:

ZA $M_{f\varepsilon}$, který přijímá L
přechodem do koncového stavu a vyprázdněním zásobníku

ZA M_f, který přijímá L
přechodem do konc. st.

ZA M_ε, který přijímá L
vyprázdněním zásobníku

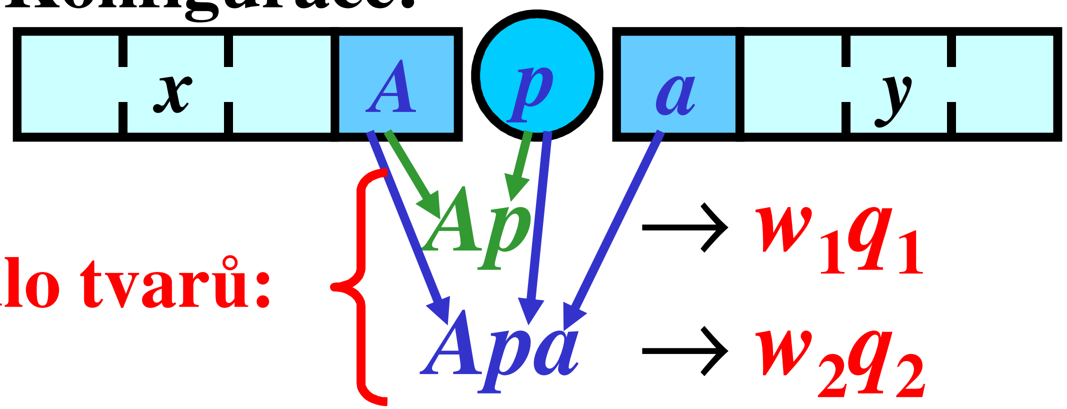
Deterministický ZA (DZA)

Myšlenka: Deterministický ZA může provést z každé konfigurace maximálně jeden přechod

Definice: Nechť $M = (Q, \Sigma, \Gamma, R, s, S, F)$ je ZA. M je *deterministický ZA*, pokud pro každé pravidlo tvaru $\boldsymbol{Apa} \to \boldsymbol{wq} \in R$ platí, že množina $R - \{\boldsymbol{Apa} \to \boldsymbol{wq}\}$ neobsahuje žádné pravidlo s levou stranou $\boldsymbol{Apa}$ nebo $\boldsymbol{Ap}$.

Ilustrace:

Konfigurace:



Maximálně jedno pravidlo tvarů:

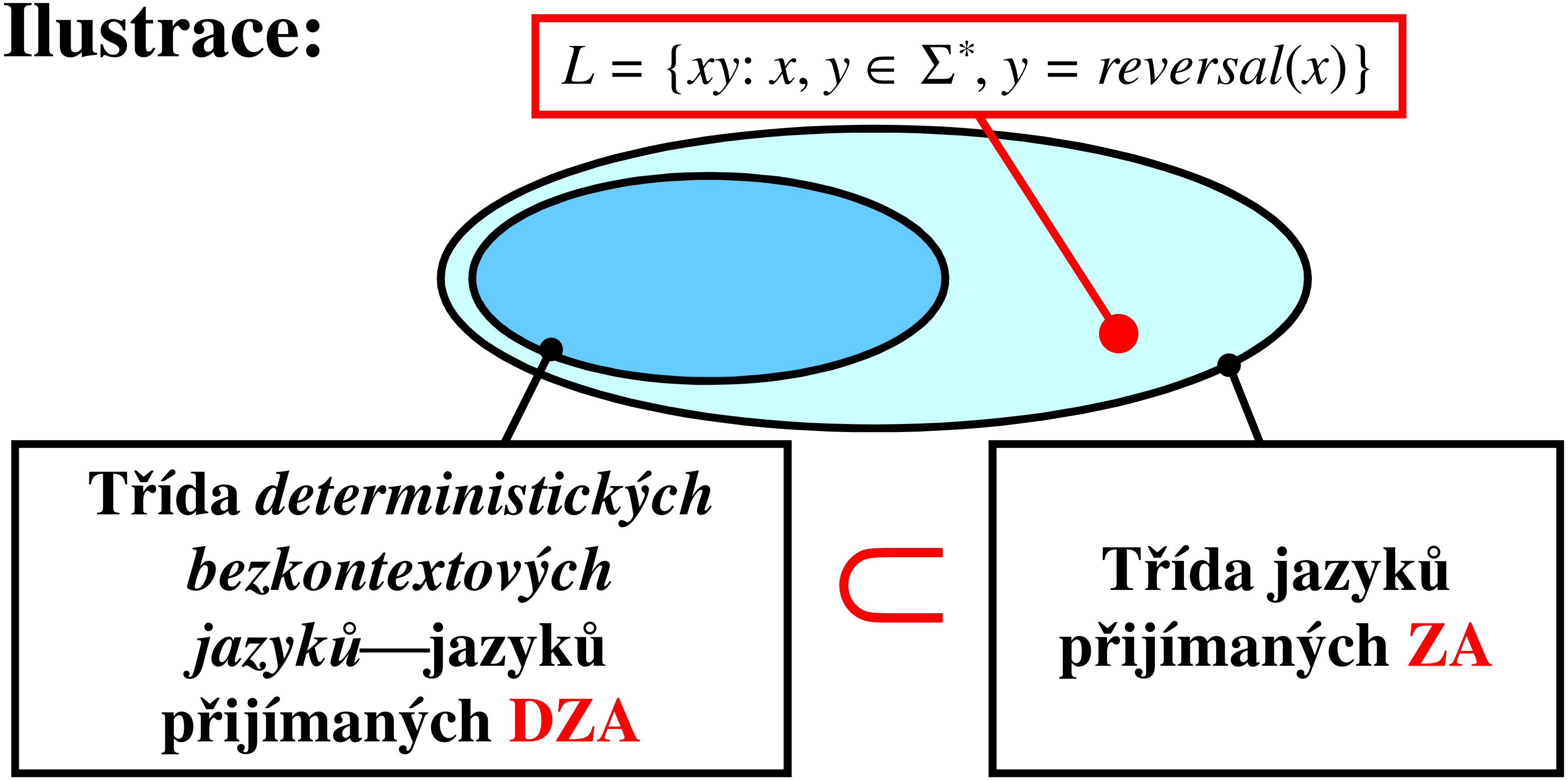
ZA jsou silnější než DZA

Tvrzení: Neexistuje žádný DZA $M_{f\varepsilon}$ přijímající:

$$L = \{xy: x, y \in \Sigma^*, y = reversal(x)\}$$

Důkaz: Viz str. 431 v knize [Meduna: Automata and Languages]

Ilustrace:



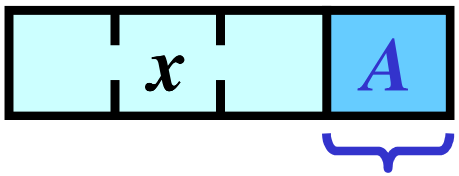
Rozšířený ZA (RZA)

Myšlenka: Z vrcholu zásobníku v RZA lze číst celý řetězec (v ZA to byl pouze jeden symbol)

Definice: *Rozšířený zásobníkový automat* (RZA) je sedmice $M = (Q, \Sigma, \Gamma, R, s, S, F)$, kde $Q, \Sigma, \Gamma, s, S, F$ jsou definovány stejně jako u ZA a R je *konečná množina pravidel* tvaru: $vpa \rightarrow wq$, kde $v, w \in \Gamma^*$, $p, q \in Q$, $a \in \Sigma \cup \{\varepsilon\}$

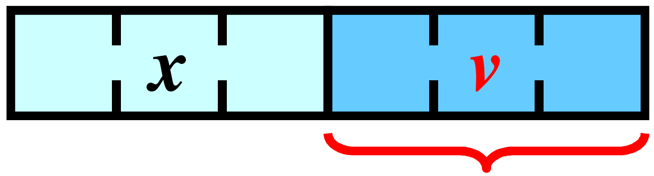
Ilustrace:

Zásobník ZA:



Ze ZA lze číst jeden symbol z vrcholu zásobníku

Zásobník RZA:



Z RZA lze číst řetězec z vrcholu zásobníku

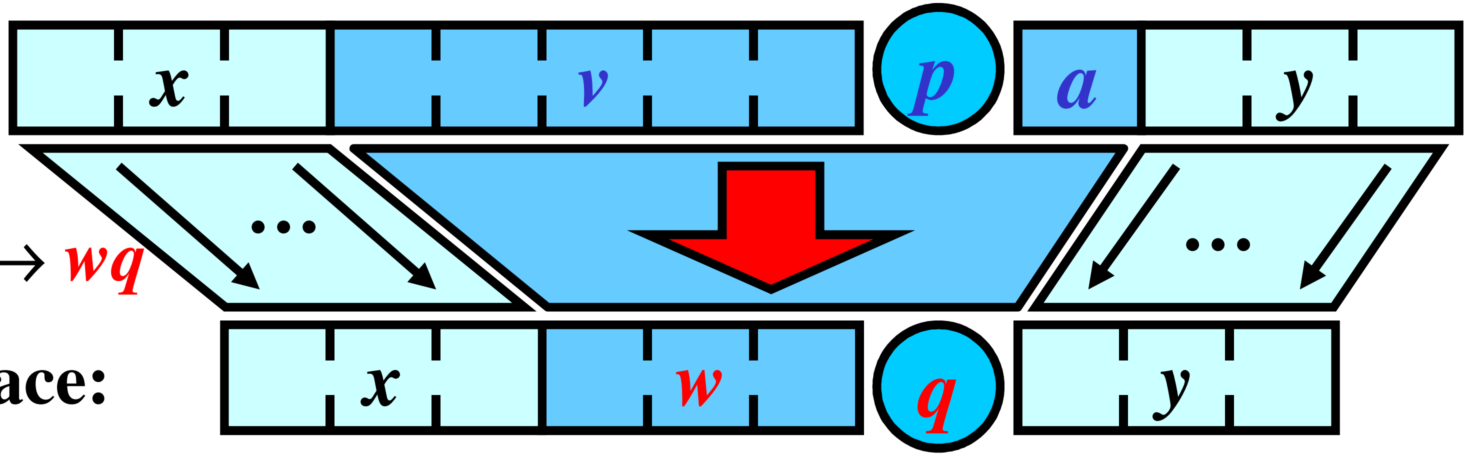
Přechod u RZA

Definice: Nechť $\boldsymbol{xvpay}$ a $\boldsymbol{xwqy}$ jsou dvě konfigurace RZA M, kde $\boldsymbol{x}, \boldsymbol{v}, \boldsymbol{w} \in \Gamma^*$, $\boldsymbol{p}, \boldsymbol{q} \in Q$, $\boldsymbol{a} \in \Sigma \cup \{\varepsilon\}$ a $\boldsymbol{y} \in \Sigma^*$. Nechť $\boldsymbol{r} = \boldsymbol{vpa} \rightarrow \boldsymbol{wq} \in R$ je pravidlo. Potom M může provést *přechod* z $\boldsymbol{xvpay}$ do $\boldsymbol{xwqy}$ za použití $\boldsymbol{r}$, zapsáno: $\boldsymbol{xvpay} \vdash \boldsymbol{xwqy}$ [$\boldsymbol{r}$] nebo $\boldsymbol{xvpay} \vdash \boldsymbol{xwqy}$.

Konfigurace:

Pravidlo: $\boldsymbol{vpa} \rightarrow \boldsymbol{wq}$

Nová konfigurace:



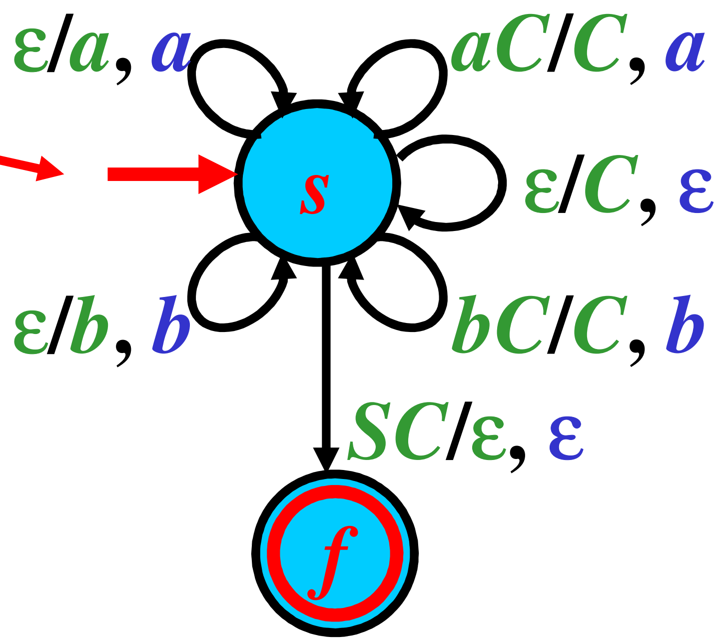
Pozn.: $\vdash^n$, $\vdash^+$, $\vdash^*$, $L(M)_f$, $L(M)_\varepsilon$ a $L(M)_{f\varepsilon}$ jsou definovány stejně jako u ZA.

RZA: Příklad

$M = (Q, \Sigma, \Gamma, R, s, S, F)$

kde:

- $Q = \{s, f\}$;
- $\Sigma = \{a, b\}$;
- $\Gamma = \{a, b, S, C\}$;
- $R = \{ \quad sa \rightarrow as,$
  $sb \rightarrow bs,$
  $s \rightarrow Cs,$
  $aCsa \rightarrow Cs,$
  $bCsb \rightarrow Cs,$
  $SCs \rightarrow f \ \}$
- $F = \{f\}$



Otázka: $abba \in L_{f\varepsilon}(M)$**?**

$Ssabba \vdash Sasbba \vdash Sabsba$
$\vdash SabCsba \vdash SaCsa$
$\vdash SCs \vdash f$

Odpověď: YES

Pozn.: $L(M)_f = L(M)_\varepsilon = L(M)_{f\varepsilon} = \{xy: x, y \in \Sigma^*, y = \text{reversal}(x)\}$

Tři typy přijímaných jazyků: Ekvivalence

Tvrzení:

- $L = L(M_f)_f$ pro RZA $M_f \Leftrightarrow L = L(M_{f\varepsilon})_{f\varepsilon}$ pro RZA $M_{f\varepsilon}$
- $L = L(M_\varepsilon)_\varepsilon$ pro RZA $M_\varepsilon \Leftrightarrow L = L(M_{f\varepsilon})_{f\varepsilon}$ pro RZA $M_{f\varepsilon}$
- $L = L(M_f)_f$ pro RZA $M_f \Leftrightarrow L = L(M_\varepsilon)_\varepsilon$ pro RZA M_ε

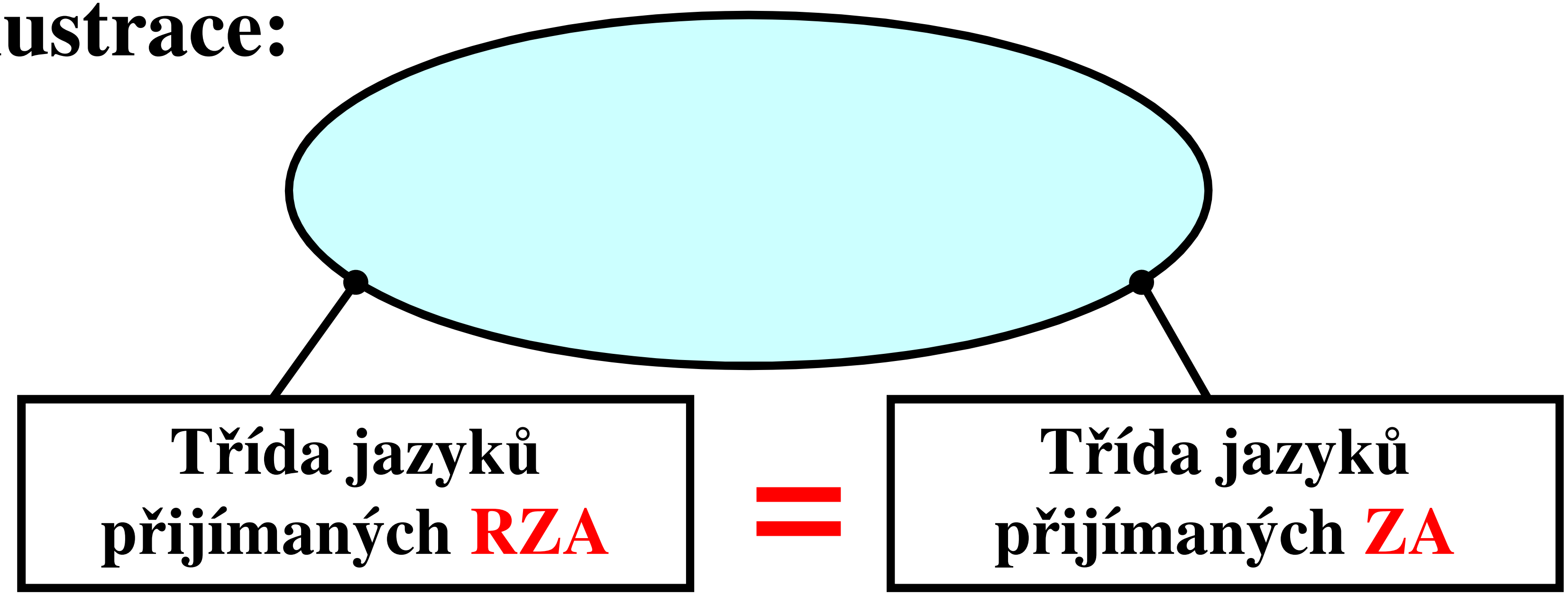
Pozn. Existují algoritmy pro následující převody:

RZA a ZA jsou ekvivalentní

Tvrzení: Pro každý RZA *M* existuje takový ZA *M*', pro který platí: $L(M)_f = L(M')_f$.

Důkaz: Viz str. 419 v knize [Meduna: Automata and Languages]

Illustrace:

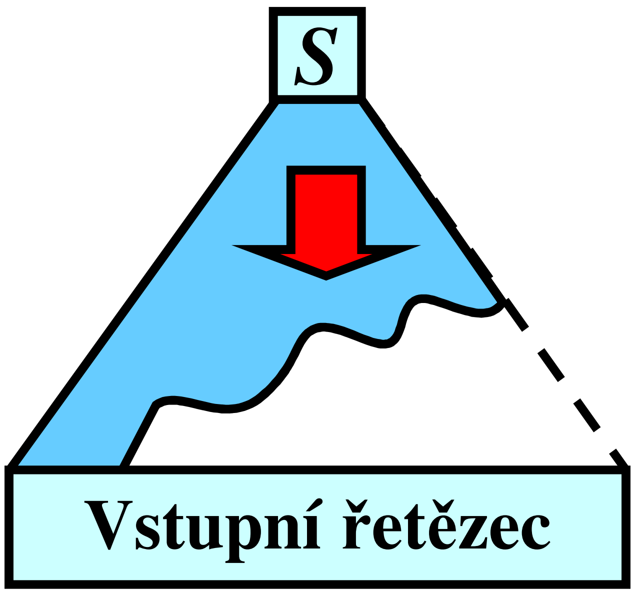


RZA a ZA jako modely pro synt. analýzu

Myšlenka: RZA nebo ZA mohou simulovat konstrukci derivačního stromu pro BKG

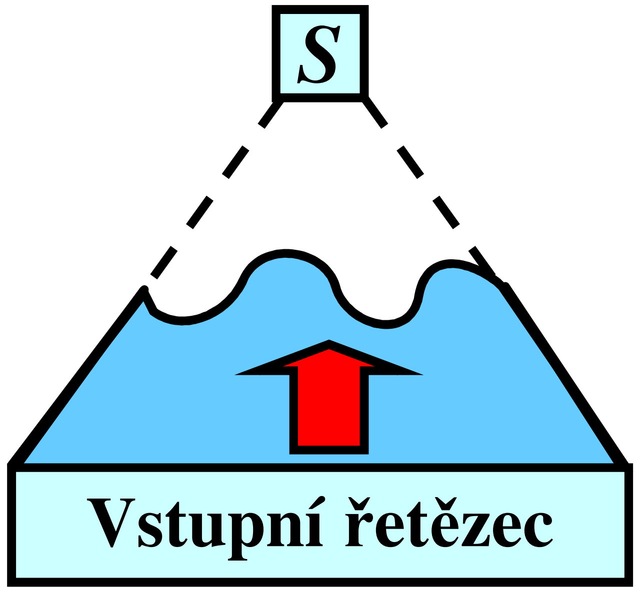
- **Dva základní přístupy:**

1) Shora dolů



Z *S* směrem ke vstupnímu řetězci

2) Zdola nahoru

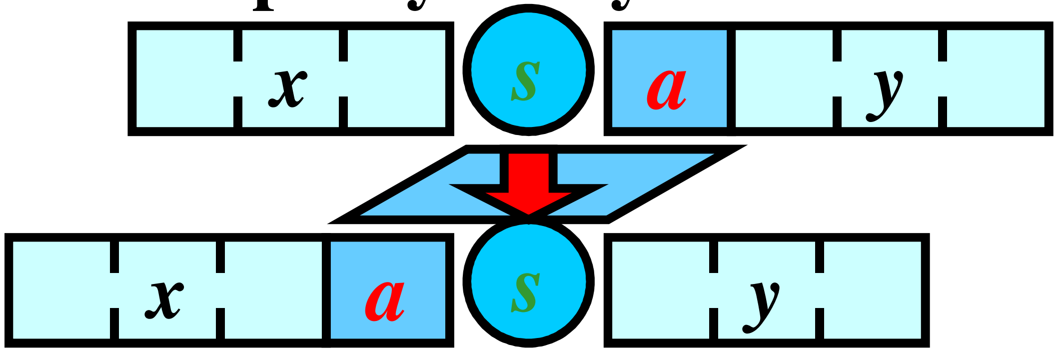


Ze vstupního řetězce směrem k *S*

RZA: Modely pro SA zdola nahoru 1/2

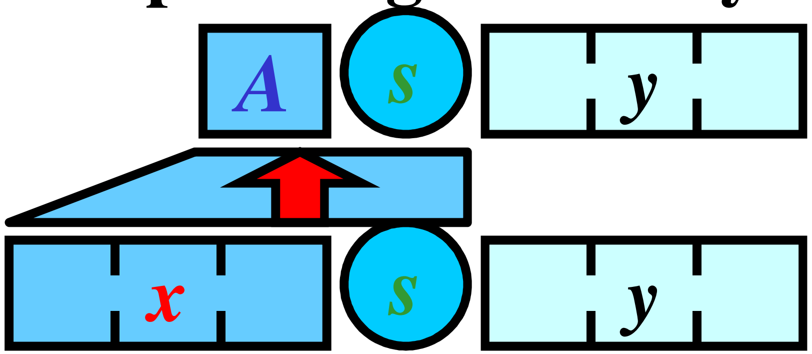
Myšlenka: Na RZA *M* je založena SA pracující zdola nahoru

1) ***M*** **obsahuje** ***shiftovací*** **pravidla, které přesouvají vstupní symboly na zásobník:**



Pro každé $a \in \Sigma$:
přidej $sa \rightarrow as$ do R;

2) ***M*** **obsahuje** ***redukční*** **pravidla, které simulují aplikaci gramatických pravidel pozpátku:**

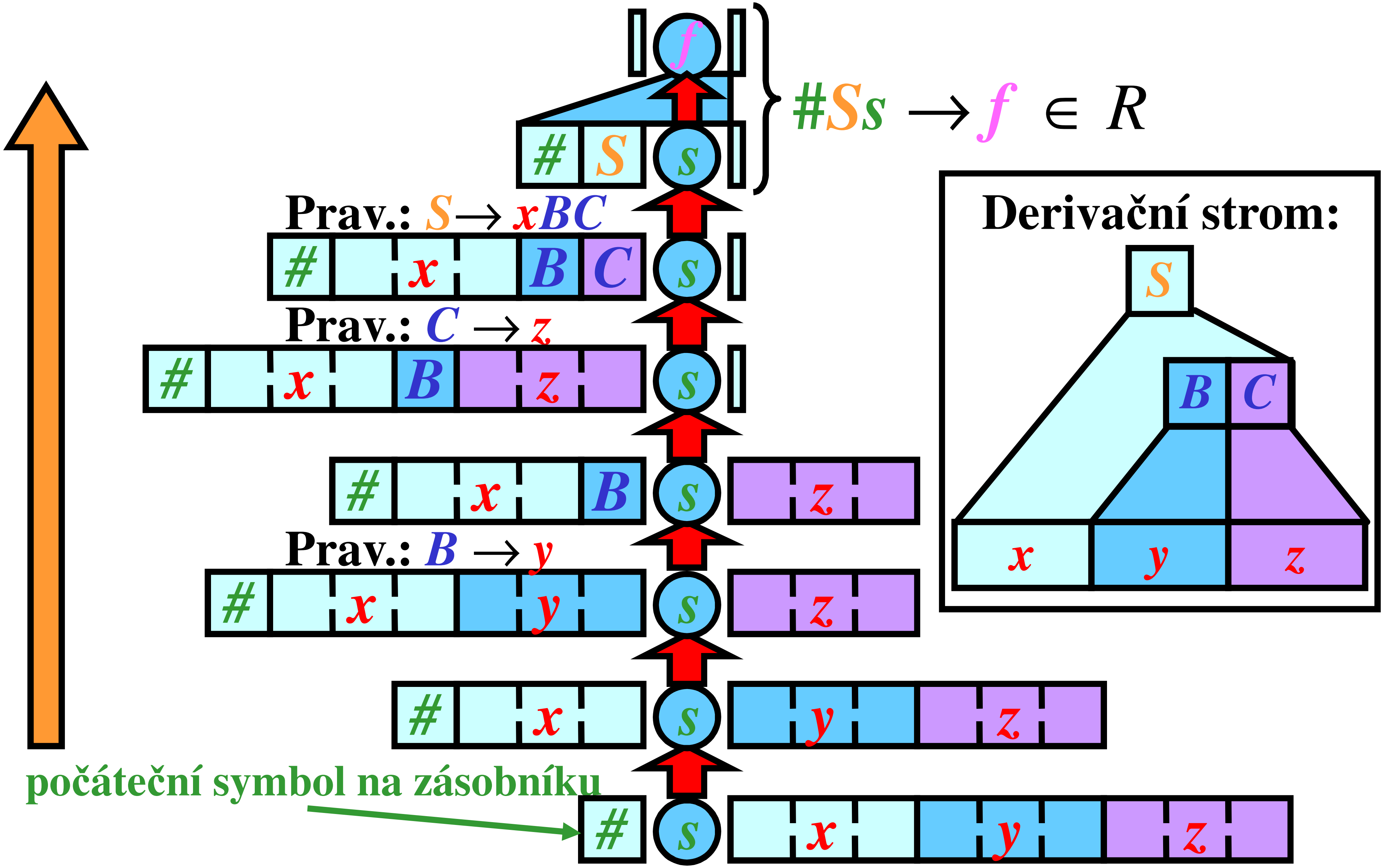


Pro každé $A \rightarrow x \in P$ v G:
přidej $xs \rightarrow As$ to R;

3) ***M*** **také obsahuje speciální pravidlo** $\#Ss \rightarrow f$, **pomocí kterého provede** ***M*** **přechod do koncového stavu**

RZA: Modely pro SA zdola nahoru 2/2

Konstrukce derivačního stromu zdola nahoru:



Algoritmus: Z BKG na RZA

- **Vstup:** BKG $G = (N, T, P, S)$
- **Výstup:** RZA $M = (Q, \Sigma, \Gamma, R, s, \#, F)$; $L(G) = L(M)_f$

- **Metoda:**
- $Q := \{s, f\}$;
- $\Sigma := T$;
- $\Gamma := N \cup T \cup \{\#\}$;
- Konstrukce množiny R:
  - **for each** $a \in \Sigma$: přidej $sa \to as$ do R;
  - **for each** $A \to x \in P$: přidej $xs \to As$ do R;
  - přidej $\#Ss \to f$ do R;
- $F := \{f\}$;

Z BKG na RZA: Příklad 1/2

- $G = (N, T, P, S)$, kde:

$N = \{S\}$, $T = \{(,)\}$, $P = \{S \to (S), S \to (\,)\}$

Máme nalézt: RZA M, pro který platí: $L(G) = L(M)_f$

$M = (Q, \Sigma, \Gamma, R, s, \#, F)$ kde:

$Q = \{s, f\}$; $\Sigma = T = \{(,)\}$; $\Gamma = N \cup T \cup \{\#\} = \{S, (,), \#\}$

“(” $\in T$ **“)” $\in T$** $\boldsymbol{S \to (S) \in P}$ $\boldsymbol{S \to (\,) \in P}$

$$R = \{\underbrace{s(\to (s,\ \ s) \to)s,}_{\textbf{shiftovací pravidla}}\ \underbrace{(S)s \to Ss,\ \ (\,)s \to Ss,}_{\textbf{redukční pravidla}}\ \#Ss \to f\ \}$$

$F = \{f\}$

Z BKG na RZA: Příklad 2/2

$M = (Q, \Sigma, \Gamma, R, s, \#, F)$, kde:

$Q = \{s, f\}$, $\Sigma = T = \{(,)\}$, $\Gamma = \{(,), S, \#\}$, $F = \{f\}$

$R = \{s(\rightarrow (s,\ s) \rightarrow)s,\ (S)s \rightarrow Ss,\ ()s \rightarrow Ss,\ \#Ss \rightarrow f\}$

Otázka: $(()) \in L(M)_f$?

$\#\ s\ (())$

Pravidlo: $s(\rightarrow (s$

$\#(\ s\ ())$ $($

Pravidlo: $s(\rightarrow (s$

$\#((\ s\ ))$ $(($

Pravidlo: $s) \rightarrow)s$

$\#(()\ s\ )$ $(()$

Pravidlo: $()s \rightarrow S$

$\#(S\ s\ )$

Pravidlo: $s) \rightarrow)s$

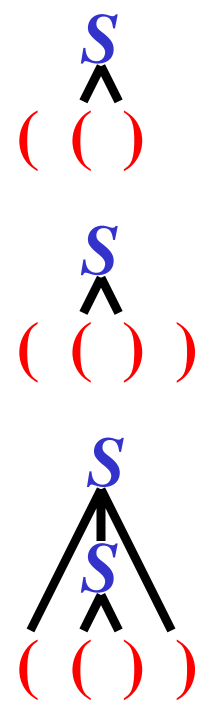
$\#(S)\ s$

Pravidlo: $(S) \rightarrow S$

$\#S\ s$

Pravidlo: $\#Ss \rightarrow f$

f



Koncový stav

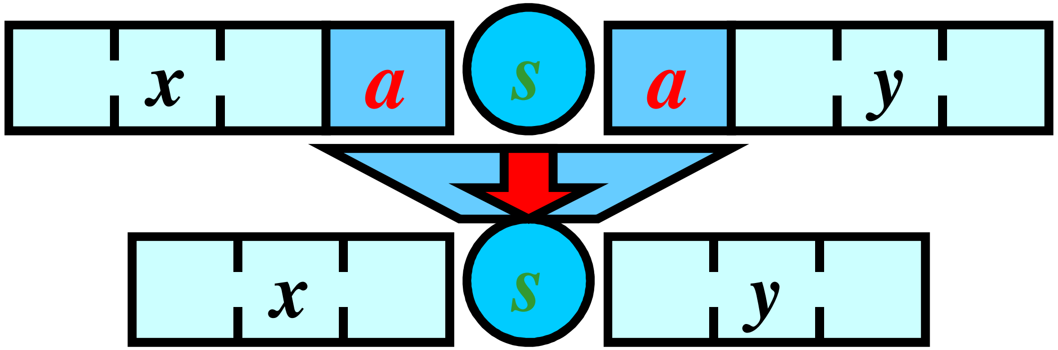
Odpověď: YES

ZA: Modely pro SA shora dolů 1/2

Myšlenka: Na ZA *M* je založena SA pracující shora dolů

---

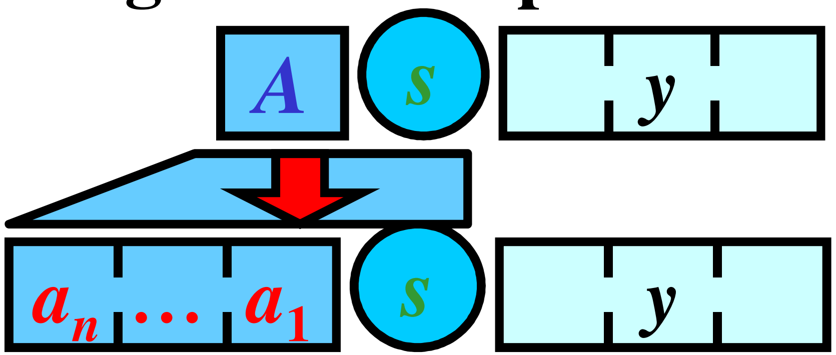
1) *M* obsahuje *porovnávací* pravidla, která porovnají symbol z vrcholu zásobníku a aktuální symbol ze vstupní pásky:



pro každé $a \in \Sigma$:
přidej $asa \rightarrow s$ do R;

---

2) *M* obsahuje *expanzivní* pravidla, která simulují gramatická pravidla:

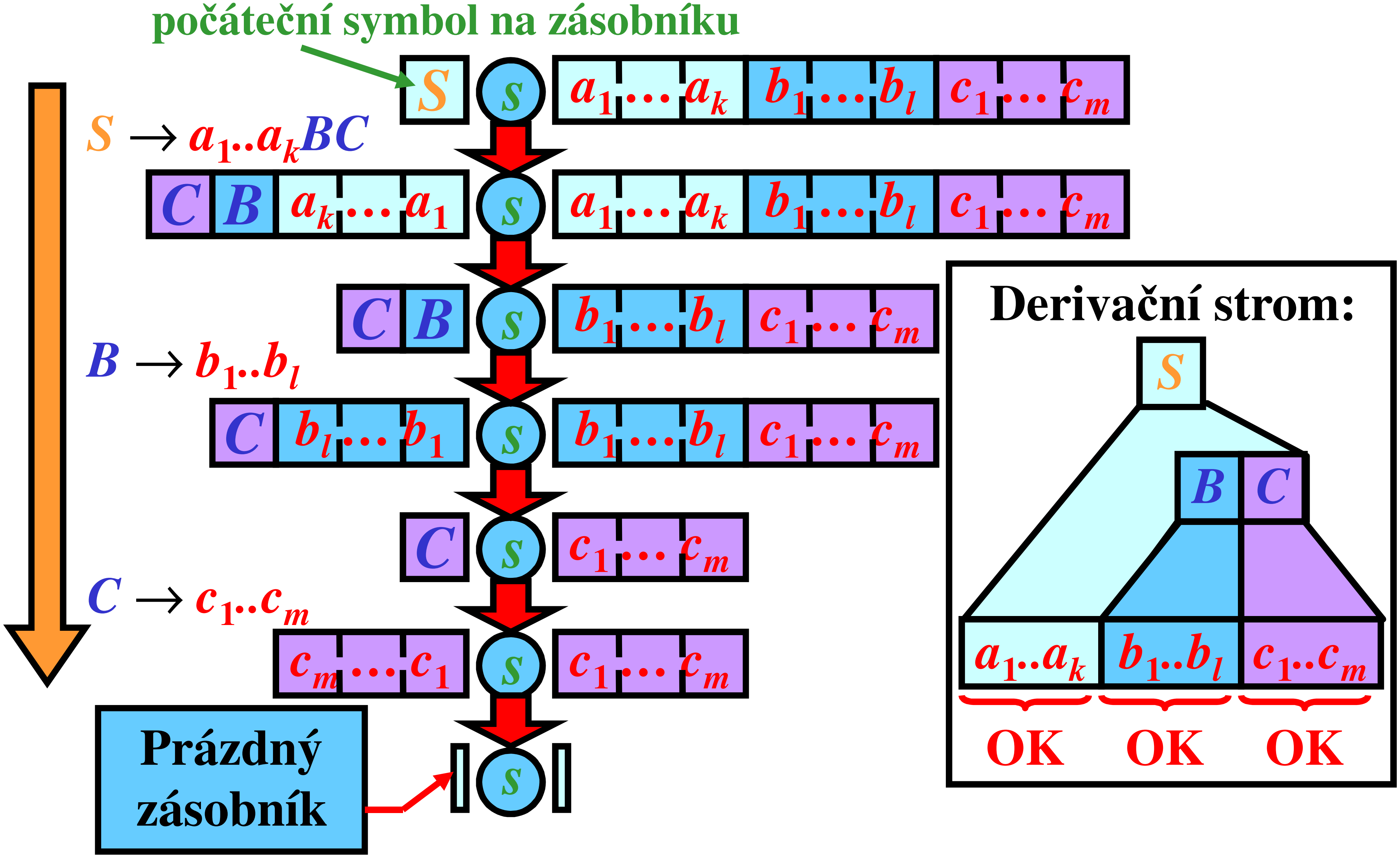


pro každé $A \rightarrow a_1 \dots a_n \in P$ v G,
přidej $As \rightarrow \underbrace{a_n \dots a_1}_{} s$ do R;

= reversal($a_1 \dots a_n$)

ZA: Modely pro SA shora dolů 2/2

Konstrukce derivačního stromu shora dolů:



Algoritmus: Z BKG na ZA

- **Vstup:** BKG $G = (N, T, P, S)$
- **Výstup:** ZA $M = (Q, \Sigma, \Gamma, R, s, S, F)$; $L(G) = L(M)_\varepsilon$

- **Metoda:**
- $Q := \{s\}$;
- $\Sigma := T$;
- $\Gamma := N \cup T$;
- Konstrukce množiny R:
  - **for each** $a \in \Sigma$: přidej $asa \to s$ do R;
  - **for each** $A \to x \in P$: přidej $As \to ys$ do R, kde y = reversal(x);
- $F := \emptyset$;

Z BKG na ZA: Příklad 1/2

- $G = (N, T, P, S)$, kde:

$N = \{S\}$, $T = \{(,)\}$, $P = \{S \to (S), S \to (\,)\}$

Máme nalézt: ZA M, pro který platí: $L(G) = L(M)_\varepsilon$

$M = (Q, \Sigma, \Gamma, R, s, S, F)$ kde:

$Q = \{s\}$; $\Sigma = T = \{(,)\}$; $\Gamma = N \cup T = \{S, (,)\}$

"(" $\in T$ ")" $\in T$ $S \to (S) \in P$ $S \to (\,) \in P$

rev rev

$$R = \{\underbrace{s(\to s,\quad)s) \to s,}_{\text{porovnávací pravidla}}\quad \underbrace{Ss \to)S(s,\quad Ss \to)(s}_{\text{expanzivní pravidla}}\}$$

$F = \emptyset$

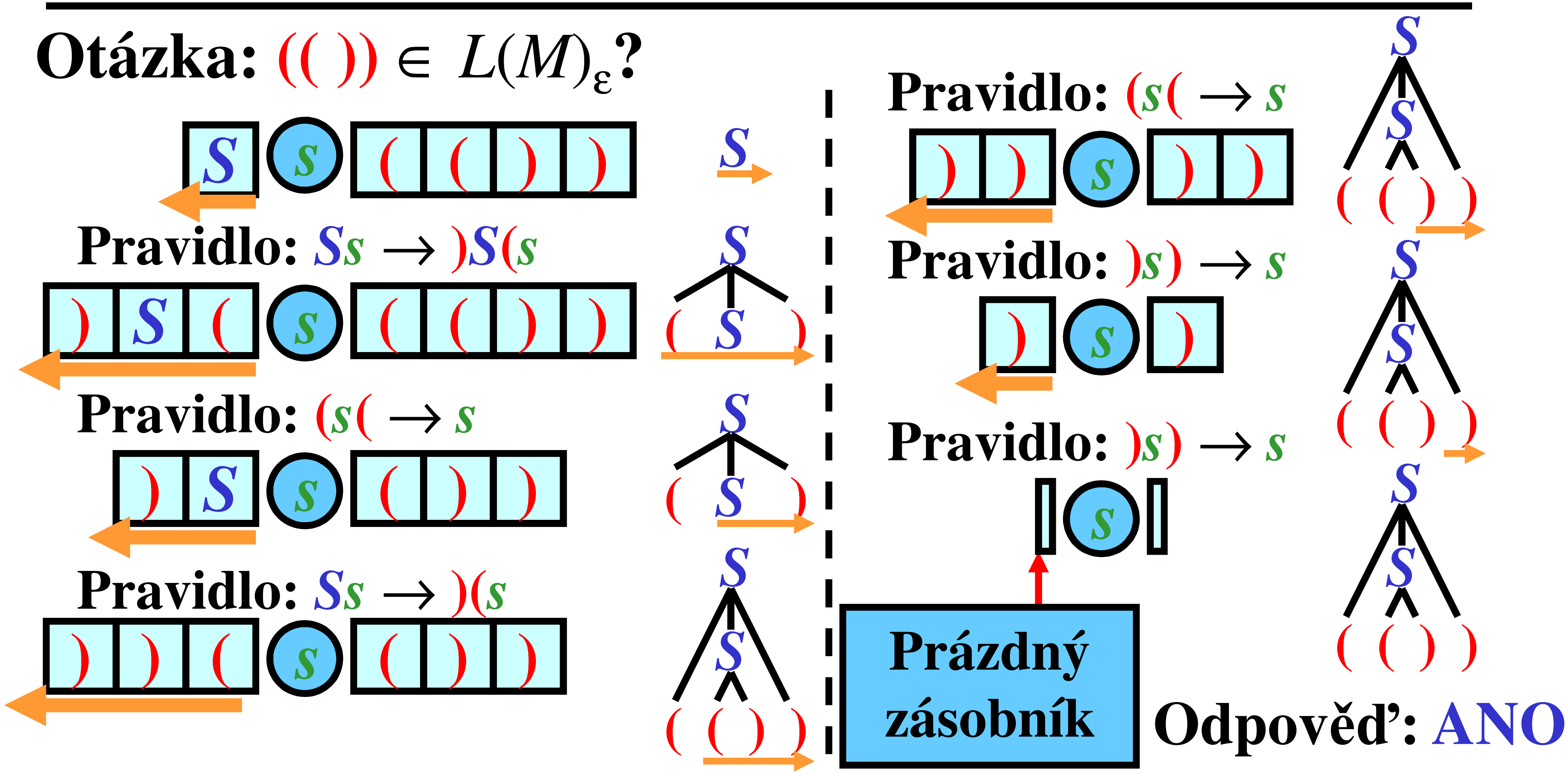
Z BKG na ZA: Příklad 2/2

$M = (Q, \Sigma, \Gamma, R, s, S, F)$, kde:

$Q = \{s\}, \Sigma = T = \{(,)\}, \Gamma = \{(,), S\}, F = \varnothing$

$P = \{(s(\rightarrow s, \quad)s) \rightarrow s, \quad Ss \rightarrow)S(s, \quad Ss \rightarrow)(s \}$

Otázka: $(())\in L(M)_\varepsilon$?



Odpověď: ANO

Modely pro bezkontextové jazyky

Tvrzení: Pro každou BKG G existuje ZA M, pro který platí: $L(G) = L(M)_\varepsilon$.

Důkaz je založen na předchozím algoritmu

Tvrzení: Pro každý ZA M existuje BKG G, pro kterou platí: $L(M)_\varepsilon = L(G)$.

Důkaz: Viz str. 486 v knize [Meduna: Automata and Languages]

Závěr: Fundamentální modely pro bezkontextové jazyky jsou:

1) **Bezkontextové gramatiky** 2) **Zásobníkové automaty**