

UNIVERZITA PALACKÉHO V OLMOUCI  
KATEDRA INFORMATIKY

Martin Rotter (rotter.martinos@gmail.com)  
Jiří Zehnula (jiri.zehnula01@upol.cz)  
Radek Janošík (radek.janostik01@upol.cz)

KMI/FJAA – Formální jazyky a automaty



### **Abstrakt**

Tento dokument je pouze přepisem zápisků a poznámek z přednášek předmětu KMI/FJAA. Přednášel doc. Vilém Vychodil PhD.

## Obsah

<b>1. Historie</b>	<b>1</b>
<b>2. Kódová analýza</b>	<b>1</b>
2.1. Lexikální analýza . . . . .	1
2.2. Syntaktická analýza . . . . .	1
<b>3. Základní pojmy</b>	<b>1</b>
<b>4. Operace s řetězci</b>	<b>2</b>
<b>5. Formální jazyk</b>	<b>3</b>
<b>6. Lexikografické uspořádání</b>	<b>4</b>
<b>7. Operace nad jazyky</b>	<b>4</b>
7.1. Množinové . . . . .	4
7.2. Ostatní . . . . .	5
<b>8. Gramatiky</b>	<b>5</b>
8.1. Přepisovací generovací pravidla . . . . .	5
8.1.1. Vlastnosti pravidel . . . . .	5
8.1.2. Příklady pravidel . . . . .	6
8.1.3. Přímé odvozování řetězců pomocí pravidel . . . . .	6
8.2. Formální gramatiky . . . . .	7
8.3. Hierarchie gramatik . . . . .	8
8.4. Gramatika nezkracující . . . . .	10
8.5. Základní vlastnosti bezkontextových gramatik . . . . .	11
8.6. Jazyky bez gramatického základu . . . . .	13
<b>9. Determinismus versus nedeterminismus</b>	<b>14</b>
<b>10. Konečné deterministické automaty</b>	<b>14</b>
10.1. Reprezentace KDA . . . . .	16
10.1.1. Reprezentace přechodovou tabulkou . . . . .	16
10.1.2. Reprezentace přechodovým diagramem . . . . .	16
10.2. Konfigurace a výpočet KDA . . . . .	16
10.3. Rozšířená přechodová funkce pro KDA . . . . .	18
10.4. KDA s neúplnou přechodovou funkcí . . . . .	19
10.5. Implementace KDA . . . . .	19

<b>11. Konečné nedeterministické automaty</b>	<b>20</b>
11.1. Reprezentace KNA . . . . .	21
11.2. Nedeterministický výpočet . . . . .	21
11.3. Rozšířená přechodová funkce . . . . .	22
11.4. Řetězce přijímané KNA . . . . .	22
11.5. Determinizace KNA . . . . .	23
11.6. Algoritmus pro převod KNA na KDA . . . . .	24
<b>12. Vztah regulárních jazyků a konečných automatů</b>	<b>25</b>
12.1. Regulární jazyky jsou rozpoznatelné KDA (implikace zleva) . . . . .	25
12.2. Jazyky rozpoznatelné KDA jsou regulární (implikace zprava) . . . . .	27
12.3. Regulární gramatiky . . . . .	28
<b>13. Nedeterministický konečný automat s <math>\varepsilon</math>-přechody</b>	<b>30</b>
13.1. Reprezentace $\varepsilon$ KNA . . . . .	31
13.2. Nedeterministický výpočet . . . . .	31
13.3. $\varepsilon$ -uzávěry množin stavů . . . . .	31
13.3.1. Tvorba $\varepsilon$ -uzávěru . . . . .	31
13.4. Rozšířená přechodová funkce . . . . .	32
13.5. Přijímaný řetězec $\varepsilon$ KNA . . . . .	32
13.6. Ekvivalence s KDA . . . . .	33
<b>14. Algoritmus na převod <math>\varepsilon</math> KNA na KDA</b>	<b>33</b>
<b>15. Regulární výrazy</b>	<b>34</b>
<b>16. Jazyky generované regulárními výrazy</b>	<b>35</b>
<b>17. Uzávěrové vlastnosti regulárních jazyků</b>	<b>36</b>
17.1. Základní uzávěrové vlastnosti . . . . .	36
17.2. Další uzávěrové vlastnosti . . . . .	38
17.3. Pumping lemma . . . . .	39
<b>18. Minimalizace KDA</b>	<b>40</b>
18.1. Zprava invariantní ekvivalence . . . . .	40
18.2. Faktorizace automatu . . . . .	41
18.3. Algoritmus pro hledání redukovaného automatu . . . . .	43
<b>19. Izomorfismus automatů</b>	<b>45</b>
<b>20. Bezkontextové gramatiky</b>	<b>45</b>
20.1. Derivační stromy . . . . .	47
20.2. Jednoznačné a nejednoznačné bezkontextové gramatiky . . . . .	49
20.3. Uzávěrové vlastnosti bezkontextových jazyků . . . . .	50
20.4. Bezkontextové gramatiky v programátorské praxi . . . . .	50

<b>21. Nedeterministický zásobníkový automat</b>	<b>51</b>
21.1. Reprezentace NZA . . . . .	52
21.1.1. Přechodová tabulka . . . . .	52
21.1.2. Přechodový diagram . . . . .	53
21.2. Jazyky rozpoznávané NZA . . . . .	53
21.3. Rozpoznání jazyk způsobem přechodu od koncových stavů k vyprázdnění zásobníku . . . . .	54
21.4. Rozpoznání jazyk způsobem přechodu od vyprázdnění zásobníku k přijímání koncovými stavy . . . . .	54
21.5. Automaty pracující s celým zásobníkem . . . . .	55
21.6. Poznámky o vlastnostech NZA . . . . .	55
21.7. Od gramatik k automatům . . . . .	56
21.8. Od automatů ke gramatikám . . . . .	58
<b>22. Deterministické zásobníkové automaty</b>	<b>60</b>

## Seznam obrázků

1.	Grafické znázornění komutativity zřetězení řetězců. . . . .	4
2.	Vychodilovo „vajíčko.“ . . . . .	9
3.	Názorný příklad KDA . . . . .	15
4.	Přechodový diagram pro KDA . . . . .	16
5.	Pseudokód pro převod KNA na KDA. . . . .	24
6.	Pseudokód pro převod $\varepsilon$ KNA na KDA. . . . .	34
7.	Náčrt zásobníkového automatu. . . . .	52
8.	Vkládání hodnot na zásobník u NZA. . . . .	52
9.	Vychodilovo „vajíčko 2.“ . . . . .	62

## Seznam tabulek

1.	Přechodová tabulka pro KDA . . . . .	16
2.	Přechodová tabulka s množinami stavů . . . . .	21
3.	Přechodová tabulka pro 29. příklad . . . . .	31
4.	Tabulkový popis rozkladů stavů . . . . .	44
5.	Ukázka přechodové tabulky pro NZA . . . . .	53

## 1. Historie

Počátek úvah, jež byly později základem seriózního zkoumání formálních jazyků počítačových automatů se datuje do 30. let. Průkopníkem této oblasti byl Noam Chomsky<sup>1</sup>.

Jako příklad selhání autora programovacího jazyka si uveďme jazyk **Fortran**, jehož konstrukce byla po syntaktické stránce špatná, což vedlo ke **gramatické ne jednoznačnosti** tohoto jazyka.

## 2. Kódová analýza

### 2.1. Lexikální analýza

Dělení kódu na tokeny<sup>2</sup>, jež se zapisují například ve stylu  $\langle \text{znak}, \text{identifikátor} \rangle$ . Příkladem je tedy i token  $\langle =, \text{assignment} \rangle$  a jiné.

### 2.2. Syntaktická analýza

Syntaktická analýza vytváří stromovou závislost jednotlivých tokenů, jejíž reprezentace se nazývá *syntaktický-derivační strom*. V rámci této analýzy rozlišme:

1. Teorii jazyků, jež se zabývá stavbou jazyka (respektive jeho syntaxí) a poskytuje tzv. **generativní aparát**. Dodejme, že gramatika říká, v jakém tvaru může být zapsán validní program.
2. Teorii automatů, jež poskytuje tzv. **analytický aparát**. Dodejme, že automatem se rozumí de-facto jednoduchý algoritmus.

## 3. Základní pojmy

- *Symbol* (případně znak). Jedná se o syntaktický pojem (význam tedy nehraje roli), který představuje *jméno* (analogicky k *písmenu* z přirozeného jazyka). Mezi symboly počítejme například **0**, **+**, **Š**, **while**.
- *Abeceda*. Abecedou rozumíme množinu (například množinu  $X$ ) všech přípustných *symbolů* (znaků), přičemž taková množina je neprázdná (tedy  $|x| > 0$ ) a konečná. Konečnost množiny je omezení dané reprezentovatelností množiny v rámci počítačové techniky. Abecedy značíme řeckými písmeny. Například  $\Sigma, \Sigma', \Gamma, \dots, \Omega$ . Například  $\Sigma = \{a, b, c\}$ .
- *Řetězec* (případně slovo). Jedná se o konečnou posloupnost symbolů (znaků) vybraných z nějaké dané abecedy. Například  $\langle a_1, a_2, \dots, a_n \rangle \in \Sigma, n$  nazvěme *délkou řetězce*. Formálně definujeme řetězec jakožto *zobrazení*

$$x : \{1, 2, 3, 4, \dots, n\} \rightarrow \Sigma$$

kde

$$1 \rightarrow a, 2 \rightarrow b, 3 \rightarrow c$$

a tak podobně. Délku řetězce označme  $|x|$ .

<sup>1</sup>Jméno této osoby čti [čomski] a zapamatuj si ke státnicím, že Chomsky byl *nebezpečný levicový intelektuál*.

<sup>2</sup>Překládej jako *část, díl nebo také fráze*.



- *Prázdný řetězec*. Jedná se o řetězec, pro který platí, že  $|x| = 0$  a značíme jej  $\varepsilon$ , přičemž platí následující zápis:

$$\varepsilon \subseteq \emptyset \rightarrow \Sigma$$

Prázdný řetězec není symbolem, tedy  $\varepsilon \notin \Sigma$ .

**Věta 1:** Nad  $k$ -prvkovou abecedou je právě  $k^n$  řetězců délky  $n$ .

**Poznámka 1:** Uvedme si rovněž značení pro dva důležité pojmy:

- $\Sigma^*$  označuje množinu všech řetězců nad abecedou  $\Sigma$ .
- $\Sigma^+$  označuje množinu všech řetězců nad abecedou  $\Sigma$  vyjma  $\varepsilon$ .

## 4. Operace s řetězci

- **Zřetězení** (konkatenace). Jde v podstatě o spojení<sup>3</sup> dvou řetězců v daném pořadí do jednoho řetězce.

**Příklad 1:** Mějme dva řetězce  $a, b$ :

$$a_1 \dots a_n \text{ a } b_1 \dots b_m$$

Pak jejich zřetězení má tvar:

$$a_1 \dots a_n b_1 \dots b_m$$

Identifikátorem<sup>4</sup> operace zřetězení je  $\circ$ , například  $x \circ y$  je zřetězením řetězců  $x$  a  $y$ . Formálně takto:

$$\begin{aligned} x &: \{1, \dots, n\} \rightarrow \Sigma \\ y &: \{1, \dots, m\} \rightarrow \Sigma \\ x \circ y &: \{1, \dots, n + m\} \rightarrow \Sigma \end{aligned}$$

**Poznámka 2:** Algebraicky je tatáž operace zapsána jako  $\langle \Sigma^*, \circ, \varepsilon \rangle$ .

- **Rovnost řetězců** Pro prohlášení dvou řetězců za sobě rovné v zadaném smyslu je třeba splnit obecně dvě následující podmínky:

1. Oba řetězce mají stejnou délku, tedy  $|x| = |y|$ .
2. Bude-li délka označena jako  $n$ , pak musí platit, že  $\forall i | i \in \{1, \dots, n\}, x(i) = y(i)$ . Tedy každé dva k sobě náležící symboly z daných řetězců jsou si rovny.

Uvažujeme-li rovnost řetězců, pak je záhodno uvažovat následující pojmy:

- **Prefix** řetězce. Označme jej  $Pfx(x) = \{y | \exists z \text{ tak, že } yz = x\}$ .
- **Infix** řetězce. Označme jej  $Ifx(x) = \{y | \exists z_1, z_2 \text{ tak, že } z_1 y z_2 = x\}$ .
- **Suffix** řetězce. Označme jej  $Sfx(x) = \{y | \exists z \text{ tak, že } zy = x\}$ .

<sup>3</sup>Pro milovníky jazyka Scheme můžeme tuto operaci přirovnat k proceduře *append*

<sup>4</sup>Identifikátor zřetězení se velmi často v zápisech zřetězení vynechává.

**Věta 2:**

$$xy = xz \implies y = z$$

$$yx = zx \implies y = z$$

Algebraicky je operace zapsána jako  $\langle \Sigma^*, \cdot, \varepsilon \rangle$ .

**Věta 3:** Vyslovme předpoklad, že platí  $xy = uv$ . Pak platí právě jedno z těchto tvrzení:

$$x = u, y = v$$

$$|x| > |u| \text{ a } \exists w|w| \neq \varepsilon, \text{ tak že } x = uw \text{ a } v = wy$$

$$|x| < |u| \text{ a } \exists w|w| \neq \varepsilon, \text{ tak že } u = xw \text{ a } y = wv$$

- **N-tá mocnina řetězce.**

$$x^n = \begin{cases} \varepsilon & \text{pro } n = 0 \\ x & \text{pro } n = 1 \\ xx^{n-1} & \text{pro } x > 1 \end{cases}$$

**Poznámka 3:** Mějme na paměti, že operace mocnění má vyšší prioritu než-li operace konkatenace (zřetězení).

**Věta 4:** Mějme  $u$  a  $v \in \Sigma^*$ , pak platí  $uv = vu$  (komutativita), právě tehdy, když  $\exists z|z| \in \Sigma^*$  a nezáporná celá čísla  $p, q$  tak, že  $u = z^p$  a  $v = z^q$ .

Předpokládejme, že po  $p, z, q$  máme  $u = z^p, v = z^q$ . Pak obecně platí následující zápis:

$$uv = z^p z^q = z^{p+q} = z^q z^p = vu$$

Předpokládejme, že  $uv = vu$ . Indukcí přes  $|uv|$  předpokládáme, že tvrzení platí pro libovolné dva řetězce, jejichž délka zřetězení je menší než-li  $|uv|$ . Mohou nastat tyto případy:

1.  $|u| = |v|$ , pak  $u = v$ , pak  $z = u, p = q = 1$
2.  $|u| < |v|$

Berme v potaz také následující zápis doplněný obrázkem: 1.

$$uw = v$$

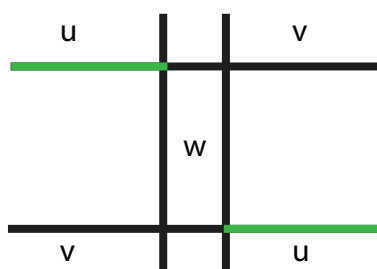
$$wu = v$$

$$uw = wu$$

$$|uw| < |uv|, \text{ tedy } \exists z, p, q \text{ tak, že } u = z^p, w = z^q, v = z^{p+q}$$

## 5. Formální jazyk

Zavedme si pojem *formální jazyk nad množinou všech řetězců*  $\Sigma^*$ . Označme tento



Obrázek 1. Grafické znázornění komutativity zřetězení řetězců.

jazyk jako  $L$ . Pak platí tato tvrzení:

- $L \subseteq \Sigma^*$  (každá podmnožina abecedy je jazykem)
- $L = \emptyset$  (prázdný jazyk)
- $L = \{\varepsilon\}$  (jazyk s prázdným řetězcem)
- $L = \text{jazyk}C++$  (jazyk C++)
- $\vdots$

**Pozor, obecně platí že prázdný jazyk  $\neq$  jazyk s prázdným řetězcem.**

## 6. Lexikografické uspořádání

Předpokládejme uspořádání na množině  $\Sigma^*$ . Nazvěme toto uspořádání *striktním totálním*. Pak toto uspořádání například pro  $\Sigma = \{a_1, \dots, a_n\}$  je  $a_1 < a_2 < a_3 < \dots < a_n$ .

Totální striktní uspořádání označme  $<_l$ .

Položme  $x <_l y$  pro  $x, y \in \Sigma^*$ . To ale platí pokud platí alespoň jedno z následujících dvou tvrzení:

1.  $|x| < |y|$
2.  $|x| = |y|$  a  $\exists i$  tak, že  $x(i) < y(i)$  a zároveň  $x(j) = y(j)$  pro  $\forall j | j < i$

**Příklad 2:**  $\Sigma = \{0, 1\}$ . Triviálně tedy  $0 < 1$ . Následně striktně  $\varepsilon <_l 0 <_l 1 <_l 00 <_l 01 <_l 10 <_l 11$ .

**Věta 5:** Striktní totální uspořádání je asymetrické a tranzitivní. A pro  $x \neq y$  platí buď  $x <_l y$  nebo  $y <_l x$ .

**Důsledek 1:** Důsledkem věty 5 je tvrzení, že množina  $\Sigma^*$  je spočetně nekonečná. Dodejme, že jazyk je (obvykle) spočetná množina.

## 7. Operace nad jazyky

### 7.1. Množinové

Množinové operace nad jazyky jsou prakticky totožné operacím na kterýchkoliv jiných množinách. Můžeme tedy použít množinový průnik, sjednocení, komplement (doplňek) nebo rozdíl.

## 7.2. Ostatní

- **Zřetězení** (produkt) množin. Vyjádříme produkt takto:

$$L_1 L_2 = \{xy | x \in L_1, y \in L_2\}$$

Produkt množin není obecně komutativní, ale je asociativní, přičemž prázdná množina tuto operaci anihiluje. Uvedme si rovněž monoid  $\langle 2^{\Sigma^*}, \circ, \{\varepsilon\} \rangle$ .

- **Mocnina** jazyka. Mocninu vyjádříme takto:

$$L^n = \begin{cases} \{\varepsilon\} & \text{pro } n = 0 \\ LL^{n-1} & \text{pro } n \geq 1 \end{cases}$$

- **Kleeneho**<sup>5</sup> uzávěr neboli **iterace**. Tento uzávěr vyjádříme takto:

$$L^* = \bigcup_{i=0}^{\infty} L^i$$

- **Pozitivní** uzávěr neboli pozitivní iterace. Tento uzávěr vyjádříme takto:

$$L^+ = \bigcup_{i=1}^{\infty} L^i$$

Všimněte si podobností mezi těmito dvěma uzávěry. Pozitivní uzávěr vynechává prázdný řetězec.

## 8. Gramatiky

Jak víme, tak jazyky mohou být *nekonečné* ve smyslu, že obsahují nekonečný počet slov. Nabízí se tedy otázka, jak tyto jazyky rozumně popsat, jak je reprezentovat resp. jak vytvořit *konečnou* sadu pravidel, jejichž aplikace by vedla k opětovné generaci původního jazyka.

### 8.1. Přepisovací generovací pravidla

Pravidlem rozumíme zpravidla každou takto definovanou dvojici.

$$\langle x, y \rangle \in \Sigma^* \times \Sigma^*$$

Pak neformálně tvrdíme, že  $x$  se *přepisuje* na  $y$ . Nutno dodat, že předchozí zápis lze zapsat i například takto.

$$x \rightarrow y, \text{ kde symbol } \rightarrow \notin \Sigma \text{ můžeme prohlásit za tzv. metasymbol.}$$

#### 8.1.1. Vlastnosti pravidel

- *Nezkracující* pravidlo je pravidlo, o kterém platí, že  $|x| \leq |y|$ . Tedy aplikaci tohoto pravidla na vstupní řetězec určité nevznikne řetězec kratší, než-li jeho předloha.
- $\varepsilon$  - pravidlo je pravidlo tvaru  $x \rightarrow \varepsilon$ .

---

<sup>5</sup>Stephen Cole Kleene je známý matematik, jenž se významně podílel na položení základů teoretických počítačových věd.

### 8.1.2. Příklady pravidel

**Příklad 3:** Mějme zadání abecedy  $\Sigma = \{a, b, c\}$ . Pravidla s využitím této abecedy by mohla být například tato.

$$\begin{aligned} aa &\rightarrow bc \\ bb &\rightarrow abba \\ c &\rightarrow \varepsilon \end{aligned}$$

**Příklad 4:** Mějme další zadání abecedy  $\Sigma = \{expr, +, \times\}$ . Pravidla s využitím této abecedy by mohla být například tato.

$$\begin{aligned} expr &\rightarrow expr + expr \\ expr &\rightarrow expr \times expr \end{aligned}$$

### 8.1.3. Přímé odvozování řetězců pomocí pravidel

Uvažujme odvozovací pravidlo  $x \rightarrow y$  nad abecedou  $\Sigma$ , pak řekneme, že řetězec  $v$  je **přímo odvozen** z řetězce  $u$  pomocí pravidla  $x \rightarrow y$ , pokud  $\exists p, q \in \Sigma^*$  tak, že

$$\begin{aligned} u &= pxq \\ v &= pyq \end{aligned}$$

Značení předchozí operace je následující:

$$u \Rightarrow_{x \rightarrow y} v$$

Slovně bychom tento zápis vystihli jako „přímý přepis dle pravidla  $x \rightarrow y$ .“

Řetězec  $v$  vznikne přímým přepisem z  $u$  pomocí pravidel  $P \subseteq \Sigma^* \times \Sigma^*$ , pokud  $\exists \pi \in P$  tak, že  $u \Rightarrow_{\pi} v$ .

Značme  $u \Rightarrow_P v$ .  $P$  je množinou užitých pravidel.  $P$  i  $\Rightarrow_P$  jsou binární relace na  $\Sigma^*$  a  $P \subseteq \Rightarrow_P$ , tedy „ $P$  je podmnožinou šipky  $p$ .“ Platí, že  $x \rightarrow y \in P$  a  $x \Rightarrow_{x \rightarrow y} y$ .

**Příklad 5:** Mějme abecedu  $\Sigma = \{a, b, c\}$  a soubor pravidel  $P = \{aa \rightarrow bc, a \rightarrow cab, bb \rightarrow \varepsilon\}$ . Pak by odvození v jednom kroku mohla vypadat například takto:

$$\begin{aligned} baaa &\rightarrow bbca \\ bac &\rightarrow bcabc \end{aligned}$$

**Definice 1:** Definujme pojem **derivace**. Jedná se o posloupnost řetězců ve tvaru:

$$x_0, \dots, x_k, \text{ kde } k \geq 0 \text{ a kde } \{x_0, \dots, x_k\} \in \Sigma^*$$

se nazývá **P-derivace délky  $k$** , pokud  $x_{i-1} \Rightarrow_P x_i, \forall 1 \leq i \leq k$ . Symbolicky totéž  $x_0 \Rightarrow_P x_1 \Rightarrow_P \dots \Rightarrow_P x_k$ . Počet odvození tedy značí *délku* derivace.

Pokud pro  $u, v \in \Sigma^*$   $\exists$  P-derivace  $u = x_0 \dots x_k = v$ , pak říkáme, že  $v$  je odvozeno z  $u$  pomocí pravidel z  $P$ , což značíme například  $u \Rightarrow_P^* v$ , tímto je pochopitelně myšleno odvození ve více krocích. Platí, že  $P \subseteq \Rightarrow_P \subseteq \Rightarrow_P^+$ .

**Příklad 6:** Mějme abecedu  $\Sigma = \{a, \dots, z\}$  a pravidla stejná jako v příkladu 5. Nyní odvozujeme například takto:

$$\underline{b}aaa, \underline{b}bca, \underline{c}a, \underline{c}cab$$

## 8.2. Formální gramatiky

Mějme následující entity:

- $\Sigma$  - abeceda terminálních symbolů (tyto symboly tvoří řetězce daného jazyka).
- $N$  - abeceda neterminálních symbolů (tyto symboly se užívají k řízení průběhu odvozování).

Dodejme, že obě množiny by měly být neprázdné a konečné.

**Definice 2:** Odvozovací pravidlo  $x \rightarrow y$  se nazývá *generativní*, pokud  $x$  obsahuje alespoň jeden neterminální symbol.

**Definice 3:** Mějme strukturu  $G = \langle N, \Sigma, P, S \rangle$ , kde  $N$  je abecedou neterminálních symbolů,  $\Sigma$  je abecedou terminálních symbolů,  $P$  je množinou odvozovacích pravidel a  $S \in N$  je tzv. *počátečním* resp. *startovním* neterminálem. Pak tuto čtveřici nazveme **gramatikou**.

**Poznámka 4:** Pokud chceme vyjádřit, že z jednoho symbolu odvozujeme několik možných alternativ, tak to zapíšeme místo klasického dlouhého zápisu  $y \rightarrow x_1, y \rightarrow x_2, \dots$  pomocí zkrácené notace např.  $y \rightarrow x_1 | x_2 | \dots$

**Příklad 7:** Gramatika může vypadat třeba takto:

$$\begin{aligned} N &= \{\varepsilon, S, D, I\} \\ \Sigma &= \{0, \dots, 9, +, -\} \\ P &= \{S \rightarrow -I \mid I \mid I, I \rightarrow DI \mid D, D \rightarrow 0 \mid 1 \mid \dots \mid 9\} \\ G &= \langle N, \Sigma, P, S \rangle \end{aligned}$$

**Příklad 8:** Nebo takto:

$$\begin{aligned} N &= \{S, X, Y\} \\ \Sigma &= \{a, b, c\} \\ P &= \{S \rightarrow XcYcX, X \rightarrow aX, X \rightarrow bX, X \rightarrow cX, X \rightarrow \varepsilon, Y \rightarrow abY, Y \rightarrow ab\} \\ G &= \langle N, \Sigma, P, S \rangle \end{aligned}$$

**Definice 4:** Každý řetězec  $x \in (N \cup \Sigma)^*$ , pro který platí  $S \rightarrow^* x$ , je **větná forma** gramatiky  $G = \langle N, \Sigma, P, S \rangle$ . Větná forma se nazývá **větou**, pokud  $x \in \Sigma^*$ .

**Definice 5:** Jazyk generovaný gramatikou definujeme jako:

$$L(G) = \{x \in \Sigma^* \mid S \Rightarrow_G^* x\}$$

Vidíme tedy, že takový jazyk obsahuje *věty*, které lze odvodit ze startovacího neterminálu pomocí pravidel této gramatiky.

**Příklad 9:** Tento příklad čerpá gramatiku z příkladu 8.

$$\begin{aligned} S &\Rightarrow_G^* abbccYcX \\ S &\Rightarrow_G^* Xcababababc \\ S &\Rightarrow_G^* cYcbaX \\ S &\Rightarrow_G^* abbccabca \\ S &\Rightarrow_G^* cabababc \end{aligned}$$

**Definice 6:** Gramatiky  $G_1$  a  $G_2$  jsou **ekvivalentní**, pokud generují stejný jazyk.

### 8.3. Hierarchie gramatik

- **Gramatiky typu 0** – jedná se o gramatiky bez omezení.
- **Gramatiky typu 1** – jedná se o tzv. *kontextové* nebo *kontextově závislé* gramatiky. Ty splňují následující omezení na tvar pravidel. Pro každé pravidlo gramatik tohoto typu platí, že:
  1. Buď je (pravidlo) ve tvaru  $pAq \rightarrow pxq$ , kde  $p, q \in (\Sigma \cup N)^*$ ,  $A \in N$ ,  $x \in (\Sigma \cup N)^*$ , kde  $p$  a  $q$  se nazývají levým resp. pravým **kontextem**.
  2. Nebo je (pravidlo) ve tvaru  $S \rightarrow \varepsilon$ , kde  $S$  je startovní terminál gramatiky, ale pouze za předpokladu, že  $S$  se nevyskytuje na pravé straně žádného pravidla.

Zároveň platí pro každé pravidlo (s výjimkou pravidla  $S \rightarrow \varepsilon$ ), že délka odvozeného řetězce je minimálně stejně velká jako délka vstupního řetězce. Gramatika tedy zároveň obsahuje pouze tzv. *nezkracující* pravidla.

- **Gramatiky typu 2** – jedná se o tzv. *bezkontextové* gramatiky, jenž obsahují pravidla ve tvaru:

$$A \rightarrow x, \text{ kde } A \in N, x \in (\Sigma \cup N)^*$$

Na levých stranách pravidel tedy očekáváme pouze neterminální symbol a na pravé straně očekáváme minimálně jeden symbol (s výjimkou  $\varepsilon$ -pravidla).

**Příklad 10:** Mějme tuto gramatiku:

$$\begin{aligned} G &= \langle N, \Sigma, P, S \rangle \\ N &= \{A, S\} \\ \Sigma &= \{0, 1\} \\ P &= \{S \rightarrow 0A, A \rightarrow \varepsilon\} \end{aligned}$$

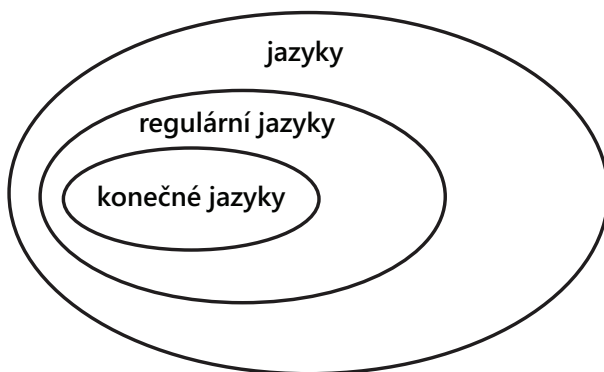
- **Gramatiky typu 3** – jedná se o tzv. *regulární* resp. *pravolínéární* gramatiky, které obsahují pravidla ve třech následujících tvarech:
  1.  $A \rightarrow bB$ , kde  $A, B \in N, b \in \Sigma$
  2.  $A \rightarrow a$
  3.  $S \rightarrow \varepsilon$  (příčemž platí stejné podmínky, jako u gramatik typu 1)

**Věta 6:** Každý konečný jazyk je regulární.

**Důkaz 1:** Mějme jazyk  $L = \{x_1, \dots, x_n\}$ . Abychom tento jazyk prohlásili za regulární, tak je třeba najít regulární gramatiku, která tento jazyk generuje.

Mějme tedy nějaké dané  $\Sigma$  a  $S$  a zvolme  $N$ . Následně platí  $\forall x_i \in L$  je dvojího typu:

1.  $x_i = \varepsilon$  a následně  $S \rightarrow \varepsilon$
2.  $x_i = a_1 \dots a_k$  a následně  $S \rightarrow a_{i1}A', A' \rightarrow a_{i2}A'', \dots, A^k \rightarrow a_{ik}$



Obrázek 2. Vychodilovo „vajíčko.“

**Příklad 11:**

$$\begin{aligned} N &= \{S\} \\ \Sigma &= \{a, b\} \\ P &= \{S \rightarrow aSb | \varepsilon\} \\ L(G) &= \{a^n b^n \mid n \geq 0\} \end{aligned}$$

Máme tedy *bezkontextový* jazyk.

**Příklad 12:**

$$\begin{aligned} N &= \{S\} \\ \Sigma &= \{a, b\} \\ P &= \{S \rightarrow SS | aSb | bSa | \varepsilon\} \end{aligned}$$

$L(G)$  je *bezkontextový* jazyk.

**Příklad 13:**

$$\begin{aligned} N &= \{S, V\} \\ \Sigma &= \{p, ), (, \Rightarrow, !\} \\ P &= \{S \rightarrow V | (S \Rightarrow S) | !S, V \Rightarrow pV | p\} \end{aligned}$$

$L(G)$  je jazyk všech výrokových formulí.



### 8.4. Gramatika nezkracující

Gramatika  $G$  se nazývá nezkracující, pokud má pouze nezkracující pravidla. Avšak navíc může mít tato gramatika pravidlo ve tvaru  $S \rightarrow \varepsilon$ , přičemž  $S$  se nesmí nacházet na pravé straně žádného pravidla.

**Příklad 14:**

$$\begin{aligned} N &= \{S, A, B, C\} \\ \Sigma &= \{a, b, c\} \\ P &= \{S \rightarrow \varepsilon | abc | Ac, A \rightarrow aBcb, Bcb \rightarrow bBc, Bcc \rightarrow Ccc, bc \rightarrow Cb, aC \rightarrow aab | aA\} \end{aligned}$$

**Věta 7:** Gramatiky typu 1 (viz. 8.3.) a 3 (viz. 8.3.) jsou nezkracující.

**Věta 8:** Ke každé gramatice  $G$ , existuje ekvivalentní gramatika  $G'$ , ve které jsou všechna pravidla obsahující terminální symboly ve tvaru  $A \rightarrow a$ , kde  $A \in N, a \in \Sigma$ .

**Důkaz 2:** Pro každý terminál  $a \in \Sigma$ , zavedeme pomocný neterminál  $N_a$  a pravidlo  $N_a \rightarrow a$ . Všechny výskyty terminálů ve výchozích pravidlech nahradíme příslušnými pomocnými neterminály, tedy:

- $Bcb \rightarrow bBc$  se změní na  $BN_cN_b \rightarrow N_bBN_c$
- Pomocná pravidla jsou následně  $N_c \rightarrow c$  a  $N_b \rightarrow b$

**Věta 9:** Ke každé nezkracující gramatice existuje ekvivalentní gramatika, která je kontextově závislá<sup>6</sup>.

**Důkaz 3:** Předpokládejme, že  $G = \langle N, \Sigma, P, S \rangle$  je nezkracující gramatika. Pak můžeme předpokládat, že všechna pravidla jsou buď ve tvaru  $A \rightarrow a$  nebo ve tvaru obecně:  $A_1A_2 \cdots A_m \rightarrow B_1B_2 \cdots B_n$ , kde  $A_1, \dots, A_m, B_1, \dots, B_n \in N$  a navíc  $m \leq n$ .

Taková pravidla lze psát ve tvaru  $A_1A_2 \cdots A_m \rightarrow B_1B_2 \cdots B_my$ , kde  $y = B_{m+1} \cdots B_n$ . Budeme uvažovat nové pomocné neterminály  $X_1, \dots, X_m$ <sup>7</sup>. A zavedeme následující pravidla:

$$\begin{aligned} A_1A_2 \cdots A_m &\rightarrow X_1A_2 \cdots A_m \\ X_1A_2 \cdots A_m &\rightarrow X_1X_2A_3 \cdots A_m \\ &\vdots \\ X_1X_2 \cdots X_{m-1}A_m &\rightarrow X_1 \cdots X_{m-1}X_my \\ X_1X_2 \cdots X_my &\rightarrow B_1X_2X_3 \cdots X_my \\ &\vdots \\ B_1B_2 \cdots B_{m-1}X_my &\rightarrow B_1B_2 \cdots B_{m-1}B_my \end{aligned}$$

Tento postup se aplikuje pro všechna pravidla. Hledaná gramatika  $G'$  se skládá z  $\Sigma, N$  a navíc všech pomocných neterminálů a odvozovacích pravidel.

<sup>6</sup>Mějme na paměti, že říkáme-li, že je gramatika kontextově závislá, tak tím myslíme, že je kontextová.

<sup>7</sup>pro každé pravidlo se uvažují zvlášť

### 8.5. Základní vlastnosti bezkontextových gramatik

Pro bezkontextové gramatiky obecně platí:

1. Levé strany odvozovacích pravidel obsahují jediný neterminál<sup>8</sup>.
2. Odvozování nezávisí na kontextu.

**Věta 10:** Mějme bezkontextovou gramatiku  $G = \langle N, \Sigma, P, S \rangle$  a necht' navíc existuje  $X_1 \cdots X_k, \dots, z$  P-derivace délky  $n$ , kde  $X_1, \dots, X_k \in (N \cup \Sigma)$  a  $z \in (N \cup \Sigma)^*$  a potom pro každé  $i = 1, \dots, k$  existuje řetězec  $z_i \in (N \cup \Sigma)^*$  a P-derivace  $X_i, \dots, z_i$  délky  $n_i$  tak, že  $z = z_1 \cdot z_2 \cdots z_k$  a  $n = n_1 + n_2 + \cdots + n_k$

**Důkaz 4:** Větu 10 prokážeme indukcí přes délku výchozí derivace  $X_1 \cdots X_k, \dots, z$ .

1. Pro  $n = 0$  je situace triviální, protože  $z = X_1 \cdots X_k, z_i = X_i, n_i = 0$ . Každé  $X_i$  je derivace délky 0.
2. Necht' tvrzení platí pro libovolnou derivaci délky  $n$  a dokážeme, že  $X_1 \cdots X_k$  je P-derivace délky  $n + 1$ . Jelikož má uvažovaná P-derivace délku  $n + 1$ , lze ji psát ve tvaru:

$$\underbrace{X_1 \cdots X_k, \dots, y, z}_{\text{délka je } n}$$

Máme  $y \Rightarrow_G z$ . Můžeme aplikovat indukční předpoklad:

Existují řetězce  $y_1, \dots, y_k$  a P-derivace  $X_1, \dots, y_1$  až  $X_k, \dots, y_k$  délek  $n_1 \cdots n_k$  tak, že  $y = y_1 y_2 \cdots y_k$  a  $n = n_1 + n_2 + \cdots + n_k$ .

Z faktu, že  $y \Rightarrow_G z$  a z toho, že gramatika je bezkontextová plyne, že  $y$  je ve tvaru  $y = y'' y' A w' w''$  pro  $i = 1, \dots, k$ . Pak  $z$  je ve tvaru  $z = y'' y' u w' w''$  a  $A \rightarrow u \in P$ , to jest  $X_i, \dots, y_i, y' u w'$  je P-derivace délky  $n_{i+1}$ . Hledané derivace jsou:

$$\begin{array}{c} X_1, \dots, y_1 \\ \vdots \\ X_{i-1}, \dots, y_{i-1} \\ X_i, \dots, y_i y' u w' \\ X_{i+1}, \dots, y_{i+1} \\ \vdots \\ X_k, \dots, y_k \end{array}$$

**Příklad 15:**

$$\begin{aligned} N &= \{S\} \\ \Sigma &= \{a, b\} \\ P &= \{S \rightarrow SS|aSb|bSa|\varepsilon\} \end{aligned}$$

Posloupnost

$$SbSaS, SbSa, SbaSba, aSbbaSba, abbaSba$$

je P-derivací délky 4. Následně hledané P-derivace jsou:

<sup>8</sup>Toto tvrzení má praktický dopad. Jakmile se v odvozované větě objeví terminální symbol, tak již nemůže zmizet. Nemůže být například odvozen pravidlem, které na levé straně tento terminál obsahuje, protože takové pravidlo neexistuje.

1.  $S, aSb, ab$  (délka 2)
2.  $b$  (délka 0)
3.  $S, aSb$  (délka 1)
4.  $a$  (délka 0)
5.  $S, \varepsilon$  (délka 1)

**Poznámka 5:** U regulárních a kontextových gramatik jest hned vidět, zda  $\varepsilon \in L(G)$ .

Pro bezkontextovou gramatiku  $G = \langle N, \Sigma, P, S \rangle$  zavedeme následující podmnožiny:

$$\begin{aligned} E_0 &= \{A \in N \mid A \rightarrow \varepsilon \in P\} \\ E_{i+1} &= E_i \cup \{A \in N \mid A \rightarrow x, \text{ kde } x \in E_i^*\} \end{aligned}$$

**Příklad 16:**

$$\begin{aligned} A &\rightarrow \varepsilon \\ B &\rightarrow \varepsilon \\ F &\rightarrow ABBA \\ G &\rightarrow BAF \\ E_0 &= \{A, B\} \\ E_1 &= \{A, B, F\} \\ E_2 &= \{A, B, F, G\} \end{aligned}$$

$$E_i \subseteq N, E_N = \bigcup_{i=0}^{\infty} E_i$$

Jelikož je  $N$  konečná, musí platit:

$$\begin{aligned} E_0 \subseteq E_1 \subseteq E_2 \subseteq \dots \subseteq E_i = E_{i+1} &= E_{i+2} \\ E_N &= E_i \end{aligned}$$

**Věta 11:** Pro každou bezkontextovou gramatiku  $G = \langle N, \Sigma, P, S \rangle$  a pro příslušné  $E_N$  platí následující  $A \Rightarrow_G^* \varepsilon$  právě když  $A \in E_N$ . Speciálně  $\varepsilon \in L(G)$  právě když  $S \in E_N$ .

**Důkaz 5:** Dokazujeme obě implikace:

1. Pokud  $A \Rightarrow_G^* \varepsilon$ , pak prokážeme indukci přes délku P-derivace, tj. triviální případ je  $A \Rightarrow_G \varepsilon$ , tj. existuje pravidlo  $A \rightarrow \varepsilon \in P$  tj.  $A \in E_0$ .

Předpokládejme, že tvrzení platí pro všechny P-derivace délky  $n$ .

Mějme  $A, \dots, \varepsilon$  P-derivace délky  $n + 1$ . Použitím předchozí věty  $(A, X_1 \dots X_k, \dots, \varepsilon) A, X_i \dots X_n, \dots, \varepsilon$ . Tzn. existují derivace  $X_i, \dots, \varepsilon$  délek nejvýše  $n$ . Z předpokladu  $X_i \in E_n$ , pro každé  $i$  tj.  $i \in E_N$ .

2. Opačnou implikaci bychom dokázali tak, že pro každé  $E_i$  platí, pokud  $E \in E_i$  a tedy pak  $A \Rightarrow_G^* \varepsilon$ . Pro  $E_0$  zřejmé.  $A \rightarrow X_0 \dots X_k, A \in E_j$ .

**Věta 12:** Pro každou bezkontextovou gramatiku  $G$  existuje bezkontextová gramatika  $G'$ , neobsahující  $\varepsilon$  - pravidla tak, že  $L(G) \setminus \{\varepsilon\} = L(G')$ .

**Důkaz 6:** Mějme gramatiku  $G = \langle N, \Sigma, P, S \rangle$ .

Stanovme množinu  $E_n$  dle předchozího postupu a pak stanovme gramatiku

$$\begin{aligned} G' &= \langle N, \Sigma, P', S \rangle \\ P' &= \{A \rightarrow y \mid A \rightarrow x \in P \text{ a } y \in D_{(x)}\} \end{aligned}$$

Kde  $D_{(x)}$  značí množinu řetězců, které jsou neprázdné a vznikly z řetězce  $x$  vynecháním libovolného množství neterminálů z  $E_N$ .

**Příklad 17:** Mějme množinu  $E_n = \{A, B\}$ .

Následně pravidla:

$$\begin{aligned} X &\rightarrow aAbAB \\ &\dots \\ X &\rightarrow aAbAB \\ X &\rightarrow abAB \\ X &\rightarrow aAbB \\ X &\rightarrow aAbB \\ X &\rightarrow aAbA \\ X &\rightarrow abB \\ X &\rightarrow abA \\ X &\rightarrow aAb \\ X &\rightarrow ab \end{aligned}$$

**Věta 13:** Pro každou bezkontextovou gramatiku existuje ekvivalentní bezkontextová gramatika, která je navíc kontextová (a tudíž nezkracující).

**Důkaz 7:** Mějme gramatiku  $G$ .

Dle předchozí věty existuje  $G'$  tak, že  $L(G) \setminus \{\varepsilon\} = L(G')$ .  $G'$  je nezkracující a kontextová, protože nemá  $\varepsilon$  - pravidla. Pokud  $\varepsilon$  nepatří do  $L(G)$ , pak jsme hotovi. Pokud  $\varepsilon \in L(G)$ . Pak  $G'$  rozšíříme tak, že přidáme startovní neterminální symbol  $S'$  a pravidlo  $S' \rightarrow \varepsilon$  a  $S' \rightarrow S$ .

## 8.6. Jazyky bez gramatického základu

**Věta 14:** Označme třídu všech jazyků nad abecedou  $\Sigma$  jako  $2^{\Sigma^*}$ . Následně o takovéto třídě můžeme prohlásit, že je nespočetná.

**Důkaz 8:** Víme, že Kleeneho uzávěr  $\Sigma^*$  je spočetná množina, lze tedy psát, že  $\Sigma^* = \{x_1, x_2, \dots, x_5, \dots\}$ .

Sporem nechť je množina  $2^{\Sigma^*}$  spočetná, lze tedy napsat, že  $2^{\Sigma^*} = \{L_1, L_2, \dots\}$  a  $L = \{x_i \mid x_i \notin L_i\}$ .

Kdyby pro nějaké  $j$  platilo, že  $L = L_j$ , pak  $x_j \in L \iff x_j \in L_j$ , což je spor.

**Věta 15:** Nad abecedou obsahující jediný symbol  $\{a\} \exists$  jazyk jež není typu 0. (Nelze jej generovat žádnou formální gramatikou.)

**Důkaz 9:** Mějme abecedu  $\Gamma = \{, , \rightarrow, \{, \}, a, n, \#\}$ .<sup>9</sup> Pomocí  $\Gamma$  jsme schopni kódovat jazyk pravidel gramatiky. (Neterminály můžeme kódovat jako  $n, nn, \dots$ ).

$\Gamma^*$  je spočetná tedy existuje spočetně mnoho gramatik.

$2^{\Sigma^*}$  je nespočetná, tedy musí existovat jazyk, které nelze generovat gramatikou z  $\Gamma^*$ .

## 9. Determinismus versus nedeterminismus

Za *determinismus* považujeme takovou konfiguraci, pro kterou platí, že je v **každém jejím kroku jasné**, co bude následovat. Naopak u *nedeterministických* konfigurací **není v určitých případech možné další krok přesně vyjádřit** na základě znalostí aktuálního kroku.

Automaty realizují tzv. analytický formalismus a naproti tomu gramatiky realizují tzv. generativní formalismus.

## 10. Konečné deterministické automaty

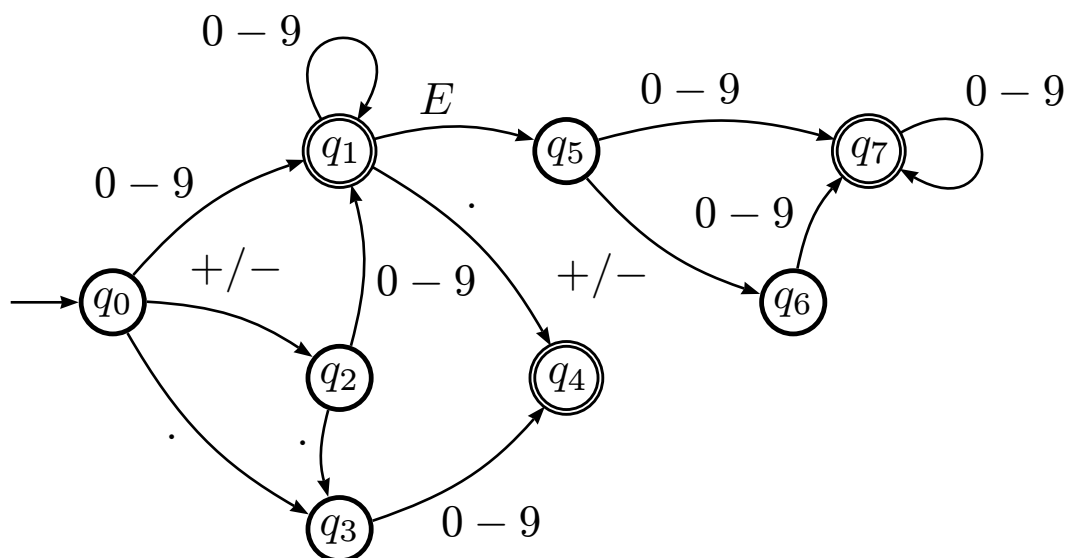
Jak již bylo řečeno, tak KDA realizují analytický (resp. výpočetní) formalismus a představují jakési „jednoduché počítače“. KDA a jejich činnost jsou charakterizovány několika prvky:

1. Vstupem jsou řetězce nad určitou abecedou, která se obvykle značí  $\Sigma$ .
2. Řídící jednotka automatu se skládá z konečně mnoha stavů (aby byla reprezentovatelná v počítačích).
3. Počátek činnosti automatu představuje okamžik, kdy na vstupu automatu je celý vstupní řetězec a řídící jednotka je v iniciálním stavu.
4. Automat vykonává svou primární činnost tím, že přečte ze vstupu aktuální symbol a na základě stavu, ve kterém se právě nachází se řídící jednotka přepne do jiného stavu a odebere onen přečtený symbol ze vstupního řetězce.
5. Konec činnosti automatu nastává v případě, že na vstupu již není co číst. Byl tedy zpracován celý vstupní řetězec. V tomto okamžiku automat nachází v určitém stavu, podle typu stavu následně řekneme, že automat řetězec přijímá nebo zamítá. Existují tedy přijímací (neboli koncové) stavy a nepřijímací (neboli nekoncové) stavy.

**Příklad 18:** Nyní si ukažme, jak vypadá typický KDA.

Dvojitě kroužkovaný stav označuje stav koncový. Šipky mezi jednotlivými stavy označují přechody, přičemž popisek u jednotlivých přechodů indikují symbol, který se čte, pokud je tento přechod využit. Automat na obrázku 3. by přijímal například slovo  $13.6E + 2$ , obecně jakékoliv číslo kodované v tradiční fixní a pohyblivé notaci s desetinnou tečkou.

<sup>9</sup>Všechny prvky množiny jsou symboly, včetně symbolu „čárka“



Obrázek 3. Názorný příklad KDA

**Definice 7:** Konečný deterministický automat s úplnou přechodovou funkcí je struktura:

$$A = \langle \Sigma, Q, \delta, q_0, F \rangle$$

Nyní si popíšeme jednotlivé členy automatu:

- $\Sigma$  = vstupní abeceda
- $Q$  = konečná množina stavů
- $q_0 \in Q$  = počáteční stav
- $F \subseteq Q$  = množina koncových stavů
- $\delta$  = zobrazení  $\delta: Q \times \Sigma \rightarrow Q$ , neboli přechodová funkce

U přechodové funkce  $\delta$  uvažujeme zápis například  $\delta(r, a) = q$ , který čteme: „Při vstupním symbolu  $a \in \Sigma$  a aktuálním stavu  $r$  přejde automat do nového stavu  $q$ .“

Předpokládáme, že  $Q$  je konečná a  $\delta$  je zobrazení.

**Příklad 19:** Automat s nadefinovanými přechody dle  $\delta$  by mohl vypadat například takto:

$$\begin{aligned}
 A &= \langle \Sigma, Q, \delta, q_0, F \rangle \\
 \Sigma &= \{a, b, c\} \\
 Q &= \{q_0, q_1, q_2, q_3, q_4\} \\
 F &= \{q_4\} \\
 \delta &= \{ \langle q_0, a, q_0 \rangle, \langle q_0, b, q_0 \rangle, \langle q_0, c, q_1 \rangle, \\
 &\quad \langle q_1, a, q_2 \rangle, \langle q_1, b, q_0 \rangle, \langle q_1, c, q_1 \rangle, \\
 &\quad \langle q_2, a, q_0 \rangle, \langle q_2, b, q_3 \rangle, \langle q_2, c, q_1 \rangle, \\
 &\quad \langle q_3, a, q_2 \rangle, \langle q_4, b, q_0 \rangle, \langle q_3, c, q_4 \rangle, \\
 &\quad \langle q_4, a, q_4 \rangle, \langle q_4, b, q_4 \rangle, \langle q_4, c, q_4 \rangle \}
 \end{aligned}$$

## 10.1. Reprezentace KDA

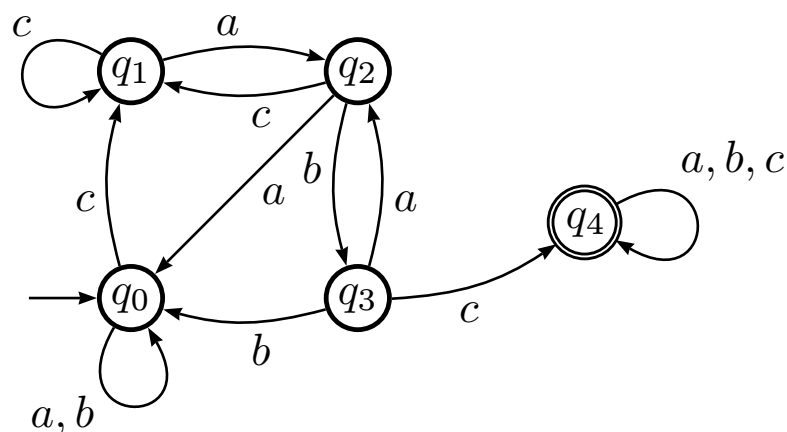
### 10.1.1. Reprezentace přechodovou tabulkou

Řádky tabulky představují stavy a sloupce jednotlivé symboly vstupní abecedy. Znak \* označuje, že stav je koncový.

	a	b	c
$q_0$	$q_0$	$q_0$	$q_1$
$q_1$	$q_2$	$q_0$	$q_1$
$q_2$	$q_0$	$q_3$	$q_1$
$q_3$	$q_2$	$q_0$	$q_4$
$q_4^*$	$q_4$	$q_4$	$q_4$

Tabulka 1. Přechodová tabulka pro KDA

### 10.1.2. Reprezentace přechodovým diagramem



Obrázek 4. Přechodový diagram pro KDA

## 10.2. Konfigurace a výpočet KDA

Konfigurace jednoznačně určuje aktuální fázi výpočtu a jsou to dvojice  $\langle q, w \rangle$ ,  $q \in Q$ ,  $w \in \Sigma^*$ , následně samotná konfigurace  $Q \times \Sigma^*$ .

Počáteční konfigurace zahrnuje obvykle počáteční stav  $q_0$ , tedy například  $\langle q_0, w \rangle$ .

Koncová konfigurace je konfigurací, kdy jsme přečetli celý vstupní řetězec, tedy  $\langle q, \varepsilon \rangle$ . Nicméně je třeba rozlišit dva stavy:

1. Koncová přijímací konfigurace, pro případ, kdy  $q \in F$ .
2. Koncová zamítací konfigurace, pro případ, kdy  $q \notin F$ .

**Definice 8:** Mějme automat  $A = \langle \Sigma, Q, \delta, q_0, F \rangle$  a řetězec  $w \in \Sigma^*$ . Pak posloupnost konfigurací  $r_i, w_i$  pro  $i \in \{0, 1, \dots, n\}$  splňující podmínky:

1.  $r_0$  je počáteční stav automatu  $A$ .
2.  $w_0 = w$
3.  $w_n = \varepsilon$
4.  $w_i = a_i w_{i+1}$  a  $\delta(r_i, a_i) = r_{i+1}$  pro každé  $i \in \{0, 1, \dots, n-1\}$

nazveme **výpočet automatu  $A$  délky  $n$  pro řetězec  $w$** .

Popíšme si nyní jeden krok výpočtu, tj. obecně:

$$\langle r_i, w_i \rangle = \langle r_{i+1}, w_{i+1} \rangle$$

tj.  $w_{i+1}$  vzniklo z  $w_i$  odebráním prvního symbolu, který si označme  $a_i$ . Do stavu  $r_{i+1}$  jsme se dostali ze stavu  $r_i$  při vstupním symbolu  $a_i$ .



**Příklad 20:** Typický automatový výpočet vypadá například takto:

$$\begin{aligned}
&\langle q_0; accbcabca \rangle \\
&\langle q_0; ccbcabca \rangle \\
&\langle q_1; cbcabca \rangle \\
&\langle q_1; bcabca \rangle \\
&\langle q_0; cabca \rangle \\
&\langle q_1; abca \rangle \\
&\langle q_2; bca \rangle \\
&\langle q_3; ca \rangle \\
&\langle q_4; c \rangle \\
&\langle q_4; \varepsilon \rangle
\end{aligned}$$

**Věta 16:** KDA  $A$  má pro řetězec délky  $n$  jednoznačný výpočet rovněž délky  $n$ .

### 10.3. Rozšířená přechodová funkce pro KDA

Rozšířená přechodová funkce má rekurzivní předpis:

$$\begin{aligned}
&\delta^* : Q \times \Sigma^* \rightarrow Q \\
&\delta^*(q, u) = \begin{cases} q & \text{pokud } u = \varepsilon \\ \delta^*(\delta(q, a), v) & \text{pokud } u = av, a \in \Sigma, \end{cases}
\end{aligned}$$

**Věta 17:** Pro každé  $u, v \in \Sigma^*$  platí, že  $\delta^*(q, uv) = \delta^*(\delta^*(q, u), v)$

**Důkaz 10:** Větu 17 dokážeme indukcí přes délku řetězce následujícím způsobem:

1. Pokud  $|u| = 0$ , pak  $\delta^*(\delta^*(q, u), v) = \delta^*(\delta^*(q, \varepsilon), v) = \delta^*(q, v) = \delta^*(q, \varepsilon v) = \delta^*(q, uv)$
2. Pokud  $|u| > 0$ , pak předpokládáme, že tvrzení platí pro  $\forall$  řetězce kratší délky. Tj. pro  $u = aw, a \in \Sigma, w \in \Sigma^*$  máme:

$$\begin{aligned}
\delta^*(\delta^*(q, u), v) &= \delta^*(\delta^*(q, aw), v) \\
&= \delta^*(\delta^*(\delta(q, a), w), v) \\
&= \delta^*(\delta(q, a), wv) \\
&= \delta^*(q, awv) \\
&= \delta^*(q, uv)
\end{aligned}$$

**Poznámka 6:** Automat  $A$  přijím řetězec  $a$ , pokud  $\delta^*(q_0, w) \in F$ .

**Věta 18:** Řetězec  $w$  je přijat automatem, právě když existuje přijímací výpočet pro  $w$ .

**Důkaz 11:** Stačí dokázat, že  $\delta^*(q_0, w) \in F$  každý výpočet  $\langle r_0, w_0 \rangle \cdots \langle r_n, w_n \rangle$  je přijímající, tj. když  $r_n \in F$ . Pro každé  $\langle r_i, w_i \rangle$  prokážeme, že  $\delta^*(r_i, w_i) = r_n$ , což dokážeme indukcí přes  $i$ .

1. Pro  $i = n$  mějme  $\delta^*(r_n, w_n) = \delta^*(r_n, \varepsilon) = r_n$ . Pak lze předpokládat, že máme nějaké  $i$  a pro každé  $j$  takové, že  $i < j \leq n$  tvrzení platí.
2. Dle bodu 4 z definice výpočtu platí, že  $w_i = a_i w_{i+1}$  a  $\delta(r_i, a_i) = r_{i+1}$ . Tj.  $\delta^*(r_i, w_i) = \delta^*(r_i, a_i w_{i+1}) = \delta^*(\delta(r_i, a_i), w_{i+1}) = \delta^*(r_{i+1}, w_{i+1}) = r_n$

Jazyk rozpoznatý konečným deterministickým automate lze formulovat jako  $L(A) = \{w \in \Sigma^* \mid \delta^*(q_0, w) \in F\}$ .

#### 10.4. KDA s neúplnou přechodovou funkcí

Mějme KDA  $A = \langle \Sigma, Q, \delta, q_0, F \rangle$  a upřesněme  $\delta$ , které se liší.

$\delta$  je neúplná přechodová funkce tvaru  $\delta \subseteq Q \times \Sigma \times Q$ , přičemž také platí, že pokud  $\langle q, a, r_1 \rangle \in \delta$  a  $\langle q, a, r_2 \rangle \in \delta$ , pak  $r_1 = r_2$  a  $\delta : Q \times \Sigma \rightarrow Q$ .

Z toho plynou dvě situace:

1.  $\delta(q, a)$  je definována,  $\langle q, a, r \rangle \in \delta$  pro nějaké  $r$  takové, že  $\delta(q, a) = r$ .
2.  $\delta(q, a)$  není definováno, pokud  $\langle q, a, r \rangle \notin \delta$  pro žádná  $r$ .

V grafu se toto projeví tak, že ze stavu  $q_i$  nevede žádná hrana pro symbol  $a_i$ .

Výpočet se liší v bodu 4  $w_i = a_i w_{i+1}$  a  $\delta(r_i, a_i)$  je definována a  $\delta(r_i, a_i) = r_{i+1}$  pro každé  $i \in \{0, 1, \dots, n\}$ .

Řetězec  $w$  je přijat automatem  $A$ , právě tehdy, když existuje výpočet  $A$  pro  $w$ , který je přijímající.

Následně jazyk rozpoznatý automatem  $A$  je  $L(A) = \{w \in \Sigma^* \mid \delta^*(q_0, w) \in F\}$ .

**Věta 19:** Ke každému KDA s neúplnou přechodovou funkcí  $\delta$  existuje KDA s úplnou přechodovou funkcí  $\delta$ , který rozpoznává stejný jazyk.

**Důkaz 12:** Každý KDA s úplnou přechodovou funkcí je zároveň KDA s neúplnou přechodovou funkcí. Tedy  $A' = \langle \Sigma, Q \cup \{\#\}, \delta', q_0, F \rangle$

$$\delta'(q, a) = \begin{cases} \delta(q, a) & \text{pokud } \delta(q, a) \text{ je definováno} \\ \# & \text{pokud } q = \#, \forall a \in \Sigma \\ \# & \text{pokud } \delta(q, a) \text{ není definováno} \end{cases}$$

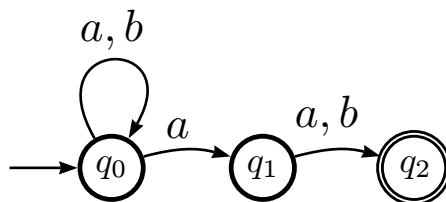
Následně tedy  $L(A) = L(A')$ .

#### 10.5. Implementace KDA

1.  $\delta$  jako dvourozměrné pole - „dábelsky rychlé“, paměťově náročné
2.  $\delta$  jako seznam/hashovací tabulka - vhodné když  $|\delta| \ll \ll |Q \times \Sigma|$
3.  $\delta$  jako orientovaný graf

## 11. Konečné nedeterministické automaty

**Příklad 21:** Mějme automat:



Vstupní řetězce: *abba* (nepřijat), *baba* (nepřijat), *baab* (přijat), *bbaa* (přijat).

V případě řetězce *baab* máme dokonce 3 možnosti výpočtu:

1.  $\langle q_0, baab \rangle, \langle q_0, aab \rangle, \langle q_0, ab \rangle, \langle q_0, b \rangle, \langle q_0, \varepsilon \rangle$  – končí neúspěchem.
2.  $\langle q_0, baab \rangle, \langle q_0, aab \rangle, \langle q_1, ab \rangle, \langle q_2, b \rangle$  – končí neúspěchem.
3.  $\langle q_0, baab \rangle, \langle q_0, aab \rangle, \langle q_0, ab \rangle, \langle q_1, b \rangle, \langle q_2, \varepsilon \rangle$  – končí úspěchem.

Předchozí zápisy můžeme pojmenovat také jako „nedeterministický výpočet.“

Jiným zápisem téhož může být také ten následující.

$$\langle \{q_0\}, baab \rangle, \langle \{q_0\}, aab \rangle, \langle \{q_0, q_1\}, ab \rangle, \langle \{q_0, q_1, q_2\}, b \rangle, \langle \{q_0, q_2\}, \varepsilon \rangle$$

**Definice 9:** Strukturu  $A = \langle \Sigma, Q, \delta, I, F \rangle$  nazvěme **konečným nedeterministickým automatem** nad abecedou  $\Sigma$ . Pro tuto strukturu následně platí tato tvrzení:

- $\Sigma, Q$  a  $F$  jsou stejné jako u konečného deterministického automatu.
- $I \subseteq Q$  označuje množinu počátečních stavů, která by měla být obecně neprázdná.
- $\delta$  označuje přechodovou funkci ve tvaru  $\delta : Q \times \Sigma \rightarrow 2^Q$ , tedy  $\delta(q, a) = \{r_1, \dots, r_k\}$ . Totéž slovně: „Automat může při stavu  $q$  při symbolu  $a$  přejít do kteréhokoliv stavu z  $\{r_1, \dots, r_k\}$ .“

**Příklad 22:**

$$\begin{aligned} \Sigma &= \{a, b\} \\ P &= \{q_0, q_1, q_2, q_3\} \\ I &= \{q_0, q_3\} \\ F &= \{q_2\} \end{aligned}$$

Následně přechodová funkce:

$$\begin{aligned} \delta = \{ & \langle q_0, a, \{q_0, q_1\} \rangle, \langle q_0, b, \{q_0\} \rangle, \langle q_1, a, \{q_2\} \rangle, \langle q_1, b, \{q_2\} \rangle, \\ & \langle q_2, a, \emptyset \rangle, \langle q_2, b, \emptyset \rangle, \langle q_3, a, \emptyset \rangle, \langle q_3, b, \{q_1\} \rangle \} \end{aligned}$$

### 11.1. Reprezentace KNA

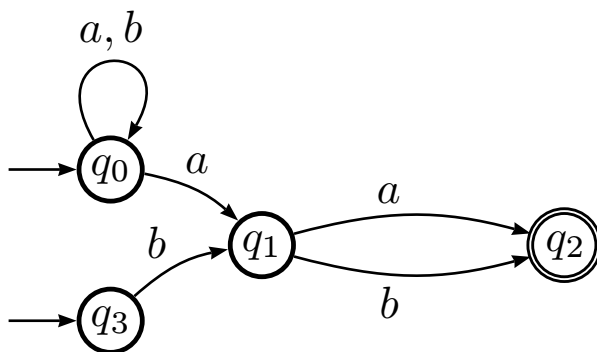
Předchozí příklad číslo 22 lze reprezentovat několika způsoby:

1. **Přechodová tabulka**, která ve svém těle obsahuje množiny stavů.

	a	b
$\rightarrow q_0$	$\{q_0, q_1\}$	$\{q_0\}$
$q_1$	$\{q_2\}$	$\{q_2\}$
$q_2^*$	$\emptyset$	$\emptyset$
$\rightarrow q_3$	$\emptyset$	$\{q_1\}$

Tabulka 2. Přechodová tabulka s množinami stavů

2. **Diagram**, který automat demonstruje v grafické podobě.



### 11.2. Nedeterministický výpočet

Nyní si popíšme **nedeterministický výpočet**, který je definován následujícími věcmi:

- Konfigurace, což je dvojice ve tvaru  $\langle stav, řetězec \rangle$ .
- Počáteční konfigurace ve tvaru  $\langle q, w \rangle$  kde  $q \in I$ .
- Koncová konfigurace ve tvaru  $\langle q, \varepsilon \rangle$ .
- Koncová přijímací konfigurace  $\langle q, \varepsilon \rangle$  kde  $q \in F$ .

**Definice 10:** Mějme  $A = \langle \Sigma, Q, \delta, I, F \rangle$  a  $w \in \Sigma^*$ . Pak posloupnost konfigurací  $\langle r_i, w_i \rangle$  pro  $i = \{0, \dots, n\}$  splňující podmínky:

$$r_0 \in I \quad (1)$$

$$w_0 = w \quad (2)$$

$$w_n = \varepsilon \quad (3)$$

$$w_i = a_i w_{i+1} \text{ a } r_{i+1} \in \delta(r_i, a_i) \text{ pro } i = \{0, \dots, n-1\} \quad (4)$$

nazveme **nedeterministický výpočet**.

### 11.3. Rozšířená přechodová funkce

**Definice 11:** Rozšířená přechodová funkce má tvar:

$$\delta^* : 2^Q \times \Sigma^* \rightarrow 2^Q$$

$$\delta^*(R, w) = \begin{cases} R & \text{pokud } w = \varepsilon \\ \delta^*(\bigcup_{q \in R} \delta(q, a), u) & \text{pokud } w = au, \text{ kde } a \in \Sigma, u \in \Sigma^* \end{cases}$$

**Věta 20:** Platí  $\delta^*(R, w) = \delta^*(\delta^*(R, u), v), \forall R \subseteq Q, uv \in \Sigma^*$ .

**Důkaz 13:** Předchozí tvrzení dokazujeme indukcí přes délku řetězce.

1. Pro  $u = \varepsilon$  je situace triviální.
2. Pokud  $u = ay, |y| < |u|$ , pak  $\delta^*(R, w) = \delta^*(R, ayv) = \delta^*(R, a(yv))$ .
3. Nyní aplikujeme definici.

$$\begin{aligned} \delta^*(\bigcup_{q \in R} \delta(q, a), yv) &= \text{indukční předpoklad} \\ \delta^*(\delta^*(\bigcup_{q \in R} \delta(q, a), y), v) &= \text{definice } \delta^* \\ \delta^*(\delta^*(R, ay), v) &= \delta^*(\delta^*(R, u), v) \end{aligned}$$

**Věta 21:** Platí následující tvrzení:

$$\delta^*(\bigcup_{i=1}^k R_i, w) = \bigcup_{i=1}^k \delta^*(R_i, w) \text{ pro každé } R_i \subseteq Q, w \in \Sigma^*$$

**Důkaz 14:** Předchozí tvrzení dokazujeme indukcí přes délku řetězce  $w$ .

$$\begin{aligned} \delta^*(\bigcup_{i=1}^k R_i, w) &= \delta^*(\bigcup_{i=1}^k R_i, au) = \delta^*(q \in \bigcup_{i=1}^k \delta(q, a), u) \\ \delta^*(\bigcup_{i=1}^k \bigcup_{q \in R_i} \delta(q, a), u) &\dots \text{indukční předpoklad} \\ \bigcup_{i=1}^k \delta^*(\bigcup_{q \in R_i} \delta(q, a), u) & \\ \bigcup_{i=1}^k \delta^*(R, a_n) &= \bigcup_{q \in R_i} \delta(R, w) \end{aligned}$$

### 11.4. Řetězce přijímané KNA

KNA  $A$  přijímá řetězec  $w$ , pokud  $\delta^*(I, w) \cap F \neq \emptyset$ . Navíc jazyk, přijímaný KNA  $A$  si definujeme jako  $L(A) = \{w \in \Sigma^* \mid \delta^*(I, w) \cap F \neq \emptyset\}$ .

**Věta 22:** Platí, že  $w \in L(A)$  právě tehdy, když KNA  $A$  má přijímací výpočet pro  $w$ .

**Důkaz 15:** Předchozí tvrzení lze dokázat indukcí přes délku řetězce  $w$ .

$q \in \delta^*(I, w)$  právě tehdy, když existuje výpočet pro  $w$ , končící ve stavu  $q$   
dekompozice navíc  $w = ua$

### 11.5. Determinizace KNA

**Věta 23:** Pro každý KDA  $A = \langle \Sigma, Q, \delta, q_0, F \rangle \exists$  KNA  $A'$  tak, že  $L(A) = L(A')$ .

**Důkaz 16:** Pro výchozí  $A$  uvažujme  $A' = \langle \Sigma, Q, \delta', \{q_0\}, F \rangle$ , pak  $\delta'(q, a) = \{\delta(q, a)\}$ . Zbytek důkazu je zřejmý.

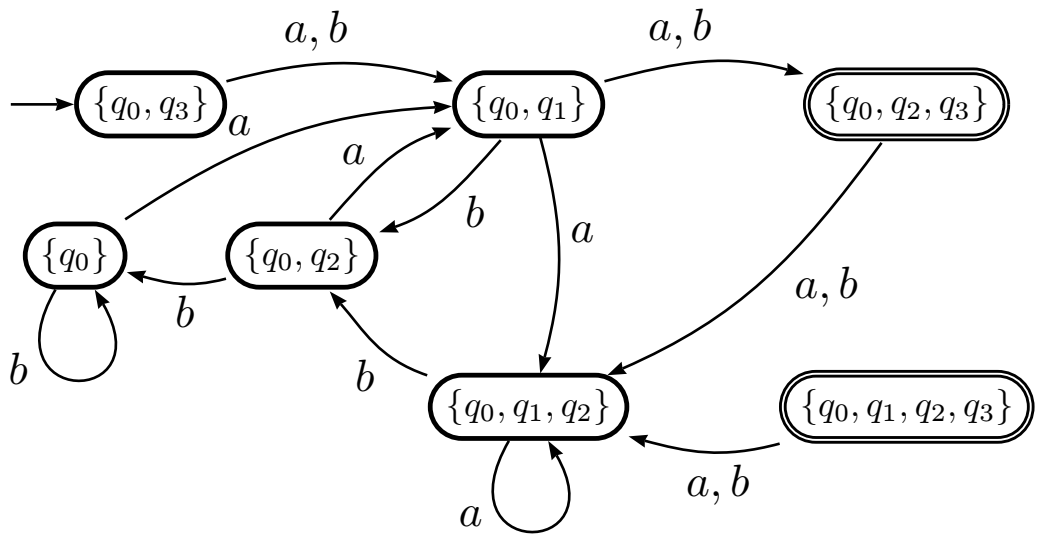
**Věta 24:** Pro každý KNA  $A$  existuje KDA  $A^D$  tak, že  $L(A) = L(A^D)$ .

**Důkaz 17:** Předchozí větu lze dokázat následujícím způsobem:

1. Uvažujme  $A^D = \langle \Sigma, 2^Q, \delta^D, I, F^D \rangle$ , kde  $F^D = \{R \subseteq 2^Q \mid R \cap F \neq \emptyset\}$ ,  $\delta^D(R, a) = \delta^*(R, a)$ . Nyní zbývá ukázat, že  $\delta^*(I, w) \cap F \neq \emptyset$  právě tehdy, když  $(\delta^D)^*(I, w) \in F^D$ , což prokážeme indukcí přes délku řetězce  $w$ .
2. Pro  $w = \varepsilon$  je situace zřejmá. Jinak  $(\delta^D)^*(R, w) = (\delta^D)^*(R, \varepsilon) = R = \delta^*(R, \varepsilon) = \delta^*(R, w)$ .
3. Předpokládejme, že tvrzení platí pro řetězce délky  $n$  a necht'  $w$  má délku  $n+1$  a  $w = au$  pro  $a \in \Sigma, |u| < |w|$ . Pak:

$$\begin{aligned} (\delta^D)^*(R, w) &= (\delta^D)^*(R, au) = (\delta^D)^*(\delta^D(R, a), u) = \\ &= \delta^*(\delta^D(R, a)u) = \delta^*(\delta^*(R, a), u) = \delta^*(R, au) = \delta^*(R, w) \end{aligned}$$

**Příklad 23:** Vemme KNA z příkladu 22.



Ještě jeden automat, nepřečtu to dobře ze sešitu. :)

### 11.6. Algoritmus pro převod KNA na KDA

Nyní si ukažeme pseudokód algoritmu pro převod konečných nedeterministických automatů na konečné deterministické automaty, pro které platí, že akceptují řetězce stejného jazyka.

```

 $\delta^*D \leftarrow \emptyset; Q^*D \leftarrow \emptyset; F^*D \leftarrow \emptyset; w \leftarrow 1$ 
while  $w \neq Q$  do
  select  $R \in w$ 
     $w \leftarrow w \setminus R; Q^*D \leftarrow Q^*D \cup R$ 
    if  $R \cap F \neq \emptyset$  then
       $F^*D \leftarrow F^*D \cup R$ 
    endif
    foreach  $a \in \Sigma$  do
       $v \leftarrow \delta^*(R, a)$ 
      if  $N \neq \emptyset$  then
        if  $N \not\subseteq w \cup Q^*D$  then
           $w \leftarrow w \cup N$ 
        endif
         $\delta^*D \leftarrow \delta^*D \cup \langle R, a, N \rangle$ 
      endif
    end
  end
end
return  $\langle \Sigma, Q^*D, \delta^*D, I, F^*D \rangle$ 

```

Obrázek 5. Pseudokód pro převod KNA na KDA.

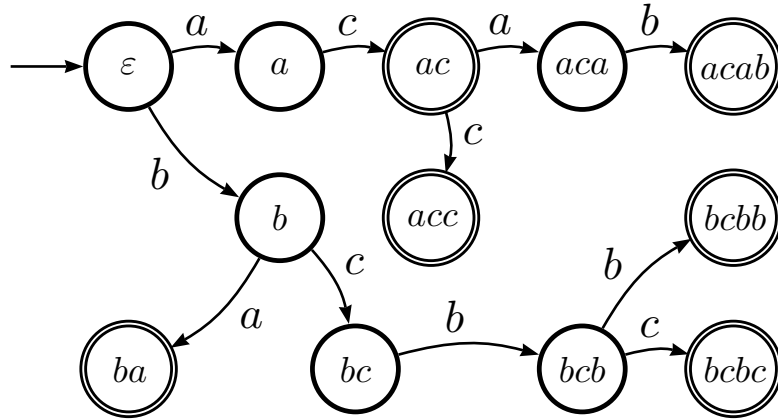
**Definice 12:** *Trie* je prefixový strom, který umožňuje „rychlé hledání ve slovníku.“

**Definice 13:** Jako *slovník* označujeme konečný neprázdný jazyk který neobsahuje  $\varepsilon$ .

**Definice 14:** Trie slovníku  $L$  je KDA  $T_L = \langle \Sigma, Q, \delta, \varepsilon, F \rangle$ , přičemž:

$$\begin{aligned}
 Q &= \text{množina prefixů všech řetězců} \\
 Q &= \bigcup_{w \in L} Pfx(w) \\
 F &= L^{w \in L} \\
 \delta(w, a) &= wa, \text{ pokud } wa \in Q, \text{ jinak } \delta(w, a) \text{ není definováno}
 \end{aligned}$$

**Příklad 24:** Uvedme si příklad konečného slovníkového automatu  $D_L$ , který je pochopitelně deterministický:



## 12. Vztah regulárních jazyků a konečných automatů

### 12.1. Regulární jazyky jsou rozpoznatelné KDA (implikace zleva)

**Věta 25:** Pro každou regulární gramatiku  $G = \langle N, \Sigma, P, S \rangle$  existuje konečný deterministický automat  $A$  tak, že jazyk generovaný gramatikou je totéž, jako jazyk rozpoznatelný automatem, tj.  $L(G) = L(A)$

**Důkaz 18:** Nejprve uvažujeme situaci, že  $\varepsilon \notin L(G)$ . Uvažujme konečný nedeterministický automat  $A = \langle \Sigma, N \cup \{\#\}, \delta, \{S\}, \{\#\} \rangle$ .

$$\delta(A, b) = \begin{cases} \{B \in N \mid A \rightarrow bB \in P\} & \text{pokud } A \in N \wedge A \rightarrow b \notin P \\ \{B \in N \mid A \rightarrow bB \in P\} \cup \{\#\} & \text{pokud } A \in N \wedge A \rightarrow b \in P \\ \emptyset & \text{jinak} \end{cases}$$

Pro důkaz  $L(G) = L(A)$  stačí prokázat, že pro každé  $A \in N$  a  $x \in \Sigma^*$  platí, že  $A \Rightarrow_G^* x$  právě když  $\# \in \delta^*(\{A\}, x)$ .

Důkaz provedeme indukcí přes délku řetězce  $x$ .

1. Pro  $|x| = 1$  zřejmé.  $A \Rightarrow_G^* x$  právě když  $A \Rightarrow_G x$ , tj. existuje pravidlo  $A \rightarrow x \in P$  tj. z definice  $\delta^*$  platí, že  $\# \in \delta^*(\{A\}, x)$ . Nechť  $|x| = n$  a nechť tvrzení platí pro všechny řetězce kratší délky. Jelikož gramatika  $G$  je regulární, má  $P$ -derivace  $A, \dots, x$  právě  $n$  kroků. Pokud  $|x| > 1$  pak  $A \Rightarrow_G bB \Rightarrow_G^* by = x$  pro nějaké  $A \rightarrow bB \in P$ .
2. Pro  $|y| < n$  z indukčního předpokladu platí, že  $\# \in \delta^*(\{B\}, y)$ . Tím spíš  $\delta^*(\{A\}, x) = \delta^*(\{A\}, by) = \delta^*(\delta(A, b), y) = \delta^*(\{B, \dots\}, y) \supseteq \delta^*(\{B\}, y)$  tj.  $\# \in \delta^*(\{A\}, \#)$  protože  $A \rightarrow bB \in P$  tj.  $B \in \delta(A, b)$ .
3. Tím jsme prokázali, že pokud  $A \Rightarrow_G^* x$  pak  $\# \in \delta^*(\{A\}, x)$ .
4. Obráceně, pokud  $\# \in \delta^*(\{A\}, x)$  pak pro  $x = by, b \in \Sigma$  máme:  $\# \in \delta^*(\{A\}, by) = \delta^*(\delta(A, b), y) = \delta^*(\bigcup_{B \in \delta(A, b)} \{B\}, y) = \bigcup_{B \in \delta(A, b)} \delta^*(\{B\}, y)$



Tj. existuje  $B \in \delta(A, b)$  tak, že  $\# \in \delta^*(\{B\}, y)$ . Ze zavedení  $\delta$  plyne, že  $A \rightarrow bB \in P$

5. Aplikací indukčního předpokladu, existuje P-derivace  $B, \dots, y$ . Hledaná P-derivace je ve tvaru:  $A, bB, \dots, by = x$ , tj.  $A \Rightarrow_G^* x$ .

V případě, že  $\varepsilon \in L(G)$ , rozšíříme automat následovně, jednou ze tří možností:

1. Přidáme  $S$  do množiny koncových stavů.
2. Přidáme  $\#$  mezi počáteční stavy.<sup>10</sup>
3. Zavedeme nový stav, který bude počáteční a zároveň koncový a nevedou z něj žádné přechody jinam.

**Poznámka 7:** Nyní zbývá automat pouze determinizovat.

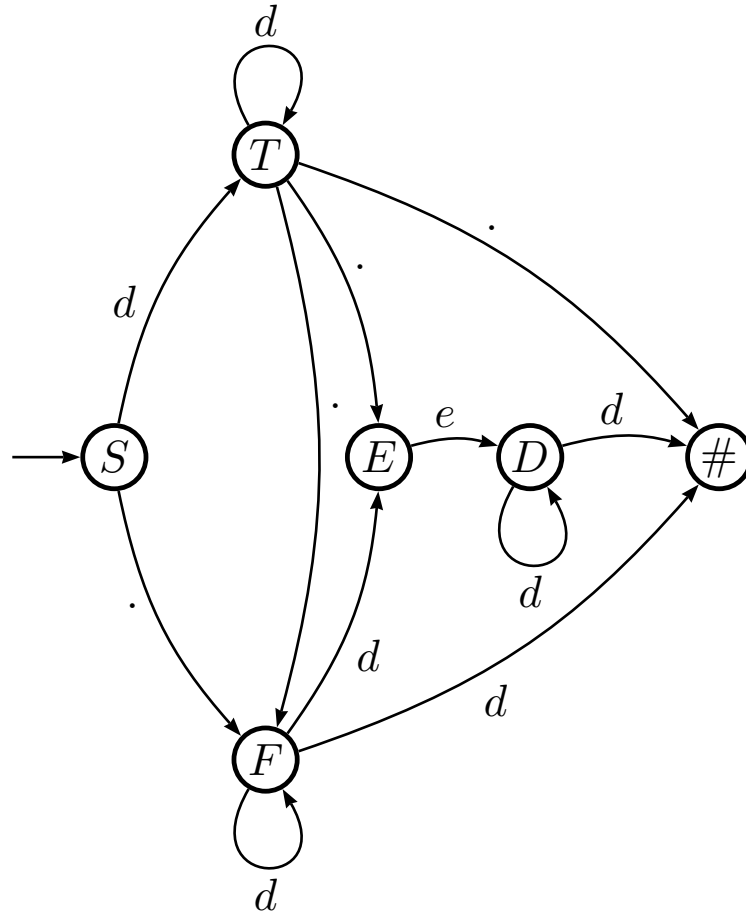
**Příklad 25:** Máme gramatiku  $G$ .

$$G = \langle N, \Sigma, P, S \rangle$$

$$\Sigma = \{e, d, \cdot\}$$

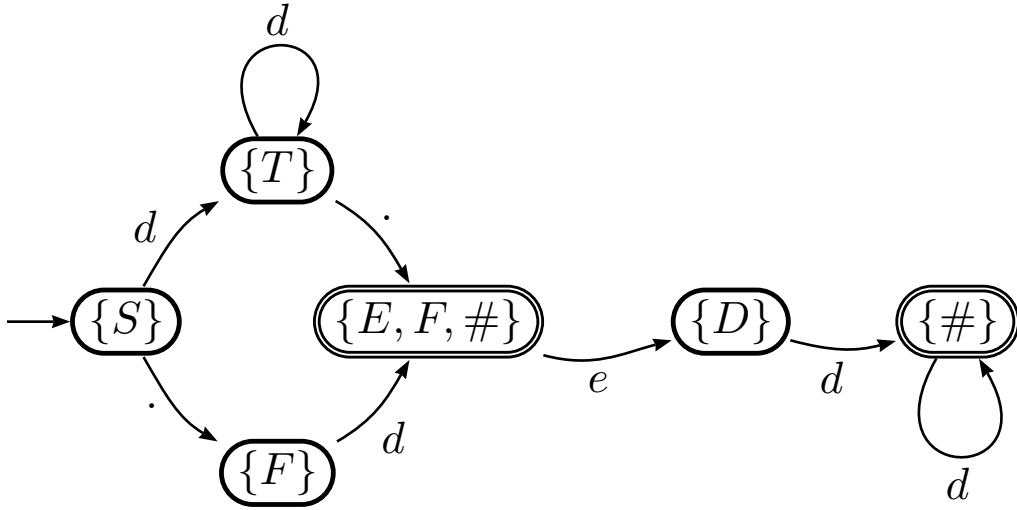
$$P = \{S \rightarrow \cdot F | dT, T \rightarrow \cdot E | \cdot F | dT | \cdot, D \rightarrow dD | d, E \rightarrow eD, F \rightarrow dE | dF | d\}$$

Automat rozpoznávající jazyk, generovaný gramatikou  $G$ , bude vypadat následovně:



<sup>10</sup>Těž nazývána jako „Konyho finta“.

Když tento automat zdeterminizujeme, dostaneme následující automat:



## 12.2. Jazyky rozpoznatelné KDA jsou regulární (implikace zprava)

**Věta 26:** Pro každý konečný deterministický automat  $A = \langle \Sigma, Q, \delta, q_0, F \rangle$  existuje regulární gramatika  $G$  tak, že  $L(A) = L(G)$ .

**Důkaz 19:** Za neterminální symboly  $G$  vezmeme stavy automatu. Startovní neterminál bude  $q_0$ . Uvažujeme gramatiku:  $G = \langle Q, \Sigma, P, q_0 \rangle$

$$P = \{q \rightarrow ar \mid \text{pokud } \delta(q, a) = r, \text{ pro } q, r \in Q \text{ a } a \in \Sigma\} \\ \cup \{q \rightarrow a \mid \text{pokud } \delta(q, a) \in F\}$$

Prokážeme že:  $q \Rightarrow_G^* x$  právě když  $\delta^*(q, x) \in F$

Pro  $|x| = 1$  platí:  $q \Rightarrow_G^* x$  právě když existuje pravidlo  $q \rightarrow x \in P$ , tj. z definice  $P$  platí  $\delta(q, x) \in F$

Pro  $x = by$ , kde  $b \in \Sigma^*$  předpokládejme, že tvrzení platí pro  $y$ . Platí, že  $q \Rightarrow_G^* br \Rightarrow_G^* by = x$  právě když  $\delta(q, b) = r$  a  $\delta^*(r, y) \in F$

To znamená  $\delta^*(q, by) = \delta^*(\delta(q, b), y) \in F$

Předchozí dokazuje, že  $x \in L(G)$  právě když  $x \in L(A)$  pro každý neprázdný  $x$ .

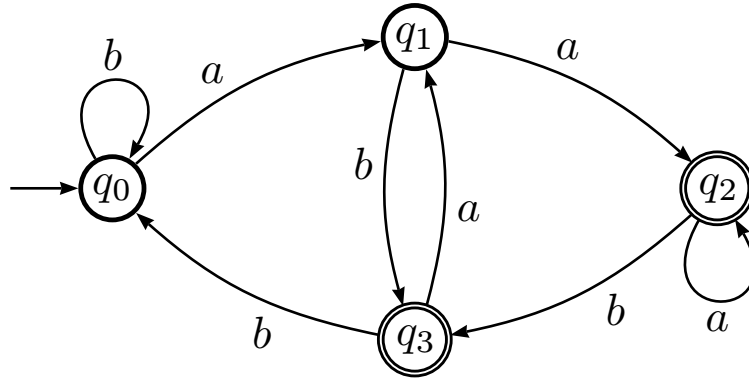
Pokud  $A$  nepřijímá  $\varepsilon$ , pak jsme hotovi.

Uvažujeme nový neterminál  $S$ , který bude nový startovní symbol, tj. místo  $G$  uvažujeme  $G' = \langle Q \cup \{S\}, \Sigma, P', S \rangle$

$$P' = \{S \rightarrow \varepsilon\} \cup \{S \rightarrow x \mid q_0 \rightarrow x \in P\} \cup P$$

Pak  $L(A) = L(G)$ .

**Příklad 26:** Mějme abecedu  $\Sigma = \{a, b\}$  a automat zadaný diagramem:



Odvozovací pravidla gramatiky, generující tento jazyk budou:

$$\begin{aligned}
 q_0 &\rightarrow aq_1 \mid bq_0 \\
 q_1 &\rightarrow aq_2 \mid a \mid bq_3 \mid b \\
 q_2 &\rightarrow aq_2 \mid a \mid bq_3 \mid b \\
 q_3 &\rightarrow aq_1 \mid bq_0
 \end{aligned}$$

### 12.3. Regulární gramatiky

Co jsou to regulární gramatiky a jaké podmínky jejich odvozovací pravidla splňují již víme, ale můžeme si je ještě rozdělit na dva druhy, právě podle tvaru odvozovacích pravidel.

1. *Zprava regulární gramatiky*: Obsahují pravidla ve tvaru  $A \rightarrow bB$ , tedy neterminál je napravo od terminálního symbolu.
2. *Zleva regulární gramatiky*: Obsahují pravidla ve tvaru  $A \rightarrow Bb$ . Analogicky se neterminál nachází vlevo od terminálního symbolu.

**Věta 27:** Pro každou zleva regulární gramatiku  $G = \langle N, \Sigma, P, S \rangle$  existuje konečný deterministický automat  $A$  tak, že  $L(A) = L(G)$ .

**Důkaz 20:** Budeme konstruovat automat, jehož stavy budou  $N$ , nový pomocný počáteční stav  $\#$  a jediný koncový stav je  $S$ .

Hledaný KNA  $A = \langle \Sigma, N \cup \{\#\}, \delta, \{\#\}, \{S\} \rangle$  s následovně definovanou přechodovou funkcí  $\delta$

$$\delta(q, a) = \begin{cases} \{A \in N \mid A \rightarrow a \in P\} & \text{pokud } q = \# \\ \{A \in N \mid A \rightarrow Ba \in P\} & \text{pokud } q = B \end{cases}$$

Ekvivalence  $L(A) = L(G)$  se dokazuje vzájemně jednoznačnou korespondencí P-derivace a nedeterministického výpočtu.

Pro derivaci:

$$x_0 = S, x_1, x_2, \dots, x_{n-1}, x_n = x$$

jsme schopni sestavit posloupnost

$$\langle \#, X_n \rangle, \langle A_{n-1}, y_{n-1} \rangle, \dots, \langle A_1, y_1 \rangle, \langle S, \varepsilon \rangle \text{ kde } x_i = A_i y_i$$

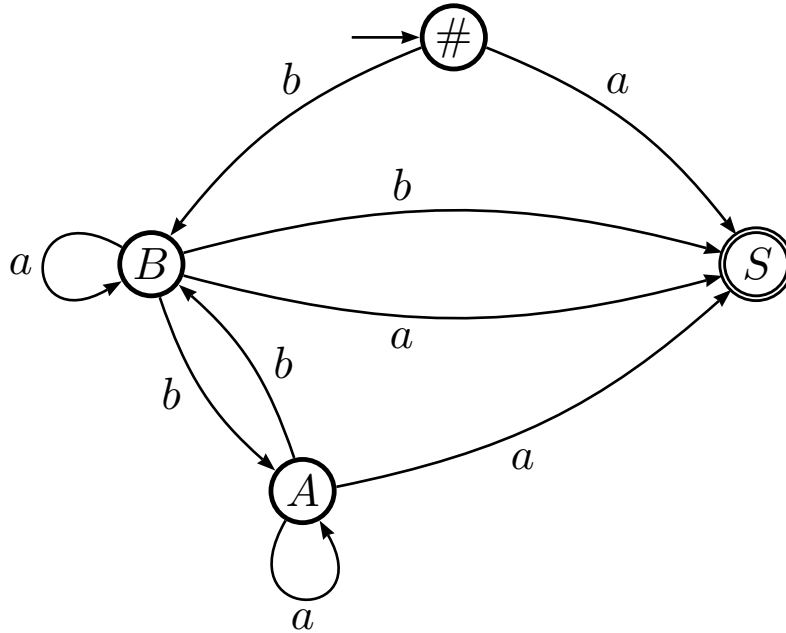
**Příklad 27:** Máme gramatiku  $G$  s následovně definovanými pravidly.

$$S \rightarrow Aa|Ba|Bb|a$$

$$A \rightarrow Aa|Bb$$

$$B \rightarrow Ab|Ba|b$$

Automat rozpoznávající jazyk generovaný touto gramatikou bude vypadat následovně:



**Věta 28:** Pro každý konečný deterministický automat  $A$  existuje zleva regulární gramatika taková, že  $L(A)=L(G)$

**Důkaz 21:** Neterminály gramatiky jsou stavy automatu a budeme uvažovat dodatečný startovní neterminál  $S$ .

$$\begin{aligned}
 P = \{ & \delta(q, a) \rightarrow qa \mid q \in Q \wedge a \in \Sigma \} \cup \\
 & \{ \delta(q_0, a) \rightarrow a \mid q_0 \text{ je počáteční stav} \} \cup \\
 & \{ S \rightarrow w \mid w \text{ je pravá strana každého pravidla } q \rightarrow w, \text{ kde } q \in F \}
 \end{aligned}$$

**Příklad 28:** Vezmeme KDA z příkladu 26. Odvozovací pravidla budou vypadat takto:

$$\begin{aligned}
q_0 &\rightarrow q_0b \mid b \mid q_3b \\
q_1 &\rightarrow q_0a \mid a \mid q_3a \\
q_2 &\rightarrow q_1a \mid q_2a \\
q_3 &\rightarrow q_1b \mid q_2b \\
S &\rightarrow q_1a \mid q_1b \mid q_2a \mid q_2b
\end{aligned}$$

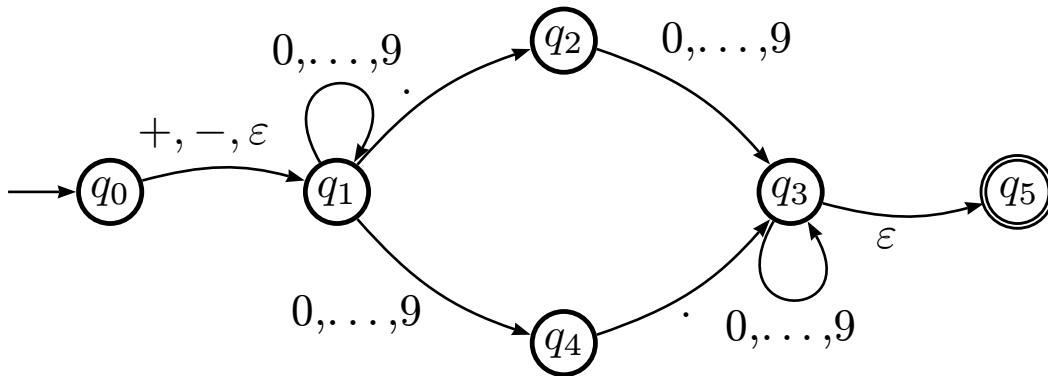
**Definice 15:** Regulární jazyky jsou jazyky, generované zprava (zleva) regulárními gramatikami, tj. jsou rozpoznatelné konečnými ne/deterministickými automaty.

**Poznámka 8:** Pravidla zprava a zleva nelze míchat.

### 13. Nedeterministický konečný automat s $\varepsilon$ -přechody

Značíme  $\varepsilon$ KNA.

**Příklad 29:** Zde je jeden motivační příklad na úvod.



**Definice 16:** Nedeterministický konečný automat s  $\varepsilon$ -přechody je struktura  $\langle \Sigma, Q, \delta, I, F \rangle$ , kde  $\Sigma, Q, \delta, I, F$  mají stejný význam jako u KNA.  $\delta$  je přechodová funkce  $\delta : Q \times (\Sigma \cup \{\varepsilon\}) \rightarrow 2^Q$ .

Fakt  $\delta(q, a) = \{r_1, \dots, r_k\}$  čteme: „automat  $A$  při čtení symbolu  $a$  přejde ze stavu  $q$  do některého ze stavů  $r_1, \dots, r_k$ .“

Fakt  $\delta(q, \varepsilon) = \{r_1, \dots, r_k\}$  čteme: „automat  $A$  přejde samovolně ze stavu  $q$  do některého ze stavů  $r_1, \dots, r_k$ .“

### 13.1. Reprezentace $\varepsilon$ KNA

1. **Přechodová tabulka**, vypadá stejně jako u KNA s tím, že přidáme jeden sloupec, ve kterém budeme zaznamenávat  $\varepsilon$ -přechody.

Takto bude vypadat předchozí příklad číslo 29, reprezentovaný pomocí tabulky.

	$+, -$	$.$	$0, \dots, 9$	$\varepsilon$
$\rightarrow q_0$	$\{q_1\}$	$\emptyset$	$\emptyset$	$\{q_1\}$
$q_1$	$\{q_2\}$	$\emptyset$	$\{q_1, q_2\}$	$\emptyset$
$q_2$	$\emptyset$	$\emptyset$	$\{q_3\}$	$\emptyset$
$q_3$	$\emptyset$	$\emptyset$	$\{q_3\}$	$\{q_5\}$
$q_4$	$\emptyset$	$\{q_3\}$	$\emptyset$	$\emptyset$
$q_5$	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$

Tabulka 3. Přechodová tabulka pro 29. příklad

2. **Přechodový diagram**, u kterého mohou být některé hrany ohodnoceny  $\varepsilon$ . Viz příklad 29.

### 13.2. Nedeterministický výpočet

Pojem **konfigurace** pro nás zůstává stejný, jedná se stále o dvojici  $\langle stav, řetězec \rangle$ .

**Definice 17:** Mějme automat  $A = \langle \Sigma, Q, \delta, I, F \rangle$  a řetězec  $w \in \Sigma^*$ . Posloupnost konfigurací  $\langle r_i, w_i \rangle$  pro  $i = 0, \dots, n$  splňující podmínky:

1.  $r_0 \in I, w_0 = w$
2.  $w_n = \varepsilon$
3. pro každé  $i = 0, \dots, n$ 
  - (a)  $w_i = aw_{i+1}, r_{i+1} \in \delta(r_i, a)$
  - (b)  $w_i = w_{i+1}, r_{i+1} \in \delta(r_i, \varepsilon)$

### 13.3. $\varepsilon$ -uzávěry množin stavů

Je dána množina stavů  $R \subseteq Q$ , jež se nazývá  $\varepsilon$ -uzavřená, pokud  $\delta(q, \varepsilon) \subseteq R$  pro každý stav  $q \in R$ .

**Příklad 30:** Lze ukázat na našem příkladě 29.

$\{q_0\}$  není  $\varepsilon$ -uzavřená protože  $\delta(q_0, \varepsilon) = \{q_1\} \not\subseteq \{q_0\}$

$\{q_0, q_1\}$  je  $\varepsilon$ -uzavřená

#### 13.3.1. Tvorba $\varepsilon$ -uzávěru

1. Mějme  $\varepsilon$ -uzávěr  $R$ .
2. Prohlašme, že  $E_0 = R$  a následně pro  $i \geq 1$  tvrdíme, že:

- $E_i = E_{i-1} \cup \{\delta(q, \varepsilon) | q \in E_{i-1}\}$
- $E_A(R) = \bigcup_{i=0}^{\infty} E_i$
- $E_0 \subseteq E_1 \subseteq E_2 \subseteq \dots \subseteq E_A(R)$

**Poznámka 9:** Vzhledem ke konečnosti množiny  $E_A(R)$  musí existovat index  $k$ , pro který platí  $E_k = E_{k+1} = \dots = E_A(R)$

Můžeme pozorovat, že  $E_A(R)$  není jen  $\varepsilon$ -uzavřená, ale je také **nejmenší**  $\varepsilon$ -uzavřená. Z toho můžeme vyvodit, že

$$E_A : 2^Q \rightarrow 2^Q$$

je **uzávěrový** operátor.

### 13.4. Rozšířená přechodová funkce

Pro automat  $A = \langle \Sigma, Q, \delta, I, F \rangle$  je rozšířená přechodová funkce definovaná jako

$$\delta^* : 2^Q \times \Sigma^* \rightarrow 2^Q$$

$$\delta^*(R, w) = \begin{cases} E_A(R) & \text{pokud } w = \varepsilon \\ \delta^*(E_A(\bigcup_{q \in E_A(R)} \delta(q, a), u)) & \text{pokud } w = au, a \in \Sigma, u \in \Sigma^* \end{cases}$$

**Věta 29:** Pro libovolné množiny  $R_i \subseteq Q$  ( $i = 1, \dots, k$ ) platí:

$$\bigcup_{i=1}^k E_A(R_i) = E_A(\bigcup_{i=1}^k R_i)$$

**Důkaz 22:** Z monotonie  $E_A$  dostáváme

$$E_A(R_i) \subseteq E_A(\bigcup_{i=1}^k R_i) \text{ pro všechna } i$$

$$\bigcup_{i=1}^k E_A(R_i) \subseteq E_A(\bigcup_{i=1}^k R_i)$$

Opačná inkluze

$$\bigcup_{i=1}^k E_A(R_i) \text{ je } \varepsilon\text{-uzavřená a obsahuje } \bigcup_{i=1}^k R_i$$

z extenzivity  $E_A$  plyne, že  $\bigcup_{i=1}^k R_i \subseteq \bigcup_{i=1}^k E_A(R_i)$

Stačí tedy ukázat, že  $\bigcup_{i=1}^k E_A(R_i)$  je  $\varepsilon$ -uzavřená.

$$q \in \bigcup_{i=1}^k E_A(R_i) \Rightarrow \exists k \ q \in E_A(R_i)$$

Protože  $E_A(R_i)$  je  $\varepsilon$ -uzavřená,  $\delta(q, \varepsilon) \in E_A(R_i) \Rightarrow \delta(q, \varepsilon) \in \bigcup_{i=1}^k E_A(R_i) \Rightarrow \bigcup_{i=1}^k E_A(R_i)$  je  $\varepsilon$ -uzavřená množina.

### 13.5. Přijímaný řetězec $\varepsilon$ KNA

Nechť  $A = \langle \Sigma, Q, \delta, I, F \rangle$  je  $\varepsilon$ KNA. Řetězec  $w \in \Sigma^*$  nazýváme řetězec **přijímaný**  $A$ , pokud

$$\delta^*(I, w) \cap F \neq \emptyset$$

jinak  $w$  nazýváme řetězec **zamítaný**  $A$ .

Jazyk přijímaný  $A$ :  $L(A) = \{w \in \Sigma^* | \delta^*(I, w) \cap F \neq \emptyset\}$

### 13.6. Ekvivalence s KDA

**Věta 30:** Ke každému  $\varepsilon$ KNA  $A = \langle \Sigma, Q, \delta, I, F \rangle$  existuje KNA  $A^S = \langle \Sigma, Q, \delta^S, I^S, F \rangle$  takový, že  $L(A) = L(A^S)$ .

**Důkaz 23:**  $I^S = E_A(I)$

$$\delta^S(q, a) = \delta^*(\{q\}, a) = E_A(\bigcup_{u \in E_A(\{q\})} \delta(u, a))$$

Indukcí přes délku řetězce  $w \in \Sigma^*$  prokážeme, že  $(\delta^S)^*(E_A(R), w) = \delta^*(R, w)$

1.  $w = \varepsilon$  platí triviálně
2. Předpokládejme, že tvrzení platí pro  $w$  délky  $n$  a dokážeme pro slova  $w$  délky  $n + 1$ .

$$w = av, \quad a \in \Sigma, \quad v \in \Sigma^*, \quad |v| = n, \quad R \subseteq Q$$

$$\begin{aligned}
 (\delta^S)^*(E_A(R), w) &= (\delta^S)^*(E_A(R), av) \\
 &= (\delta^S)^*\left(\bigcup_{q \in E_A(R)} \delta^S(q, a), u\right) \\
 &= (\delta^S)^*\left(\bigcup_{q \in E_A(R)} E_A\left(\bigcup_{u \in E_A(\{q\})} \delta(u, a)\right), u\right) \\
 &= (\delta^S)^*\left(E_A \bigcup_{q \in E_A(R)} \bigcup_{u \in E_A(\{q\})} \delta(u, a), u\right) \\
 &= (\delta^S)^*\left(E_A \bigcup_{q \in E_A(R)} \delta(q, a), u\right) \\
 &= (\delta^S)^*\left(E_A\left(E_A \bigcup_{q \in E_A(R)} \delta(q, a)\right), u\right) \\
 &= \delta^*\left(E_A \bigcup_{q \in E_A(R)} \delta(q, a), u\right) \\
 &= \delta^*(R, w)
 \end{aligned}$$

## 14. Algoritmus na převod $\varepsilon$ KNA na KDA

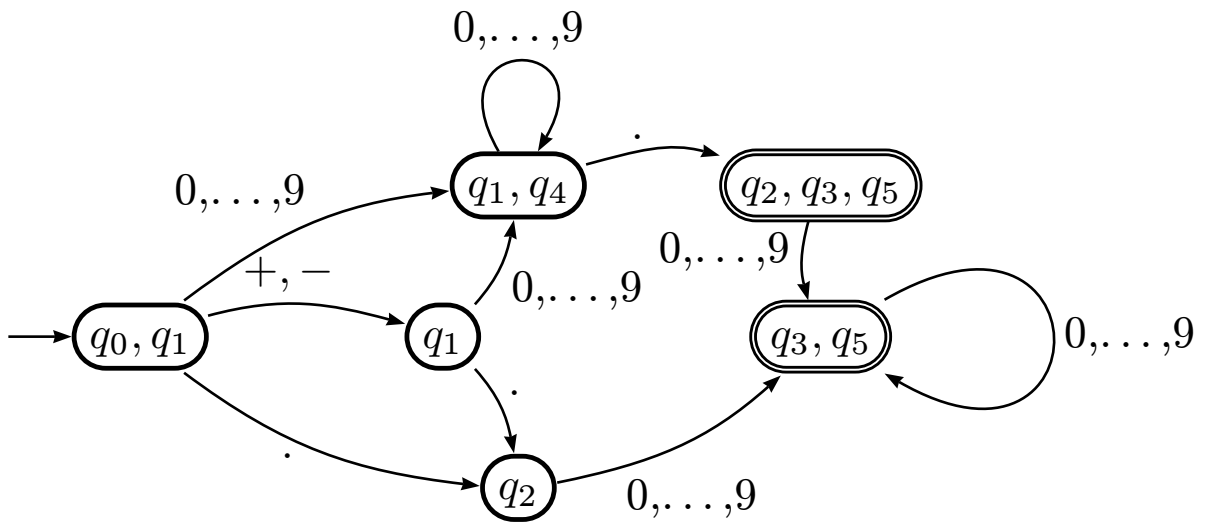
Máme  $A = \langle \Sigma, Q, \delta, I, F \rangle$  a víme, že  $W$  je neprázdná množina dosud nezpracovaných stavů.



```

 $\delta^s \leftarrow \emptyset; Q^s \leftarrow \emptyset; F^s \leftarrow \emptyset; W \leftarrow E_A(I)$ 
while  $W \neq Q$  do
  select  $R \in W$ 
   $W \leftarrow W \setminus R; Q^s \leftarrow Q^s \cup R$ 
  if  $R \cap F \neq \emptyset$  then
     $F^s \leftarrow F^s \cup R$ 
  endif
  foreach  $a \in \Sigma$  do
     $N \leftarrow \delta^*(R, a)$ 
    if  $N \neq \emptyset$  then
      if  $N \notin W \cup Q^s$  then
         $W \leftarrow W \cup N$ 
      endif
       $\delta^s \leftarrow \delta^s \cup \langle R, a, N \rangle$ 
    endif
  endfor
endwhile
return  $\langle \Sigma, Q^s, \delta^s, I, F^s \rangle$ 

```

Obrázek 6. Pseudokód pro převod  $\varepsilon$  KNA na KDA.

## 15. Regulární výrazy

**Definice 18:** Necht' je dána  $\Sigma = \{a_1, \dots, a_k\}$ . Pak regulární výraz na  $\Sigma$  je:

1.  $\emptyset$
2.  $\varepsilon$
3. symboly  $a_1, \dots, a_k$
4. pokud  $R_1, R_2$  jsou RV, pak  $(R_1|R_2)$  je RV
5. pokud  $R_1, R_2$  jsou RV, pak  $(R_1R_2)$  je RV

6. pokud  $R$  je RV, pak  $R^*$  je RV

**Příklad 31:** Podívejme se na následující výrazy a rozhodněme, zda-li jsou regulární:

- $((ab)c)^*$ ,  $((a|b)c)^*$  – jsou RV
- $a^*b$ ,  $a||b$  – nejsou RV

**Poznámka 10:** Priorita  $|, \circ, *$  je stejná jako (po řadě)  $+, \cdot, ^{-1}$  v aritmetických výrazech.

## 16. Jazyky generované regulárními výrazy

**Definice 19:** Necht  $R$  je RV nad  $\Sigma$ . Pak  $L(R) \subseteq \Sigma^*$ . Pak také platí:

1.  $L(R) = \emptyset$ , pokud  $R = \emptyset$ .
2.  $L(R) = \{\varepsilon\}$ , pokud  $R = \varepsilon$ .
3.  $L(R) = \{a_i\}$  pokud  $R = a_i$ .
4.  $L(R) = L(R_1) \cup L(R_2)$ , pokud  $R = \{R_1|R_2\}$ .
5.  $L(R) = L(R_1) \circ L(R_2)$ , pokud  $R = \{R_1 \circ R_2\}$ .
6.  $L(R) = L(R_1)^*$ , pokud  $R = R_1^*$ .

**Věta 31:** Každý regulární výraz lze převést na konečný automat.

**Věta 32:** Každý jazyk generovaný regulárním výrazem je regulární.

**Důkaz 24:** Předchozí body dokážeme indukcí dle složitosti regulárního výrazu.

- Pro body 1 až 3 je vše zřejmé.
- Pro bod 4 –  $R = R_1|R_2$ , kde  $R_1, R_2$  jsou regulární výrazy. Předpokládáme, že existuje KDA, rozpoznávající jazyky  $L(R_1)$  a  $L(R_2)$ ,  $L(R_1) = L(A_1)$ ,  $L(R_2) = L(A_2)$ .  
Pak vytvoříme KNA, který má tvar  $\langle \Sigma, Q_1 \cup Q_2, \delta_1 \cup \delta_2, \{q_0, q_0'\}, F_1 \cup F_2 \rangle$ .
- Pro bod 5 –  $R = R_1R_2$ . Z koncových stavů  $A_1$  vytvoříme  $\varepsilon$ -přechody do počátečního stavu automatu  $A_2$  a počátečním stavem bude stav  $q_0$  z automatu  $A_1$ .
- Pro bod 6 –  $R = R_1^*$ ,  $L(A_1) = L(R_1)$ . Následně sestavujeme  $\varepsilon$  KNA tak, že před počátečním stavem vytvoříme nový koncový stav, který bude navíc novým počátečním stavem a do kterého vedeme  $\varepsilon$  přechody z již existujících koncových stavů a z našeho nového koncového stavu navíc vedeme  $\varepsilon$  přechod do původního počátečního stavu.

**Věta 33:** Každý regulární jazyk lze generovat regulárním výrazem.

## 17. Uzávěrové vlastnosti regulárních jazyků

### 17.1. Základní uzávěrové vlastnosti

- **Komplement**

Pokud je  $L$  regulární, pak je  $\Sigma^* \setminus L$  regulární.

**Důkaz 25:** Pro  $L$  existuje KDA s úplnou přechodovou funkcí tak, že  $L(A) = L$ . Na základě tohoto automatu zkonstruujeme  $A' = \langle \Sigma, Q, \delta, q_0, Q \setminus F \rangle$ ,  $A$  i  $A'$  mají stejnou rozšířenou přechodovou funkci  $\delta^*$ .

Tj.  $\Sigma^* \setminus L = L(A')$

- **Sjednocení**

Jsou-li  $L_1$  a  $L_2$  regulární, pak je  $L_1 \cup L_2$  regulární.

**Důkaz 26:** Pro  $L_1$  existuje  $A_1 = \langle \Sigma, Q_1, \delta_1, q_{01}, F_1 \rangle$ , tak že  $L_1 = L(A_1)$

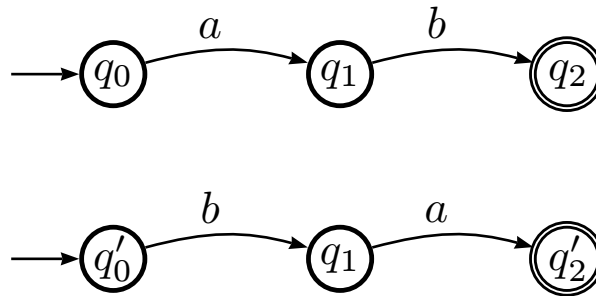
Pro  $L_2$  existuje  $A_2 = \langle \Sigma, Q_2, \delta_2, q_{02}, F_2 \rangle$ , tak že  $L_2 = L(A_2)$

Předpokládáme-li, že  $A_1$  a  $A_2$  mají disjunktní množiny stavů, tj.  $Q_1 \cap Q_2 = \emptyset$ , sestavíme následující KNA:

$A = \langle \Sigma, Q_1 \cup Q_2, \delta, \{q_{01}, q_{02}\}, F_1 \cup F_2 \rangle$  s přechodovou funkcí  $\delta$  definovanou jako

$$\delta(q, a) = \begin{cases} \{\delta_1(q, a)\} & \text{pokud } q \in Q_1 \\ \{\delta_2(q, a)\} & \text{pokud } q \in Q_2 \end{cases}$$

**Příklad 32:** Nyní si uvedeme protipříklad, co by se stalo, kdyby množiny stavů nebyly disjunktní.



První automat přijímá řetězec  $ab$ , druhý automat přijímá řetězec  $ba$ , čili od jejich sjednocení očekáváme, že bude přijímat  $ab$  i  $ba$ . Jelikož množiny stavů nejsou disjunktní ( $q_1$  je společný pro oba), sjednocení těchto automatů může stejně dobře přijímat i řetězce  $aa$  nebo  $bb$ , což je nežádoucí.

- **Průnik**

S použitím De Morganových zákonů, dostáváme:

$$L_1 \cap L_2 = \Sigma^* \setminus (\Sigma^* \setminus L_1 \cup \Sigma^* \setminus L_2)$$

tj.  $L_1 \cap L_2$  je regulární.

**Důkaz 27:** Uvažujeme konečné deterministické automaty s úplnou přechodovou funkcí

$$A_1 = \langle \Sigma, Q_1, \delta_1, q_{01}, F_1 \rangle \text{ a } A_2 = \langle \Sigma, Q_2, \delta_2, q_{02}, F_2 \rangle$$

Zkonstruujeme automat  $A_1 \times A_2$  (direktní součin):

$$A_1 \times A_2 = \langle \Sigma, Q_1 \times Q_2, \delta, \langle q_{01}, q_{02} \rangle, F_1 \times F_2 \rangle$$

s přechodovou funkcí  $\delta$ :

$$\delta(\langle q, r \rangle, a) = \langle \delta_1(q, a), \delta_2(r, a) \rangle$$

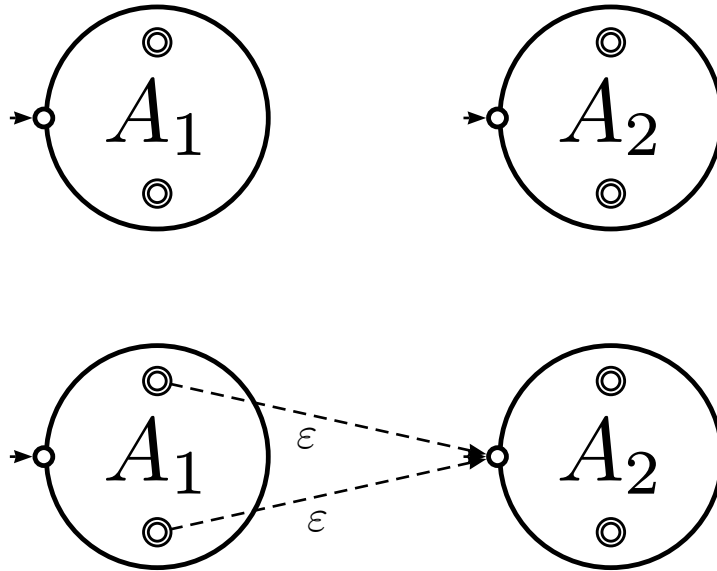
platí:

$$L_1 \cap L_2 = L(A_1 \times A_2)$$

- **Produkt (zřetězení)**

$$L_1 \cdot L_2$$

Předpokládáme existenci automatů  $A_1 = \langle \Sigma, Q_1, \delta_1, q_{01}, F_1 \rangle$  a  $A_2 = \langle \Sigma, Q_2, \delta_2, q_{02}, F_2 \rangle$  takových, že  $L_1 = L(A_1)$  a  $L_2 = L(A_2)$ .



Sestavíme  $\varepsilon$ KNA

$$A = \langle \Sigma, Q_1 \cup Q_2, \delta, \{q_{01}\}, F_2 \rangle$$

s přechodovou funkcí  $\delta : (Q_1 \cup Q_2) \times (\Sigma \cup \{\varepsilon\}) \rightarrow 2^{Q_1 \cup Q_2}$

$$\delta(q, a) = \begin{cases} \{\delta_1(q, a)\} & \text{pokud } q \in Q_1 \\ \{\delta_2(q, a)\} & \text{pokud } q \in Q_2 \end{cases}$$

$$\delta(q, \varepsilon) = \begin{cases} \{q_{02}\} & \text{pokud } q \in F_1 \\ \emptyset & \text{pokud } q \notin F_1 \end{cases}$$

- **Kleeneho uzávěr**

Pokud je  $L$  regulární, pak je  $L^*$  regulární.

Předpokládáme, že existuje automat  $A$  tak, že  $L = L(A)$

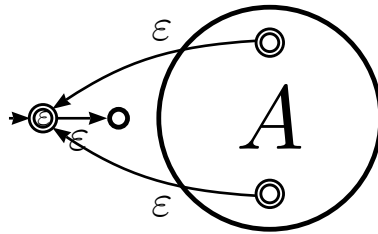
**Poznámka 11:** Automat rozpoznávající  $L^*$  musí mít možnost dostat se z koncového stavu zpět na začátek.

Pro automat  $A = \langle \Sigma, Q, \delta, q_0, F \rangle$  je potřeba zavést nový počáteční stav  $q_T \notin Q$ . Konstruovaný  $\varepsilon$ KNA bude vypadat následovně:

$$A' = \langle \Sigma, Q \cup \{q_T\}, \delta', \{q_T\}, F \cup \{q_T\} \rangle$$

s přechodovou funkcí  $\delta'$ :

$$\begin{aligned} \delta'(q, a) &= \{\delta(q, a)\} && \text{pokud } q \in Q \\ \delta'(q, \varepsilon) &= \emptyset && \text{pokud } q \in Q \setminus F \\ \delta'(q, \varepsilon) &= \{q_T\} && \text{pokud } q \in F \\ \delta'(q_T, \varepsilon) &= \{q_0\} \\ \delta'(q_T, a) &= \emptyset && a \in \Sigma \end{aligned}$$



## 17.2. Další uzávěrové vlastnosti

- **Množinový rozdíl**

Jsou-li  $L_1$  a  $L_2$  regulární, pak  $L_1 \setminus L_2 = L_1 \cap (\Sigma^* \setminus L_2)$  je regulární.

- **Kleeného pozitivní uzávěr**

Je-li  $L$  regulární, pak  $L^+ = (L^* \setminus \{\varepsilon\}) \cup L$  je regulární.

- **N-tá mocnina jazyka**

$L^n \dots$  plyne z uzavření na produkt

- **Jazyk reverzních řetězců**

$$L^R = \{w^R \mid w \in L\}$$

Zdůvodníme pomocí konstrukce automatu rozpoznávající  $L^R$  (viz cvičení 6)

- **Jazyk sufixů**

$$Sfx(L) = \bigcup_{w \in L} Sfx(w)$$

**Důkaz 28:** Pro  $L$  uvažujeme  $A$  tak, že  $L = L(A)$ . Navíc  $A$  je KDA s úplnou přechodovou funkcí takový, že všechny jeho stavy jsou dosažitelné.

Námi hledaný automat je KNA definován jako:

$$A' = \langle \Sigma, Q, \delta, Q, F \rangle$$

s přechodovou funkcí

$$\delta(q, a) = \{\delta(q, a)\}$$

**Poznámka 12:** Všechny stavy jsou označeny za počáteční, aby měl automat možnost skočit do libovolné fáze výpočtu a tím "uhádnout" vynechané znaky řetězce, jehož sufix zkoumáme.

- **Jazyk prefixů**

$$\begin{aligned} Pfx(L) &= \bigcup_{w \in L} Pfx(w) \\ Pfx(L) &= (Sfx(L^R))^R \end{aligned}$$

Důkaz plyne z uzavření na Sfx a reverzní řetězec.

- **Jazyk infixů**

$$Ifx(L) = Pfx(Sfx(L))$$

Důkaz je taktéž zřejmý.

### 17.3. Pumping lemma

**Poznámka 13:** Jen drobné upozornění na začátek: jedná se o tvrzení ve tvaru když  $\rightarrow$  pak, tj. "Pokud je  $L$  regulární, pak ..."

**Věta 34:** Necht  $L$  je regulární jazyk nad  $\Sigma$ . Pak existuje  $n \in \mathbb{N}$  tak, že pro každý řetězec  $w \in L$  délky alespoň  $n$  platí, že existují  $x, y, z \in \Sigma^*$  tak, že jsou splněny podmínky:

1.  $w = xyz$
2.  $|xy| \leq n$
3.  $y \neq \varepsilon$
4. pro každé  $i \geq 0$  platí, že  $xy^iz \in L$

**Důkaz 29:** Rozlišíme dva případy.

- **$L$  je konečný**

pak je tvrzení triviální. Hledané  $n$  je ve tvaru  $l + 1$ , kde  $l$  je délka nejdelšího řetězce z  $L$ . Pak není v  $L$  žádný řetězec delší než  $n$  a tvrzení 1. – 4. platí triviálně.

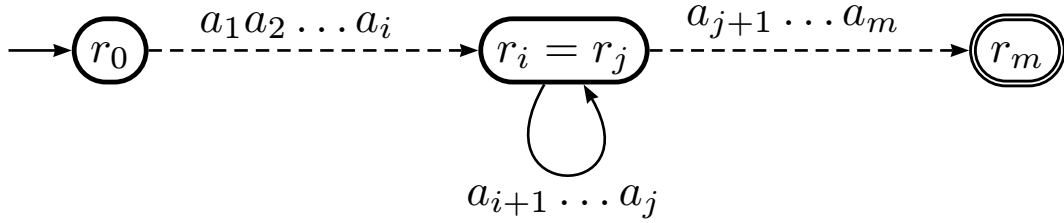
- **$L$  je nekonečný**

pak pro něj existuje KDA s množinou stavů  $Q$  tak, že  $L(A) = L$ . Položíme  $n = |Q|$ .

Pro každý řetězec  $w = a_1a_2 \cdots a_m$  kde  $m \geq n$  existuje přijímací výpočet délky  $m$ :

$$\langle q_0, w \rangle = \langle r_0, a_1 \cdots a_m \rangle, \langle r_1, a_2 \cdots a_m \rangle \cdots \langle r_{m-1}, a_m \rangle, \langle r_m, \varepsilon \rangle$$

Jelikož je  $m \geq n$ , tak nevyhnutelně musí existovat alespoň 1 stav, který je v tomto přijímacím výpočtu zopakován.



Následně dle obrázku mějme:

$$\begin{aligned} x &= a_1 a_2 \dots a_i \\ y &= a_{i+1} a_{i+2} \dots a_j \\ z &= a_{j+1} a_{j+2} \dots a_m \end{aligned}$$

Vidíme tedy, že všechny podmínky Pumping lemma jsou splněny.

**Příklad 33:** Jazyk  $L = \{a^n b^n \mid \text{kde } n \in N\}$  není regulární.

**Důkaz 30:** Předchozí příklad 33 není regulární, což dokážeme sporem. Nechť je tedy  $L$  regulární a dle předchozí věty existuje číslo  $n$  tak, že vezmeme řetězec  $a^n b^n = xyz$  tak, že  $x = a^k, y = a^l, z = a^{n-k-l} b^n$ .

Tím jsme došli ke sporu, protože  $xy^i z$  pro každé  $i \geq 0$  obecně nemusí patřit do jazyku  $L$  (ovlivníme tím počet  $a$  v řetězci a tím se počet znaků  $a$  a  $b$  nerovná).

## 18. Minimalizace KDA

**Poznámka 14:** Pro regulární jazyk  $L$  existuje více, než jeden automat  $A$  tak, že  $L = L(A)$  a navíc můžeme mít  $A_1, A_2$  tak, že  $L(A_1) = L(A_2)$ , ale  $|Q_1| < |Q_2|$ .

### 18.1. Zprava invariantní ekvivalence

**Definice 20:** Předpokládejme, že máme  $A = \langle \Sigma, Q, \delta, q_0, F \rangle$  a relaci ekvivalence  $\Theta \subseteq Q \times Q$  nazveme **zprava invariantní ekvivalencí** vzhledem k  $\delta$ , pokud platí, že  $\langle q, r \rangle \in \Theta$  a  $a \in \Sigma$ , pak  $\langle \delta(q, a), \delta(r, a) \rangle \in \Theta$ .

Pravá invariance reprezentuje přirozenou vlastnost, kterou musí mít relace nerozlišitelnosti stavů.

Mezními případy invariantních relací zprava jsou:

1. identita  $\Theta = \{\langle q, q \rangle \mid q \in Q\}$
2.  $\Theta = Q \times Q$

## 18.2. Faktorizace automatu

**Definice 21:** Mějme KDA  $A = \langle \Sigma, Q, \delta, q_0, F \rangle$  a zprava invariantní ekvivalenci  $\Theta$  vzhledem k  $\delta$ , pak zavedeme  $A/\theta = \langle \Sigma, Q/\Theta, \delta^{A/\Theta}, [q_0]_\Theta, F^{A/\Theta} \rangle$ , kde

$$\delta^{A/\Theta} = ([q]_\Theta, a) = [\delta(q, a)]_\Theta \text{ a } F^{A/\Theta} = \{[q]_\Theta \mid q \in F\}$$

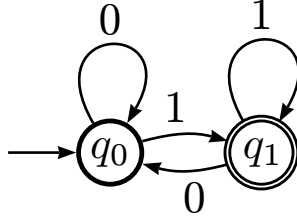
**Věta 35:** Automat  $A$  je dobře definovaný KDA.

**Důkaz 31:** Větu 35 si ověříme tak, že definice  $\delta^{A/\Theta}$  nezávisí na výběru prvků z třídy rozkladu dle  $\Theta$ .

Mějme  $[q]_\Theta = [r]_\Theta$ , to jest  $\langle q, r \rangle \in \Theta$ . Potom pro  $a \in \Sigma$  platí  $\langle \delta(q, a), \delta(r, a) \rangle \in \Theta$ , to jest  $[\delta(q, a)]_\Theta = [\delta(r, a)]_\Theta$ .

Obecně  $L(A/\Theta) \neq L(A)$ . Snažíme se najít co největší  $\Theta$ , tak aby tato rovnost platila.

**Příklad 34:**  $\Theta = Q \times Q$



**Věta 36:** Platí, že  $(\delta^{A/\Theta})^*([q]_\Theta, x) = [\delta^*(q, x)]_\Theta$  pro každý  $x \in \Sigma^*$ .

**Důkaz 32:** Dokážeme indukci přes délku řetězce.

- Tedy  $x = \varepsilon$ , pak

$$(\delta^{A/\Theta})^*([q]_\Theta, \varepsilon) = [q]_\Theta = [\delta^*(q, \varepsilon)]_\Theta$$

- pak nechť toto tvrzení platí pro  $u \in \Sigma^*$  a řetězec  $x = au$ , kde  $a \in \Sigma$ .

$$\begin{aligned} (\delta^{A/\Theta})^*([q]_\Theta, au) &= (\delta^{A/\Theta})^*(\delta^{A/\Theta}([q]_\Theta, a), u) = \\ &= (\delta^{A/\Theta})^*([\delta(q, a)]_\Theta, u) = [\delta^*(\delta(q, a), u)]_\Theta = [\delta^*(q, au)]_\Theta \end{aligned}$$

**Definice 22:** Mějme KDA  $A = \langle \Sigma, Q, \delta, q_0, F \rangle$  s úplnou přechodovou funkcí. Pro stavy  $q, r \in Q$  položme  $q \equiv_A r$ , pokud pro každý řetězec  $x \in \Sigma^*$  platí, že  $\delta^*(q, x) \in F$ , právě když  $\delta^*(r, x) \in F$ .

**Věta 37:**  $\equiv_A$  je zprava invariantní operace.



**Důkaz 33:** Dokazujeme větu 37. Důkaz reflexivity je zřejmý, stejně tak symetrie i tranzitivita.

Nechť  $q \equiv_A r$  a máme  $a \in \Sigma$ . Máme dokázat, že  $\delta(q, a) \equiv_A \delta(r, a)$ . Pro každé  $x \in \Sigma^*$  platí:

$$\begin{aligned}\delta^*(\delta(q, a), x) &= \delta^*(q, ax) \\ \delta^*(\delta(r, a), x) &= \delta^*(r, ax)\end{aligned}$$

Užitím faktu, že  $\delta^{*(q, ax)} \in F$  dostaneme  $\delta^*(r, ax) \in F$ , tedy  $\delta^*(\delta(q, a), x) \in F$  právě tehdy, když  $\delta^*(\delta(r, a), x) \in F$ , to jest  $\delta(q, a) \equiv_A \delta(r, a)$  a tedy  $A/\equiv_A$ .

**Věta 38:** Pro  $A/\equiv_A$  a stav  $q \in Q$  platí, že  $q \in F$  právě, když  $[q]_{\equiv_A} \in F^{A/\equiv_A}$ .

**Důkaz 34:** Předchozí větu dokážeme tak, že dokážme implikace z obou stran.

- Implikace zleva doprava („ $\Rightarrow$ “) plyne z definice.
- Implikace zprava doleva („ $\Leftarrow$ “): pokud  $[q]_{\equiv_A} \in F^{A/\equiv_A}$ , pak z definice  $\exists r \in F$  tak, že  $r \in [q]_{\equiv_A}$ . To znamená, že  $r \equiv_A q$  pro každý  $x \in \Sigma^*$  platí, že  $\delta^*(r, x) \in F$  právě tehdy, když  $\delta^*(q, x) \in F$ , speciálně pro  $x = \varepsilon$ , navíc  $\delta^*(r, \varepsilon) = r \in F$ , to jest  $\delta^*(q, \varepsilon) = q \in F$ .

**Důsledek 2:** KDA  $A$  nazveme redukovaný, pokud je  $\equiv_A$  identita.

**Věta 39:** KDA  $A/\equiv_A$  je redukovaný.

**Důkaz 35:** Automat  $A \equiv_A$  označme jako  $B$ , následně prokážeme, že  $\equiv_B$  je identita, tozn., že pokud  $[q]_{\equiv_A} \equiv_B [r]_{\equiv_A}$ , tak pak  $[q]_{\equiv_A} = [r]_{\equiv_A}$ .

Předpokládejme, že platí  $[q]_{\equiv_A} \equiv_B [r]_{\equiv_A}$ .

- Pak podle definice  $\equiv_B$  tozn, že pro  $\forall x \in \Sigma^*$  platí  $(\delta^{A/\equiv_A})^*([q]_{\equiv_A}, x) \in F^{A/\equiv_A}$  právě, když  $(\delta^{A/\equiv_A})^*([r]_{\equiv_A}, x) \in F^{A/\equiv_A}$ .
- S využitím věty 36 pro každé  $x \in \Sigma^*$ :

$$[\delta^*(q, x)]_{\equiv_A} \in F^{A/\equiv_A} \text{ právě tehdy, když } [\delta^*(r, x)]_{\equiv_A} \in F^{A/\equiv_A}$$

Následně aplikujeme větu 38:

$$\delta^*(q, x) \in F \text{ právě tehdy, když } \delta^*(r, x) \in F, \text{ tozn., že } q \equiv_A r \text{ a tedy } [q]_{\equiv_A} = [r]_{\equiv_A}$$

**Věta 40:**  $L(A) = L(A/\equiv_A)$ , což plyne užitím vět 38 a 36.

**Důkaz 36:** Máme dokázat, že  $\delta^*(q_0, x) \in F$ , právě, když  $(\delta^{A/\equiv_A})^*([q_0]_{\equiv_A}, x) \in F^{A/\equiv_A}$ .

Z věty 36:  $\delta^*(q_0, x)$  právě, když  $[\delta^*(q_0, x)]_{\equiv_A} \in F^{A/\equiv_A}$ , to ale platí z věty 38.

**Důsledek 3:** Obecně platí, že  $L(A) \subseteq L(A/\Theta)$ .

**Věta 41:** Pokud je každý stav automatu  $A$  dosažitelný, pak má  $A/\equiv_A$  také každý stav dosažitelný.

### 18.3. Algoritmus pro hledání redukovaného automatu

1. Označíme  $A/\equiv_A$  jako  $A^R$ , konstruuje posloupnost rozkladů na  $Q$  tak, že  $\varphi_1, \varphi_2, \dots, \varphi_i = \varphi_{i+1}$ , kde  $\varphi_1 = \{F, Q \setminus F\}$
2. Při odvození rozkladů  $\varphi_i$  z  $\varphi_{i-1}$  postupujeme následovně:  
Pro  $\forall R \in \varphi_{i-1}$  provedeme:
  - (a) Vezmeme libovolný stav  $r \in R$ .
  - (b) Položíme  $S = \{s \in R \mid \text{pro každé } a \in \Sigma \text{ platí také, že } \delta(S, a) \in [\delta(R, a)]\}$ .
  - (c) Pokud  $S = R$ , pak vložíme  $R$  do  $\varphi_i$ , pokud  $S \subsetneq R$ , pak vložíme  $S$  a  $R \setminus S$  do  $\varphi_i$ .

**Věta 42:** Korektnost algoritmu pro nalezení  $\equiv_A$ : pokud  $\varphi_i = \varphi_{i+1}$ , pak  $\varphi_i = Q/\equiv_A$ .

**Důkaz 37:** Ověříme, že  $q \equiv_A r$  právě, když  $q \in [r]_{\varphi_i}$ .

Pokud  $q \in [r]_{\varphi_i}$ , pak z toho, jak jsme zavedli posloupnost rozkladů, plyne, že nemůže platit  $q \equiv_A r$ .

Pokud  $q \equiv_A r$ , pak  $q \in [r]_{\equiv_A}$  a zbývá dokázat opačnou implikaci, což provedeme indukcí přes délku řetězce:

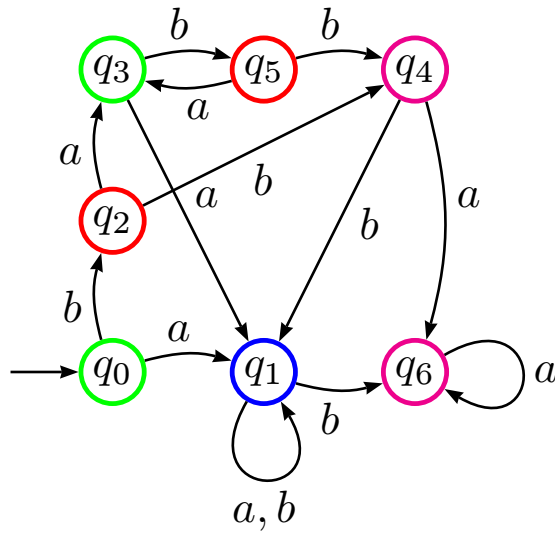
- Máme libovolná  $q, r \in Q$ .
  - Pokud  $q \in [r]_{\varphi_i}$ , pak  $\delta^*(q, x) \in F$  právě, když  $\delta^*(r, x) \in F$ .
1. Pro  $x = \varepsilon$  je situace triviální. Máme dokázat, že  $q \in F$  právě, když  $r \in F$ , ale to obecně platí, protože  $\varphi_i = \{F, Q \setminus F\}$
  2. Pro  $x \in \Sigma^*$ ,  $x = au$ , kde  $a \in \Sigma, u \in \Sigma^*$ . Z předpokladu, že  $q \in [r]_{\varphi_i}$  a  $\varphi_i = \varphi_{i+1}$  platí, že  $[\delta(q, a)]_{\varphi_i} = [\delta(r, a)]_{\varphi_i}$ , to jest  $\delta(q, a) \in [\delta(r, a)]_{\varphi_i}$  - zde použijeme indukční předpoklad.

$$\begin{aligned} \delta^*(\delta(q, a)r) &\in F \text{ právě tehdy, když } \delta^*(q, x) \in F \\ \delta^*(\delta(r, a)r) &\in F \text{ právě tehdy, když } \delta^*(r, x) \in F \end{aligned}$$

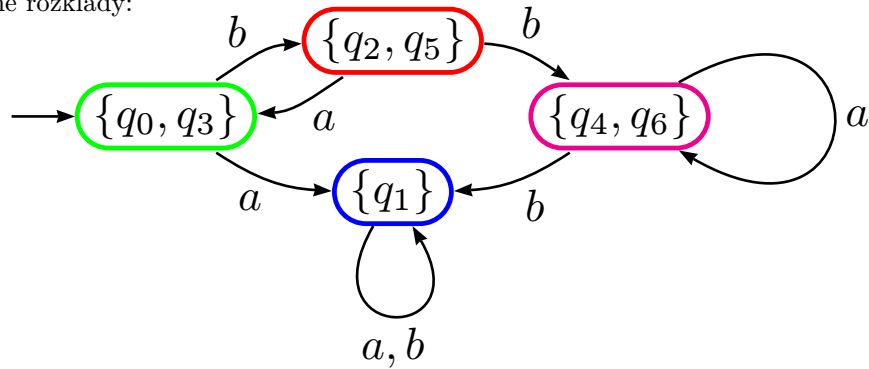
Tvrzení dokázáno pro každé  $x$ , tedy  $q \equiv_A r$ .

**Příklad 35:** Mějme výraz  $b(ab)^*ba^*$ . Dále

$$\begin{aligned} \varphi_1 &= \{\{q_0, q_1, q_2, q_3, q_5\}, \{q_4, q_6\}\} \\ \varphi_2 &= \{\{q_0, q_1, q_3\}, \{q_2 \cdot q_5\}, \{q_4, q_6\}\} \\ \varphi_3 &= \{\{q_0, q_3\}, \{q_1\}, \{q_2 \cdot q_5\}, \{q_4, q_6\}\} \end{aligned}$$



Následně rozklady:



	$q_0$	$q_1$	$q_2$	$q_3$	$q_4$	$q_5$	$q_6$
$q_0$	*	X	X	*	X	X	X
$q_1$	-	*	X	X	X	X	X
$q_2$	-	-	*	X	X	*	X
$q_3$	-	-	-	*	X	X	X
$q_4$	-	-	-	-	*	X	*
$q_5$	-	-	-	-	-	*	X
$q_6$	-	-	-	-	-	-	*

Tabulka 4. Tabulkový popis rozkladů stavů

V tabulce 4. se objevují znaky **X** na pozicích, pro které platí, že:

$$T[q, x] \quad \text{kde platí} \quad q \in F, x \notin F \\ \text{nebo} \quad q \notin F, x \in F$$

Projdeme prázdná místa,  $T[q, r] = \text{„prázdnó, pokud“ } \exists a \in \Sigma \text{ tak, že:}$

- buď  $T[\delta(q, a), \delta(r, a)] = X$
- nebo  $T[\delta(r, a), \delta(q, a)] = X$

Pokud ano, tak „X“ na pozici  $[q, r]$ .

## 19. Izomorfismus automatů

Pro KDA  $A_1$  a  $A_2$  mějme zobrazení

$$h : Q_1 \rightarrow Q_2$$

a toto zobrazení označme jako **izomorfismus**, pokud platí:

1. Zobrazení  $h$  je bijektivní.
2. Počáteční stav automatu  $A_1$  e zobrazí na počáteční stav automatu  $A_2$ .
3. Pro všechna  $q \in Q$  platí, že  $Q \in F_1$  právě, pokud  $q \in F_2$ .
4. Zobrazení  $h$  je kompatibilní s přechodovou funkcí.

**Věta 43:** Jsou-li dva automaty izomorfní, pak  $L(A_1) = L(A_2)$ .

**Definice 23:** Mějme regulární jazyk  $L$ , pak KDA s úplnou přechodovou funkcí je **minimálním automatem** pro  $L$ , pokud  $L(A) = L$  a pro každý KDA  $B$  takový, že  $L(B) = L$  platí, že  $B$  má tolik stavů jako  $A$ .

**Věta 44:** Je-li automat  $A$  minimální pro jazyk  $L$ , pak je **redukováný** nemá nedosažitelné stavy.

**Věta 45:** Pokud jsou automaty  $A, B$  KDA bez nedosažitelných stavů a pokud jsou tyto automaty navíc redukováné a existuje stejný jazyk, který generují, pak jsou izomorfní.

**Věta 46:**  $A$  je minimální automat pro jazyk  $L$  právě tehdy, když  $A$  neobsahuje nedosažitelné stavy a je redukováný.

## 20. Bezkontextové gramatiky

V této části se vrátíme k problematice bezkontextových gramatik. Před dalším čtením je potřeba ovládat základní pojmy, zejména

- Bezkontextové gramatiky
- Bezkontextové jazyky
- P-derivace
- Odvozování řetězců
- Věty a větné formy

Navíc si zavedeme duální pojem k derivaci - **redukci**.

**Definice 24:** Řetězec  $v$  lze redukovat na řetězec  $u$ , pokud  $u \Rightarrow_G^* v$ . Značíme  $v \Leftarrow_G^* u$ .

V následujícím příkladě si ukážeme problém nejednoznačnosti bezkontextových gramatik.

**Příklad 36:** Mějme gramatiku:

$$\begin{aligned} G &= \langle N, \Sigma, P, S \rangle \\ \Sigma &= \{a, b, c, 0, 1, +, -, *, /, (, )\} \\ N &= \{S, E, C, V\} \\ P &= \{ \begin{array}{l} S \rightarrow E, \\ E \rightarrow E * E | E / E | E + E | E - E | - E | C | V | (E), \\ C \rightarrow 0C | 1C | 0 | 1, \\ V \rightarrow aV | bV | cV | a | b | c \end{array} \} \end{aligned}$$

Uvažujme větu  $w = ac * 1 - c$ . K ní se lze dostat buď:

$$\begin{aligned} S &\Rightarrow_G E \Rightarrow_G E * E \Rightarrow_G V * E \Rightarrow_G V * E - E \Rightarrow_G V * E - V \Rightarrow_G aV * E - V \Rightarrow_G \\ ac * E - V &\Rightarrow_G ac * C - V \Rightarrow_G ac * C - c \Rightarrow_G \mathbf{ac*1-c} \end{aligned}$$

nebo

$$\begin{aligned} S &\Rightarrow_G E \Rightarrow_G E * E \Rightarrow_G V * E \Rightarrow_G aV * E \Rightarrow_G ac * E \Rightarrow_G ac * E - E \Rightarrow_G \\ ac * C - E &\Rightarrow_G ac * 1 - E \Rightarrow_G ac * 1 - V \Rightarrow_G \mathbf{ac*1-c} \end{aligned}$$

Ačkoliv odvozujeme stejnou větu, můžeme zde pozorovat jakousi nejednoznačnost, způsobenou tím, že neterminály derivujeme v libovolném pořadí.

Tento problém bychom mohli vyřešit použitím tzv. lineární bezkontextové gramatiky, tedy takové, jejíž odvozovací pravidla obsahují pouze jeden neterminální symbol na pravé straně.

**Příklad 37:** Lineární gramatika může vypadat např. takto:

$$\begin{aligned} G &= \langle N, \Sigma, P, S \rangle \\ P &= \{ \begin{array}{l} S \rightarrow abB, \\ A \rightarrow aaBb | \varepsilon, \\ B \rightarrow bbAa \end{array} \} \\ L(G) &= \{ab(bbaa)^n bba(ba)^n \mid n \geq 0\} \end{aligned}$$

Ovšem k nejednoznačnosti může stejně dojít, pokud by se z různých neterminálních symbolů daly odvodit stejná slova.

**Příklad 38:** Příklad nejednoznačné lineární gramatiky:

$$\begin{aligned} G &= \langle N, \Sigma, P, S \rangle \\ P &= \{ \begin{array}{l} S \rightarrow aA \mid aB, \\ A \rightarrow bA \mid a, \\ B \rightarrow bB \mid a \end{array} \} \end{aligned}$$

Uvažujme slovo  $w = abba$ , ke kterému lze dojít buď:

$$S \Rightarrow_G aA \Rightarrow_G abA \Rightarrow_G abbA \Rightarrow_G abba$$

nebo:

$$S \Rightarrow_G aB \Rightarrow_G abB \Rightarrow_G abbB \Rightarrow_G abba$$

Mějme bezkontextovou gramatiku  $G = \langle N, \Sigma, P, S \rangle$ .

**Definice 25:** P-derivace  $x_0, \dots, x_k$  se nazývá **nejlevější** P-derivace, pokud pro každé  $i \in \{1, \dots, k\}$  platí, že  $x_{i-1}$  je ve tvaru  $uAv$ , kde  $u \in \Sigma^*$ ,  $A \in N$ ,  $v \in (\Sigma \cup N)^*$  a  $x_i$  je ve tvaru  $uyv$ , kde  $A \rightarrow y \in P$ .

**Poznámka 15:** Řetězci  $u$  se říká uzavřená forma a řetězci  $Av$  otevřená forma (větné formy  $uAv$ ).

Odvození pomocí nejlevější derivace značíme  $u \Rightarrow_{G,l} v$ .

**Věta 47:** Mějme  $v \in \Sigma^*$  a  $X \in N$ . Pokud existuje P-derivace  $X, \dots, v$ , pak existuje nejlevější P-derivace  $X, \dots, v$  používající stejnou množinu pravidel jako výchozí P-derivace.

**Důkaz 38:** Tvrzení dokážeme indukcí přes délku P-derivace

- Pro délku 0 platí triviálně.
- Předpokládejme, že tvrzení platí pro P-derivaci délky  $\leq n$ .

Uvažujme P-derivaci délky  $n+1$ , ve tvaru  $x_0, x_1, \dots, x_{n+1}$ . Pokud  $x_1$  vzniklo z  $x_0$  použitím pravidla  $X \rightarrow w_0 X_{i_1} w_1 X_{i_2} \dots X_{i_k} w_k$  kde  $w_0, \dots, w_j \in \Sigma^*$ ,  $X_{i_j} \in N$ ,  $1 \leq j \leq k$  pak  $X_{n+1}$  je ve tvaru  $w_0 u_1 w_1 u_2 w_2 \dots w_k$  tak, že  $x_{i_j} \Rightarrow_G^* u_j$ .

To znamená, že existují P-derivace délek  $\leq n$ :  $X_{i_j}, \dots, u_j$

Z indukčního předpokladu: existují nejlevější P-derivace  $X_{i_j}, \dots, u_j$  používající stejnou množinu pravidel.

Dále platí:

$$\begin{aligned} X &\Rightarrow_{G,l} w_0 X_{i_1} w_1 X_{i_2} \dots X_{i_k} w_k \\ &\Rightarrow_{G,l} w_0 u_1 w_1 X_{i_2} \dots X_{i_k} w_k \\ &\Rightarrow_{G,l} w_0 u_1 w_1 u_2 \dots X_{i_k} w_k \\ &\dots \\ &\Rightarrow_{G,l} w_0 u_1 w_1 u_2 w_2 \dots u_k w_k \end{aligned}$$

To jest, existuje nejlevější P-derivace  $X, \dots, w_0 \dots w_k$

**Příklad 39:** Uvažujme gramatiku z příkladu 36

$S \Rightarrow_G^* E + C$  tady problém není

$S \Rightarrow_{G,l}^* E + C$  tento výraz už smysl nedává. Pomocí nejlevější derivace bychom zákonitě museli nejdříve derivovat  $E$  na levé straně výrazu  $E + E$

**Poznámka 16:** Lze zavést duálně nejpravější derivaci.

**Příklad 40:** Zase se odkážeme na příklad 36.

Výraz  $10 + (ca * 110)$  má jedinou nejlevější derivaci

Naopak  $a + 10 * c$  jich má několik:

$$\begin{aligned} S &\Rightarrow_{G,l} E \Rightarrow_{G,l} E + E \Rightarrow_{G,l} V + E \Rightarrow_{G,l} a + E \Rightarrow_{G,l} a + E * E \Rightarrow_{G,l} a + C * E \Rightarrow_{G,l} \\ &a + 1C * E \Rightarrow_{G,l} a + 10 * E \Rightarrow_{G,l} a + 10 * V \Rightarrow_{G,l} a + 10 * c \end{aligned}$$

nebo:

$$S \Rightarrow_{G,l} E \Rightarrow_{G,l} E * E \Rightarrow_{G,l} \dots \Rightarrow_{G,l} a + 10 * c \text{ (už ve druhém odvození můžeme pozorovat rozdíl)}$$

Je to způsobeno jinou nejednoznačností než tou, kterou jsme eliminovali pomocí nejlevější derivace.

## 20.1. Derivační stromy

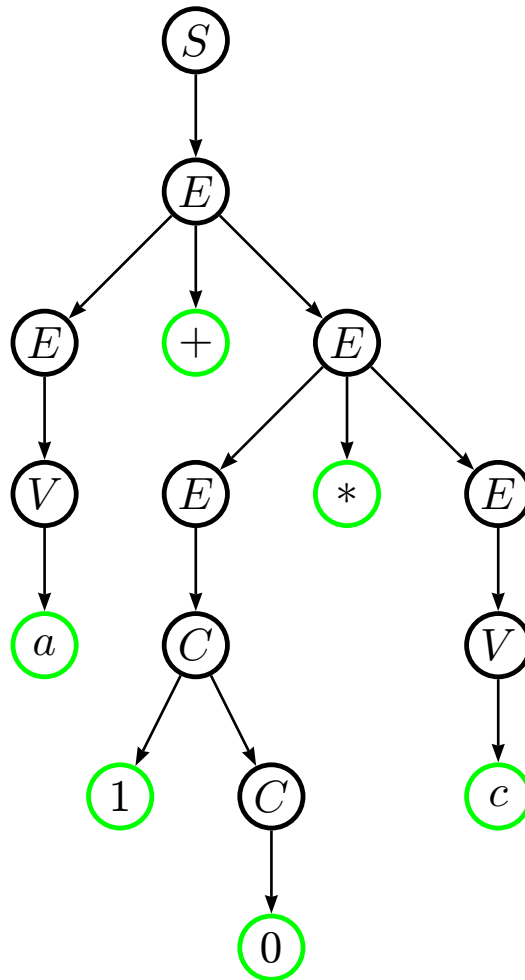
Slouží ke grafickému znázornění nejlevějších derivací. Mějme gramatiku  $G = \langle N, \Sigma, P, S \rangle$ .

**Definice 26:** Derivačním stromem slova  $x \in (\Sigma \cup N)^*$  z  $X \in N$  podle pravidel z  $G$  je každý strom splňující:

1. každý vnitřní uzel je ohodnocen neterminálem
2. kořen je ohodnocen  $X$
3. pokud je vnitřní uzel označený  $A \in N$  a jeho potomci jsou zleva doprava označení  $X_1, \dots, X_k \in N \cup \Sigma$  pak existuje pravidlo  $A \rightarrow X_1 \dots X_k \in P$
4. zřetěžením hodnot v listech při průchodu stromu do hloubky a zleva doprava získáme řetězec  $x$

**Příklad 41:** Opět pracujeme s gramatikou z příkladu 36.

$S \Rightarrow_{G,l}^* a + 10 * c$  Derivační strom bude vypadat následovně:



**Věta 48:** Pokud  $X \Rightarrow_{G,l}^* x$  pak existuje derivační strom  $x$  z  $X$  podle  $G$ .

**Důkaz 39:** Tvrzení dokážeme jak jinak, než indukcí přes délku P-derivace.

Předpokládejme, že tvrzení platí pro derivace délky  $n$  a méně.

Mějme derivaci  $X = x_0, \dots, x_{n+1}$  délky  $n + 1$  pak existuje pravidlo  $x \rightarrow x_1 \in P$

$x_1 = w_0 X_1 w_1 X_2 w_2 \dots X_k w_k$  kde  $w_i \in \Sigma^*$ ,  $X_i \in N$

$x_{n+1}$  je ve tvaru  $w_0 u_1 w_1 u_2 \dots u_k w_k$

$X_i \Rightarrow_{G,l}^* u_i \dots$  délky nejvýše  $n$ . Z indukčního předpokladu vyplývá, že existují derivační stromy  $X_i \Rightarrow_{G,l}^* u_i$ .

**Věta 49:** Pokud existuje derivační strom  $x$  z  $X$  podle  $G$ , pak  $X \Rightarrow_{G,l}^* x$

**Důkaz 40:** Dokážeme indukcí přes výšku stromu.

- Pro 0 je to jasné
- Předpokládejme, že tvrzení platí pro stromy výšky  $n$  a méně.  
Mějme strom výšky  $n + 1$ , podle tohoto víme, jak vypadal první krok derivace  
 $X \rightarrow w_0 X_0 w_1 \dots$   
Z indukčního předpokladu existují derivace  $X_i \Rightarrow_{G,l}^* u_i$  a proto
 
$$\begin{aligned} X &\Rightarrow_{G,l} w_0 X_1 w_1 X_2 \dots X_{i_k} w_k \\ &\Rightarrow_{G,l} w_0 u_1 w_1 X_2 \dots X_{i_k} w_k \\ &\dots \\ &\Rightarrow_{G,l} w_0 u_1 w_1 u_2 w_2 \dots u_k w_k \end{aligned}$$

**Věta 50:**  $S \Rightarrow_{G,l}^* u$  právě když existuje derivační strom  $u$  z  $S$  podle  $G$ .

## 20.2. Jednoznačné a nejednoznačné bezkontextové gramatiky

**Definice 27:** Bezkontextová gramatika  $G = \langle N, \Sigma, P, S \rangle$  se nazývá nejednoznačná, pokud existuje věta  $x \in L(G)$ , která má více než jeden derivační strom z  $S$  podle  $G$ . V opačném případě se nazývá jednoznačná bezkontextová gramatika.

**Definice 28:** Bezkontextový jazyk  $L$  se nazývá jednoznačný pokud existuje jednoznačná gramatika  $G$  tak, že  $L = L(G)$ .

**Definice 29:** Bezkontextový jazyk  $L$  se nazývá **inherentně** nejednoznačný, pokud neexistuje žádná jednoznačná gramatika  $G$  taková, že  $L = L(G)$ .

**Věta 51:** Každý regulární jazyk je jednoznačný.

**Důkaz 41:** Pokud  $L$  je regulární, pak existuje KDA  $A$  s úplnou přechodovou funkcí  $\delta$ . Pak pro  $A$  lze sestavit gramatiku  $G$  tak, že bude platit  $L(A) = L(G)$ .

Z toho jak byla gramatika  $G$  konstruována plyne, že pro každý  $A \in N$  a  $a \in \Sigma$  existuje nejvýše jedno pravidlo  $A \rightarrow aB$ . To jest, pro  $x \in L$  existuje právě jedna P-derivace.

**Důkaz 42:**



**Příklad 42:**  $L = \{a^n b^n c^p d^p \mid n, p \geq 0\}$  je inherentně nejednoznačný.

### 20.3. Uzávěrové vlastnosti bezkontextových jazyků

- **Sjednocení**

Mějme dány bezkontextové jazyky  $L_1, L_2$  a korespondující bezkontextové gramatiky  $G_1$  a  $G_2$ . Lze předpokládat, že množiny stavů obou gramatik jsou disjunktní (tj.  $N_1 \cap N_2 = \emptyset$ ).

Dále uvažujeme gramatiku  $G = \langle N, \Sigma, P, S \rangle$ , kde

$$N = N_1 \cup N_2 \cup \{S\}$$

$$\Sigma = \Sigma_1 \cup \Sigma_2$$

$$P = P_1 \cup P_2 \cup \{S \rightarrow S_1 \mid S_2\}$$

Platí, že  $L(G) = L(G_1) \cup L(G_2)$ .

- **Produkt**

$L_1$  a  $L_2$  jsou bezkontextové jazyky,  $G_1$  a  $G_2$  jsou odpovídající bezkontextové gramatiky.

Zavedeme gramatiku  $G = \langle N, \Sigma, P, S \rangle$ , kde

$$N = N_1 \cup N_2 \cup \{S\}$$

$$\Sigma = \Sigma_1 \cup \Sigma_2$$

$$P = P_1 \cup P_2 \cup \{S \rightarrow S_1 S_2\}$$

Platí, že  $L(G) = L(G_1) \cdot L(G_2)$ .

- **Kleeného uzávěr**

Mějme bezkontextový jazyk  $L$  a k němu odpovídající gramatiku  $G$ .

Vytvoříme novou gramatiku  $G' = \langle N \cup \{S'\}, \Sigma, P \cup \{S' \rightarrow \varepsilon \mid SS'\}, S' \rangle$ .

- **Pozitivní uzávěr**

Podobné jako u Kleeného uzávěru, akorát výsledná gramatika se bude lišit v množině pravidel.

$$G' = \langle N \cup \{S'\}, \Sigma, P \cup \{S' \rightarrow S \mid SS'\}, S' \rangle$$

- **Průnik**

$\mathcal{L}_2$  není uzavřená na průnik.

**Příklad 43:** Uvedeme si příklad, který nám tvrzení potvrdí.

$$\Sigma = \{a, b, c\}$$

$$L_1 = \{a^n b^n c^* \mid n \geq 0\}$$

$$L_2 = \{a^* b^n c^n \mid n \geq 0\}$$

$$L_1 \cap L_2 = \{a^n b^n c^n \mid n \geq 0\}$$

Z toho také plyne, že není uzavřená ani na Komplement či Množinový rozdíl (De Morganovy zákony).

### 20.4. Bezkontextové gramatiky v programátorské praxi

**BNF**(Backus-Naurova forma): Vytvořili ji John Backus a Peter Naur. Zápis se řídí několika pravidly:

- neterminální symboly se zapisují do  $\langle \dots \rangle$
- používá se  $::=$  místo  $\rightarrow$
- terminální symboly se píší do uvozovek (" \* ")
- pravidla se oddělují pomocí |

**Příklad 44:** Uvedeme si příklad zápisu gramatiky pomocí BNF.

$\langle expr \rangle ::= "(" \langle expr \rangle ")" \mid \langle expr \rangle \langle op \rangle \langle expr \rangle \mid \langle number \rangle$   
 $\langle op \rangle ::= "+" \mid "-" \mid "*" \mid "/"$   
 $\langle number \rangle ::= \langle digit \rangle \langle number \rangle \mid \langle digit \rangle$   
 $\langle digit \rangle ::= "0" \mid \dots \mid "9"$

**Extended BNF:** Zavedl Niklaus Wirth v roce 1977. Odpovídá BNF, pouze zjednodušuje její notaci a to tak že:

- každé pravidlo je ukončeno znakem ";"
- terminální symboly se píší do uvozovek, neterminální ne
- $[, ]$  značí volitelnou část výrazu
- $\{, \}$  indukují možnost opakování
- $(, )$  jsou použity pro shlukování výrazů
- místo  $::=$  se používá  $=$
- terminální a neterminální symboly se oddělují čárkou

**Příklad 45:** Gramatiku z předchozího příkladu můžeme zapsat pomocí EBNF takto:

$expr = "(" , expr , ")" \mid expr , op , expr \mid number ;$   
 $op = "+" \mid "-" \mid "*" \mid "/" ;$   
 $number = [ signum ] , digit , \{ digit \} ;$   
 $digit = "0" \mid \dots \mid "9" ;$   
 $signum = "+" \mid "-" ;$

## 21. Nedeterministický zásobníkový automat

Hledáme silnější analytický aparát, než jsou KDA, protože ty v určitých situacích selhávají.

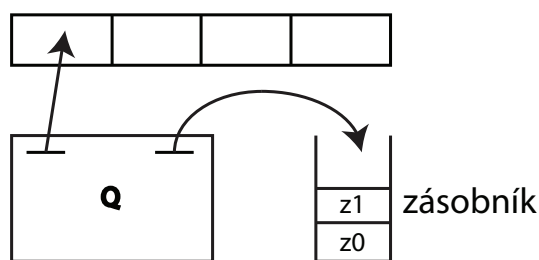
**Příklad 46:** Mějme jazyk

$$L = \{x \in \Sigma^* \mid x \text{ je korektně uzávorkovaný výraz}\}$$

Následně podrobněji

$$L = \{x \in \Sigma^* \mid \Sigma = \{ (, ), [, ] \} \quad x \text{ je korektně uzávorkovaný výraz} \}$$

Do tohoto jazyka patří například řetězce  $()$ ,  $[], ([ ]), \dots$ , ale nepatří do něj řetězce  $()$ ,  $[[[$ ,  $\dots$



Obrázek 7. Náčrt zásobníkového automatu.

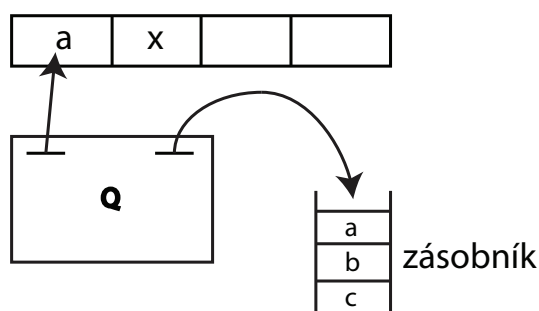
**Definice 30:** Jako **nedeterministický zásobníkový automat** označme následující strukturu:

$$A = \langle \Sigma, \Gamma, Q, \delta, q_0, z_0, F \rangle$$

Nyní si popíšeme význam nám dosud neznámých prvků tohoto automatu:

- $\Gamma$  — abeceda zásobníkových symbolů.
- $z_0$  — počáteční zásobníkový symbol.
- $Q$  — konečná množina stavů.
- $\delta$  — přechodová funkce ve tvaru  $\delta : Q \times (\Sigma \cup \{\varepsilon\}) \times \Gamma \rightarrow 2^{Q \times \Gamma^*}$

Navíc  $\langle \sigma, abc \rangle \in \delta(q, a, d)$  znamená slovně: „Ze stavu  $q$  po přečtení  $a$  ze vstupu a symbolu  $d$  ze zásobníku přejde automat  $A$  do stavu  $\sigma$  a na zásobník запиše řetězec „ $abc$ “.“ Mějme na paměti, že tento řetězec je na zásobník zapsán „obráceně“.



Obrázek 8. Vkládání hodnot na zásobník u NZA.

## 21.1. Reprezentace NZA

### 21.1.1. Přechodová tabulka

V přechodové tabulce řádky odpovídají stavům a sloupce odpovídají prvkům  $(\Sigma \cup \{\varepsilon\}) \times \Gamma$ .

**Příklad 47:**

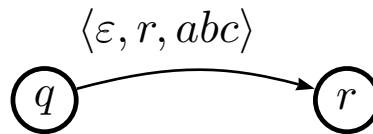
$$\begin{aligned} Q &= \{q_0, q_1\} \\ \Sigma &= \{a, b\} \\ \Gamma &= \{0, 1\} \end{aligned}$$

	$\langle a, 0 \rangle$	$\langle b, 0 \rangle$	$\langle c, 0 \rangle$	$\langle a, 1 \rangle$	$\langle b, 1 \rangle$	$\langle c, 1 \rangle$
$\rightarrow q_0$	$\dots$	$\dots$	$\dots$	$\dots$	$\dots$	$\dots$
$q_1$	$\dots$	$\{\langle r, w \rangle\}$	$\dots$	$\dots$	$\dots$	$\dots$

Tabulka 5. Ukázka přechodové tabulky pro NZA

**21.1.2. Přechodový diagram**

Přechodové diagramy NZA jsou velmi podobné diagramům KA.



**Definice 31:** Konfigurací NZA je každý prvek z množiny  $Q \times \Sigma^* \times \Gamma^*$ , tedy  $\langle q, abc, 001 \rangle \in Q \times \Sigma^* \times \Gamma^*$  znamená, že NZA je ve stavu  $q$ , na vstupu zbývá přečíst řetězec „abc“ a na zásobníku je řetězec „001“.

Pro konfigurace  $\langle q_i, w_i, x_i \rangle$  a  $\langle q_j, w_j, x_j \rangle$  klademe  $\langle q_i, w_i, x_i \rangle \vdash \langle q_j, w_j, x_j \rangle$  právě, když platí:

- $x_i = zy$ , kde  $z \in \Gamma, y \in \Gamma^*$
- $w_i = aw_j$ , kde  $a \in (\Sigma \cup \{\varepsilon\}), w_j \in \Sigma^*$
- $\langle q_j, x \rangle \in \delta(q_i, a, z)$

Přičemž  $x_j = xy$ .

Reflexivní a tranzitivní uzávěr  $\vdash$  označíme  $\vdash^*$  a platí:

$$\begin{aligned} \langle q_i, w_i, x_i \rangle \vdash^* \langle q_j, w_j, x_j \rangle &\text{ právě, když} \\ \langle q_i, w_i, x_i \rangle \vdash \dots \vdash \langle q_j, w_j, x_j \rangle \end{aligned}$$

Pokud  $\langle q_i, w_i, x_i \rangle \vdash^* \langle q_j, w_j, x_j \rangle$  tak říkáme, že existuje výpočet NZA, kterým přejde tento NZA z  $\langle q_i, w_i, x_i \rangle$  do  $\langle q_j, w_j, x_j \rangle$ .

**21.2. Jazyky rozpoznávané NZA**

1. Jazyk rozpoznávaný pomocí koncových stavů automatu, tedy:

$$L(A) = \{w \in \Sigma^* \mid \langle q_0, w, z_0 \rangle \vdash^* \langle q, \varepsilon, x \rangle, q \in F, x \in \Gamma^*\}$$

2. Jazyk rozpoznávaný vyprazdňováním zásobníků, tedy:

$$N(A) = \{w \in \Sigma^* \mid \langle q_0, w, z_0 \rangle \vdash^* \langle q, \varepsilon, \varepsilon \rangle, q \in Q\}$$

### 21.3. Rozpoznání jazyk způsobem přechodu od koncových stavů k vyprázdnění zásobníku

**Věta 52:** Ke každému NZA  $A$  existuje NZA  $A'$ , pro který platí  $L(A) = N(A') = L(A')$

**Důkaz 43:** Mějme automat  $A = \langle \Sigma, \Gamma, Q, \delta, q_0, z_0, F \rangle$ . Zkonstruujeme automat  $A'$  tak, že platí  $L(A) = N(A') = L(A')$ , načež je třeba vyřešit dvě věci:

1. V případě, že automat  $A$  vyprázdní svůj zásobník, ale v koncovém stavu, potom  $A'$  nesmí vyprázdnit zásobník, což se řeší přidáním symbolu  $z'_0$  na dno zásobníku automatu  $A$ .
2. V případě, že automat  $A$  skončí (s prázdným vstupem), tak by v koncovém stavu automat  $A'$  měl navíc ještě vyprázdnit zásobník, to ale zajistíme tak, že přidáme slovo  $q_\#$ , ve kterém  $A'$  vyprazdňuje svůj zásobník.

Uvažujme

$$A' = \langle \Sigma, \Gamma \cup \{z'_0\}, Q \cup \{q'_0, q'_\#\}, \delta', q'_0, z'_0, \{q'_\#\} \rangle$$

kde  $\delta'$  je přechodová funkce, jež vznikne rozšířením původní přechodové funkce o následující přechody:

$$\begin{aligned} \delta'(q'_0, \varepsilon, z'_0) &= \{\langle q_0, z_0, z'_0 \rangle\} \\ \delta'(q, \varepsilon, z) &= \delta(q, \varepsilon, z) \cup \{\langle q'_\#, \varepsilon \rangle\} \text{ pokud } q \in F, a \in \Gamma \cup \{z'_0\} \\ \delta'(q'_\#, \varepsilon, z) &= \{\langle q'_\#, \varepsilon \rangle\} \text{ pro } z \in \Gamma \cup \{z'_0\} \end{aligned}$$

### 21.4. Rozpoznání jazyk způsobem přechodu od vyprázdnění zásobníku k přijímání koncovými stavy

**Věta 53:** Ke každému NZA  $A = \langle \Sigma, \Gamma, Q, \delta, q_0, z_0, F \rangle$  existuje NZA  $A'$  tak, že  $N(A) = L(A') = N(A')$ .

**Důkaz 44:** Některé je dán  $A = \langle \text{nemám poznačeno} \rangle$ . Konstruujeme  $A'$ , načež je třeba vyřešit dvě věci:

1. V případě, kdy výchozí automat  $A$  vyprázdní vstup a zásobník (to může nastat v libovolném stavu), pak přejdeme do koncového stavu. To řešíme přidáním nového zásobníkového symbolu  $z'_0$ .
2. Je potřeba zamezit přijetí slova koncovým stavem bez vyprázdnění zásobníku. To řešíme tak, že máme jediný koncový stav, do kterého přejdeme jen pokud máme prázdný zásobník automatu  $A$ .

$$\begin{aligned} A' &= \langle \Sigma, \Gamma \cup \{z'_0\}, Q \cup \{q'_\#, q'_0\}, \delta', q'_0, z'_0, \{q'_\#\} \rangle \\ \delta' &\text{ je navíc rozšířením } \delta \\ \delta'(q'_0, \varepsilon, z'_0) &= \{\langle q_0, z_0, z'_0 \rangle\} \\ \delta'(q, \varepsilon, z'_0) &= \{\langle q'_\#, \varepsilon \rangle\}, \quad \forall q \in Q \end{aligned}$$

### 21.5. Automaty pracující s celým zásobníkem

Pro automat pracující s celým zásobníkem má platit, že  $\delta : Q \times (\Sigma \cup \{\varepsilon\}) \times \Gamma^* \rightarrow 2^{Q \times \Gamma^*}$  s omezením, že přechodová funkce je pro každé  $q \in Q, a \in (\Sigma \cup \{\varepsilon\})$  definována pouze pro konečně mnoho řetězců  $z \in \Gamma^*$  a navíc musí platit, že pro každý z konečně mnoha řetězců je  $\delta(q, a, z)$  konečná.

Platí také:

$$\langle q_i, w_i, z_i \rangle \vdash \langle q_j, w_j, z_j \rangle$$

1.  $z_i = zy, z \in \Gamma^*, y \in \Gamma^*$
2.  $w_i = aw_j, a \in (\Sigma \cup \{\varepsilon\})$
3.  $\langle q_j, x \rangle \in \delta(q_i, a, z)$
4.  $z_j = xy$

**Věta 54:** Pro každý NZA  $A$  pracující s celým zásobníkem existuje NZA  $A'$  pracující pouze s vrcholem zásobníku.

**Důkaz 45:** Mějme  $A = \langle \Sigma, \Gamma, Q, \delta, q_0, z_0, F \rangle$  pracující s celým zásobníkem. Následně vytvoříme  $A' = \langle \Sigma, \Gamma \cup \{z'_0\}, Q \cup \{q'_0, q_\#\} \cup Q', q'_0, z'_0, F \rangle$  pracující s vrcholem zásobníku.

- $q_\#$  je nový nekonečný stav, ve kterém se vyprázdní zásobník automatu.
- $Q'$  je pomocná množina stavů, které potřebujeme pro nahrazení pravidel, která pracují s více zásobníkovými symboly.
- $\delta'$  je definována následovně:
  - Na počátku je prázdná.
  - Přidáme přechody  $\delta'(q'_0, \varepsilon, z'_0) = \{\langle q_0, z_0, z'_0 \rangle\}$ .
    1. Pokud  $\langle r, y \rangle \in \delta(q, a, \varepsilon)$  pak pro každé  $z \in \Gamma \cup \{z'_0\}$  přidáme  $\langle r, yz \rangle$  do  $\delta'(q, a, z)$ .
    2. Pokud  $\langle r, y \rangle \in \delta(q, a, z), z \in \Gamma$ , pak přidáme přechod  $\langle r, y \rangle \in \delta'(q, a, z)$ .
    3. Pokud  $\langle r, y \rangle \in \delta(q, a, z), z = z_1 z_2 z_3 \dots z_n$  pro  $n > 1$ , v tomto případě zavedeme dosud neuvažované stavy  $q'_1, q'_2, \dots, q'_{n-1}$  a ty přidáme do  $Q'$  a zavedeme přechody  $\delta'(q, a, z_1) = \{\langle q'_1, \varepsilon, \rangle\}, \delta(q'_1, \varepsilon) = \{\langle q'_2, \varepsilon \rangle\} = \dots = \delta'(q'_{n-1}, \varepsilon, z_n) = \{\langle q_\#, \varepsilon \rangle\}$
- Zavedeme přechod  $\delta'(q, \varepsilon, z'_0) = \{\langle q_\#, \varepsilon \rangle\}$ .

**Důsledek 4:** Třídy automatu jsou ekvivalentní.

### 21.6. Poznámky o vlastnostech NZA

1. Pokud  $\langle q, u, x \rangle \vdash^* \langle p, v, y \rangle$  pak  $\langle q, uw, x \rangle \vdash^* \langle p, vw, y \rangle$  pro libovolný řetězec  $w \in \Sigma^*$ .
2. Pokud  $\langle q, u, x \rangle \vdash^* \langle p, v, y \rangle$  pak  $\langle q, u, xz \rangle \vdash^* \langle p, v, yz \rangle$  pro libovolný řetězec  $w \in \Gamma^*$ .

3. Pokud  $\langle q, uw, x \rangle \vdash^* \langle p, vw, y \rangle$  pak  $\langle q, u, x \rangle \vdash^* \langle p, v, y \rangle$ .
- 1.+2. Pokud  $\langle q, u, x \rangle \vdash^* \langle p, v, y \rangle$  pak  $\langle q, uw, xz \rangle \vdash^* \langle p, vw, yz \rangle$  pro všechny řetězce  $w \in \Sigma^*$  a  $z \in \Gamma^*$ .
- 1.+3.  $\langle q, u, x \rangle \vdash^* \langle p, v, y \rangle$  právě když  $\langle q, uw, x \rangle \vdash^* \langle p, vw, y \rangle$ .

**Poznámka 17:** Opak 2. vlastnosti neplatí: Pokud  $\langle q, u, xz \rangle \vdash^* \langle p, v, yz \rangle$  pak obecně nelze odvodit, že  $\langle q, u, x \rangle \vdash^* \langle p, v, y \rangle$ .  
Speciálně:  $x = y = \varepsilon$

**Poznámka 18:**  $\langle q, u, z \rangle \vdash^* \langle p, \varepsilon, \varepsilon \rangle$ : automat je schopen přejít ze stavu  $q$  do  $p$ , přitom zkonzumuje řetězec  $u$  ze vstupu a řetězec  $z$  ze zásobníku, ale bez toho, aniž by se někdy během činnosti dostal pod úroveň řetězce  $z$  na zásobníku.

## 21.7. Od gramatik k automatům

Prokážeme, že ke každé bezkontextové gramatice existuje NZA  $A$  tak, že  $L(G) = N(A)$ .

Uvažujme bezkontextovou gramatiku  $G = \langle N, \Sigma, P, S \rangle$ . Zkonstruujeme zásobníkový automat  $A = \langle \Sigma, N \cup \Sigma, \{q\}, \delta, q, S, \emptyset \rangle$ .

- Zásobníkové symboly se skládají z terminálních a neterminálních symbolů gramatiky
- Množina stavů je jednoprvková
- Množina koncových stavů je prázdná
- $S$  je počáteční zásobníkový symbol
- $\delta$  simuluje provádění nejlevějších derivací

$$\delta(q, a, z) = \begin{cases} \{\langle q, \varepsilon \rangle\} & \text{pokud } a \in \Sigma \text{ a } z = a \\ \{\langle q, y \rangle \mid z \rightarrow y \in P\} & \text{pokud } a \in \Sigma \text{ a } z \in N \\ \emptyset & \text{jinak} \end{cases}$$

- Prvnímu případu říkáme **operace srovnání**: pokud je na vstupu i na vrcholu zásobníku stejný znak (terminál) pak jsou oba odebrány.
- Druhý případ se jmenuje **operace expanze**: bez čtení vstupu při neterminálu na vrcholu zásobníku se tento neterminál nahradí pravou stranou některého pravidla s tímto neterminálem na levé straně.
- U třetího případu dochází k zastavení činnosti.

**Příklad 48:** Uvedeme si gramatiku a na ní ukážeme činnost takového automatu:

$$G = \langle N, \Sigma, P, S \rangle$$

$$\Sigma = \{a, b, c, 0, 1, +, -, *, /, (, )\}$$

$$N = \{S, E, C, V\}$$

$$P = \left\{ \begin{array}{l} S \rightarrow E, \\ E \rightarrow E * E \mid E / E \mid E + E \mid E - E \mid - E \mid C \mid V \mid (E), \\ C \rightarrow 0C \mid 1C \mid 0 \mid 1, \\ V \rightarrow aV \mid bV \mid cV \mid a \mid b \mid c \end{array} \right\}$$

Řetězec na vstupu    Stav zásobníku

$10 + (ca * 110)$	$S$
$10 + (ca * 110)$	$E$
$10 + (ca * 110)$	$E + E$
$10 + (ca * 110)$	$C + E$
$10 + (ca * 110)$	$1C + E$
$0 + (ca * 110)$	$C + E$
$0 + (ca * 110)$	$0 + E$
$+(ca * 110)$	$+E$
$(ca * 110)$	$E$
$(ca * 110)$	$(E)$
$ca * 110)$	$E)$
$ca * 110)$	$E * E)$
$ca * 110)$	$V * E)$
$ca * 110)$	$cV * E)$
$a * 110)$	$V * E)$
$a * 110)$	$a * E)$
$*110)$	$*E)$
$110)$	$E)$
$110)$	$C)$
$110)$	$1C)$
$10)$	$C)$
$10)$	$1C)$
$0)$	$C)$
$0)$	$0)$
$)$	$)$
$\varepsilon$	$\varepsilon$

**Věta 55:** Pro každou bezkontextovou gramatiku  $G$  existuje NZA  $A$  tak, že  $L(G) = N(A)$ .

**Důkaz 46:** Uvažujeme gramatiku  $G = \langle N, \Sigma, P, S \rangle$  a odtud  $A$  zkonstruovaný pomocí operací expanze a srovnání (předchozí způsob).

Tvrdíme, že pro libovolný řetězec  $w \in \Sigma^*$  platí:

$$S \Rightarrow_{G,l}^* w \text{ právě když } \langle q, w, S \rangle \vdash^* \langle q, \varepsilon, \varepsilon \rangle$$

• **Implikace zleva:**

Nechť  $S \Rightarrow_{G,l}^* w$ , to znamená, že existuje nejlevější derivace  $S = X_0, \dots, X_n = w$ . Indukcí pro  $i = 0, \dots, n$  prokážeme, že  $\langle q, w, S \rangle \vdash^* \langle q, u_i, z_i \rangle$  pro  $w = y_i \cdot u_i$ .  $X_i = y_i z_i$ , kde buď

1.  $z_i = \varepsilon$   
nebo
2.  $z_i = Av_i$ , kde  $A \in N$

Pro  $i = 0$  je tvrzení zřejmé, protože  $x_0 = S$  a můžeme položit  $y_0 = \varepsilon$ ,  $u_i = w$  a  $z_0 = S$ .

Nechť platí tvrzení pro  $i$ . Prokážeme jej pro  $i + 1$ .

Z indukčního předpokladu  $\langle q, w, S \rangle \vdash^* \langle q, u_i, z_i \rangle$  pro  $w = y_i u_i$  a  $x_i = y_i z_i$ . Dále musí platit, že  $z_i = Av_i$ , jinak by byl  $x_i$  ze samých terminálních symbolů a posloupnost  $X_1, X_2, \dots, X_i, X_{i+1}, \dots$  by nemohla být P-derivace.



Dále z předpokladu  $X_i \Rightarrow_{G,l} X_{i+1}$  to jest z indukčního předpokladu  $x_i = y_i z_i = y_i A v_i$ ,  $A$  je nejlevější neterminál, tj. existuje pravidlo  $A \rightarrow w_i \in P$  tak, že  $x_{i+1} = y_i w_i v_i$ .

NZA  $A$  může přejít z konfigurace  $\langle q, u_i, z_i \rangle = \langle q, u_i, A v_i \rangle$  do konfigurace  $\langle q, u_i, w_i v_i \rangle$  (pomocí expanze). Dále může NZA přecházet do dalších konfigurací postupnou aplikací operace srovnání, dokud není vyprázdněn zásobník nebo není na vrcholu neterminál. Automat nakonec skončí v konfiguraci  $\langle q, u_{i+1}, z_{i+1} \rangle$ , kde buď  $z_{i+1} = \varepsilon$  nebo  $z_{i+1} = B v_{i+1}$  kde  $B \in N$ .

Zřejmě  $w = y_{i+1} u_{i+1}$  kde  $x_{i+1} = y_i w_i v_i = y_{i+1} z_{i+1}$ .

Ve speciálním případě:  $i = n$  dostáváme  $\langle q, w, S \rangle \vdash^* \langle q, u_n, z_n \rangle$ , kde  $w = y_n u_n$ ,  $X_n = y_n z_n = w$  tj. dle 1. vlastnosti, protože  $z_n$  je celý z terminálních symbolů, platí, že  $z_n = \varepsilon$ , to jest  $x_n = y_n z_n = y_n = w$  ale  $w = y_n u_n$ , odtud  $u_n = \varepsilon$ .

Takže  $\langle q, w, S \rangle \vdash^* \langle q, \varepsilon, \varepsilon \rangle$ .

- **Implikace zprava:**

Prokážeme obecnější tvrzení:

Pro každý neterminál  $A \in N$  platí: pokud  $\langle q, w, A \rangle \vdash^* \langle \varepsilon, \varepsilon \rangle$  pak  $A \Rightarrow_{G,l}^* w$ .

Požadované tvrzení získáme jako důsledek pro  $S \in N$ .

Tvrzení se dokazuje přes délku výpočtu (pro  $n = 0$  nelze).

Pro  $n = 1$  je triviální, protože  $\langle q, w, A \rangle \vdash \langle q, \varepsilon, \varepsilon \rangle$  pak  $w = \varepsilon$  a  $A \rightarrow \varepsilon \in P$ .

Předpokládejme, že tvrzení platí pro libovolný výpočet nejvýše  $u$  kroků dlouhý a předpokládejme, že výpočet, který se dostane z  $\langle q, w, A \rangle$  do  $\langle q, \varepsilon, \varepsilon \rangle$  má délku  $n + 1$ .

První přechod v tomto výpočtu musí nahradit  $A$  na vrcholu zásobníku pravou stranou některého pravidla s  $A$  na levé straně. Musí tedy existovat  $A \rightarrow z_1 z_2 \dots z_k \in P$  a  $\langle q, w, A \rangle \vdash \langle q, w, z_1 z_2 \dots z_k \rangle$  je první krok výpočtu. Zbývajících  $n$  kroků výpočtu musí odstranit  $z_1 z_2 \dots z_k$  ze zásobníku.

Řetězec  $w$  je ve tvaru  $w = x_1 x_2 \dots x_i$ , kde každá část  $x_i$  odpovídá podřetězci řetězce  $w$ , který byl odstraněn ze vstupu po odstranění  $z_{i-1}$  z vrcholu zásobníku až do chvíle, kdy byl odstraněn  $z_i$  z vrcholu zásobníku.

Formálně:  $\langle q, x_i, z_i \rangle \vdash^* \langle q, \varepsilon, \varepsilon \rangle$

Jelikož výpočet nezávisí na části řetězce, který nebyl dosud zkoumán, platí

$$\langle q, x_i x_{i+1} \dots x_k, z_i \rangle \vdash^* \langle q, x_{i+1} \dots x_k, \varepsilon \rangle$$

Každý z příslušných výpočtů má délku nejvýše  $n$ . Pokud je tedy  $z_i$  neterminál, aplikujeme indukční předpoklad  $z_i \Rightarrow_{G,l}^* x_i$ .

To znamená:

$$A \Rightarrow_{G,l} z_1 z_2 \dots z_k \Rightarrow_{G,l}^* x_1 z_2 \dots z_k \Rightarrow_{G,l}^* x_1 x_2 z_3 \dots z_k \Rightarrow_{G,l}^* x_1 \dots x_k = w$$

## 21.8. Od automatů ke gramatikám

**Věta 56:** Pro každý NZA  $A = \langle \Sigma, \Gamma, Q, \delta, q_0, z_0, F \rangle$  existuje bezkontextová gramatika  $G$  tak, že  $L(G) = N(A)$ .

**Důkaz 47:** Množina  $N$  je  $N = Q \times \Gamma \times Q \cup \{S\}$ . Pravidla gramatiky vypadají následovně:

1. Pro každý stav  $p \in Q$  obsahuje pravidlo  $S \rightarrow \langle q_0, z_0, p \rangle$
2. Pokud  $\langle u, z_1 z_2 \dots z_k \rangle \in \delta(q, a, z)$ , kde  $a \in \Sigma \cup \{S\}$ , pak pro libovolnou posloupnost stavů  $r_1, \dots, r_n = p \in Q$  platí, že  $P$  obsahuje pravidlo

$$\langle q, z, p \rangle \rightarrow a \langle r, z_1, u_1 \rangle \langle r_1, z_2, u_2 \rangle \dots \langle r_{k-1}, z_k, u_k \rangle$$

Speciálně pokud  $\langle r, \varepsilon \rangle \in \delta(q, a, z)$  pak  $P$  obsahuje pravidlo  $\langle q, z, r \rangle \rightarrow a$

Budeme se snažit dokázat tvrzení  $\langle q, z, p \rangle \Rightarrow_{G,l}^* w$  právě když  $\langle q, w, z \rangle \vdash^* \langle p, \varepsilon, \varepsilon \rangle$

#### • Implikace zleva

Důkaz provedeme indukcí přes délku nejlevější derivace. předpokládejme, že  $\langle q, z, p \rangle \Rightarrow_{G,l}^* w$

$\langle q, z, p \rangle \Rightarrow_{G,l} w$ . Pak existuje  $\langle q, z, p \rangle \rightarrow w$ , tj. pravidlo je speciálním případem pravidla z bodu 2. To jest  $w \in \Sigma \cup \{\varepsilon\}$  a  $\langle p, \varepsilon \rangle \in \delta(q, w, z)$  - platí z toho, jak jsme zavedli pravidla. Potom ale  $\langle q, w, z \rangle \vdash \langle p, \varepsilon, \varepsilon \rangle$ .

Předpokládejme, že nejlevější derivace  $w$  z  $\langle q, z, p \rangle$  má délku  $n > 1$  a tvrzení platí pro všechny derivace kratší délky.

Rozepsáním  $\langle q, z, p \rangle \Rightarrow_{G,l}^* w$  dostaneme, že  $\langle q, z, p \rangle \Rightarrow_{G,l} a \langle r_0, z_1, r_1 \rangle \dots \langle r_{k-1}, z_k, r_k \rangle \Rightarrow_{G,l}^* w$  kde  $r_1, \dots, r_k \in Q$ ,  $r_k = p$  a  $\langle q, z, p \rangle \rightarrow a \langle r_0, z_1, r_1 \rangle \dots \langle r_{k-1}, z_k, r_k \rangle$  je pravidlo dle bodu 2.

Dle bodu 2, musí platit, že  $\langle r_0, z_1 \dots z_k \rangle \in \delta(p, x, z)$  a  $w = a \cdot x$ , kde  $x \in \Sigma^*$  a  $a \in \Sigma \cup \{\varepsilon\}$ . Jelikož  $\langle r_i, z_{i+1}, r_{i+1} \rangle$  jsou všechno neterminály gramatiky.

Z vlastností bezkontextových gramatik platí, že  $x$  lze vyjádřit jako  $x = w_1 w_2 \dots w_k$  tak, že  $\langle r_{i-1}, z_i, r_i \rangle \Rightarrow_{G,l}^* w_i$ .

Každá z těchto nejlevějších derivací má délku nejvýše  $n$ . Na každou z nich aplikujeme indukční předpoklad, tj.:  $\langle r_{i-1}, w_i, z_i \rangle \vdash^* \langle r_i, \varepsilon, \varepsilon \rangle$ .

Nyní můžeme zřetězit posloupnosti odpovídajících konfigurací:

$$\begin{aligned} \langle q, w, z \rangle &= \langle q, ax, z \rangle \vdash \langle r_0, x, z_1 z_2 \dots z_k \rangle \vdash^* \langle r_1, w_2 \dots w_k, z_2 \dots z_k \rangle \vdash^* \dots \vdash^* \langle r_k, \varepsilon, \varepsilon \rangle \\ &= \langle p, \varepsilon, \varepsilon \rangle \end{aligned}$$

což jsme měli dokázat.

#### • Implikace zprava

Dokážeme indukcí přes délku výpočtu automatu  $A$ .

Předpokládejme, že platí  $\langle q, w, z \rangle \vdash^* \langle p, \varepsilon, \varepsilon \rangle$ . Výpočet má alespoň jeden krok, protože je třeba vyprázdnit zásobník. V tomto případě  $w \in \Sigma \cup \{\varepsilon\}$ .

Dále musí  $\langle p, \varepsilon \rangle \in \delta(q, w, z)$ . Ze speciálního případu bodu 2, pak máme, že  $\langle q, z, p \rangle \rightarrow w$  je jedno z pravidel gramatiky. použitím tohoto pravidla  $\langle q, z, p \rangle \Rightarrow_{G,l}^* w$ .

Předpokládejme, že výpočet  $\langle q, w, z \rangle \vdash^* \langle q, \varepsilon, \varepsilon \rangle$  má délku  $n > 1$  a tvrzení platí pro všechny výpočty ostře kratších délek.

První krok výpočtu  $\langle q, w, z \rangle \vdash \langle r_0, x, z_1 \dots z_k \rangle \vdash^* \langle p, \varepsilon, \varepsilon \rangle$ , kde  $w$  lze psát jako  $w = a \cdot x$ , kde  $a \in \Sigma \cup \{\varepsilon\}$  a platí, že  $\langle r_0, z_1 \dots z_k \rangle \in \delta(q, a, z)$

Z bodu 2, definujícího odvozovací pravidla gramatiky, máme, že existují pravidla ve tvarech  $\langle q, z, p \rangle \rightarrow a \langle r_0, z_1, r_1 \rangle \dots \langle r_{k-1}, z_k, r_k \rangle$  kde  $r_1, \dots, r_{k-1}$  jsou libovolné stavy z  $Q$  a  $r_k = p$ .

Jelikož  $\langle r_0, x, z_1 \dots z_k \rangle \vdash^* \langle p, \varepsilon, \varepsilon \rangle$  pak můžeme  $x$  psát ve tvaru  $x = w_1 w_2 \dots w_k$  a existují stavy  $r_1, \dots, r_{k-1}$  tak, že  $\langle r_{i-1}, w_i, z_i \rangle \vdash^* \langle r_i, \varepsilon, \varepsilon \rangle$

Dle indukčního předpokladu:

$$\begin{aligned} \langle r_{i-1}, z_i, r_i \rangle &\Rightarrow_{G,l}^* w_i \\ \langle q, z, p \rangle &\Rightarrow_{G,l} a \langle r_0, z_1, r_1 \rangle \cdots \langle r_{k-1}, z_k, r_k \rangle \end{aligned}$$

**Důsledek 5:** Ke každému NZA existuje ekvivalentní automat rozpoznávající stejný jazyk, který má pouze 1 stav.

## 22. Deterministické zásobníkové automaty

**Definice 32:** Řekneme, že zásobníkový automat  $A = \langle \Sigma, \Gamma, Q, \delta, q_0, z_0, F \rangle$  je **deterministický** (DZA), pokud jsou pro každé  $q \in Q$  a  $z \in \Gamma$  splněny následující podmínky:

1. pro všechny  $a \in \Sigma \cup \{\varepsilon\}$  platí  $|\delta(q, a, z)| \leq 1$
2. pokud  $\delta(q, \varepsilon, z) \neq \emptyset$  pak  $\delta(q, a, z) = \emptyset$  pro každé  $a \in \Sigma$

Nabízí se otázka, jaké třídy jazyků, jsou vlastně rozpoznatelné deterministickým zásobníkovým automatem pomocí buď koncových stavů, nebo vyprázdněním zásobníku.

**Definice 33:** Bezkontextový jazyk  $L$  se nazývá **deterministický**, pokud existuje deterministický zásobníkový automat  $A$  tak, že  $L = L(A)$ .

**Definice 34:** Bezkontextový jazyk  $L$  má **prefixovou vlastnost**, pokud  $L$  neobsahuje dva různé řetězce  $x, y$  tak, že  $x$  je prefixem  $y$ .

**Věta 57:** Pro libovolný bezkontextový jazyk  $L$  jsou následující tvrzení ekvivalentní:

1.  $L = N(A)$  pro nějaký DZA  $A$
2.  $L = L(A')$  pro nějaký DZA  $A'$  a  $L$  má prefixovou vlastnost

**Důkaz 48:** Nechť platí 1. tvrzení:  $L = N(A)$  pro DZA  $A$

1. pozorování:  $L$  má prefixovou vlastnost  
 $x \in L = N(A)$  tj.  $x$  je přijat vyprázdněním zásobníku. Zjišťujeme, že  $\langle q_0, x, z_0 \rangle \vdash^* \langle p, \varepsilon, \varepsilon \rangle$  je **jednoznačně** daný výpočet.
2. pozorování:  
Hledaný automat  $A'$  můžeme najít tak, že vytvoříme nový pomocný zásobníkový symbol  $z'_0$ , který umístíme na počátku činnosti na dno zásobníku. Pak rozšíříme automat o přechody  $\delta(p, \Sigma, z'_0) = \{\langle p\#, \varepsilon \rangle\}$ .  $p\#$  je nový a zároveň jediný koncový stav automatu  $A'$ .

Nechť platí 2. tvrzení.

Jelikož má  $L$  prefixovou vlastnost, pak automat  $A'$  můžeme upravit tak, že z každého koncového stavu zrušíme všechny vycházející přechody. Vzniklý automat opět přijímá  $L$  pomocí koncových stavů. Z koncových stavů vytvoříme  $\varepsilon$ -přechody do nového stavu, ve kterém vyprázdníme zásobník.

**Důsledek 6:** Deterministický bezkontextový jazyk má prefixovou vlastnost, právě když je přijímán nějakým DZA pomocí vyprázdnění zásobníku.

**Věta 58:** Každý regulární jazyk je deterministický.

**Důkaz 49:** Pro  $L$  existuje KDA  $A = \langle \Sigma, Q, \delta, q_0, F \rangle$  s úplnou přechodovou funkcí.

**Příklad 49:** Následující jazyk je deterministický:

$$L = \{a^n b^n \mid n \geq 1\}$$

**Poznámka 19:** Mnoho regulárních jazyků **nemá** prefixovou vlastnost.

**Věta 59:** Pokud  $L = N(A)$  pro DZA  $A$ , pak je  $L$  jednoznačný.

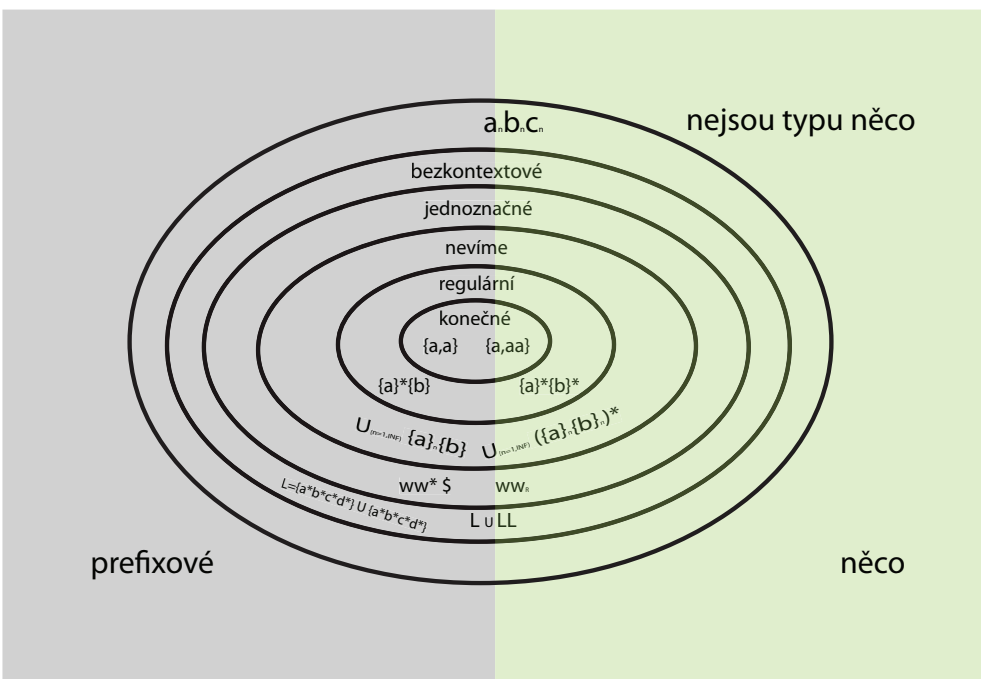
**Důkaz 50:** Náznak důkazu je následující:

Na základě  $A$  sestavíme gramatiku  $G$  tak, že  $L(G) = N(A)$  a navíc prokážeme, že gramatika  $G$  má jednoznačné odvození každé věty.

**Věta 60:** Pokud  $L = L(A)$  pro DZA  $A$ , pak je  $L$  jednoznačný. Deterministické bezkontextové jazyky jsou jednoznačné.

**Důkaz 51:** Náznak důkazu:

Vyjdeme z toho, že  $L = L(A)$  a uvažujeme, že  $L' = L \cdot \{\$ \}$ . I ten je deterministický. Tj. existuje  $A'$  tak, že  $L(A') = L' = L \cdot \{\$ \}$ . Dle předchozí věty existuje jednoznačná gramatika  $G$  tak, že  $L(G) = L \cdot \{\$ \}$ .



Obrázek 9. Vychodilovo „vajčko 2.“