

Automaty a Gramatiky

Poznámky z přednášek

Letní semestr 2020/2021

Viktor Soukup, Lukáš Salak

Obsah

1 První přednáška

Poznámka (Chomského hierarchie): Automaty a gramatiky - dva způsoby popisu:

Turingovy stroje	\leftrightarrow	gramatiky Typu 0
lineárně omezené automaty	\leftrightarrow	kontextové gramatiky, monotónní gramatiky
zásobníkové automaty	\leftrightarrow	bezkontextové gramatiky
konečné automaty (DFA, NFA, λ NFA)	\leftrightarrow	regulární jazyky

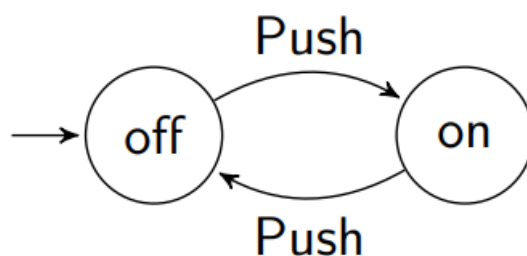
Nejjednodušší jsou nejnižší, turingův stroj je nejkomplikovanější. Každá gramatika odpovídá nějaké třídě automatů.

Proč to řešíme?

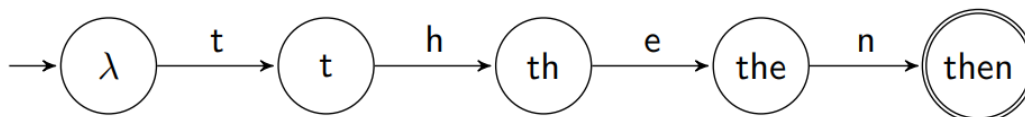
- zpracování přirozeného jazyka,
- překladače (lexikální, syntaktická analýza...),
- návrh, popis, verifikace hardware...
- hledání výskytu slova v textu (grep),
- verifikace systémů s konečně mnoha stavy

Příklad:

1. Návrh a verifikace integrovaných obvodů, např. Konečný automat modelující spínač on/off



2. Lexikální analýza, např. Konečný automat rozpoznávající slovo *then*



Definice (Deterministický konečný automat (DFA)): $A = (Q, \Sigma, \delta, q_0, F)$ sestává z:

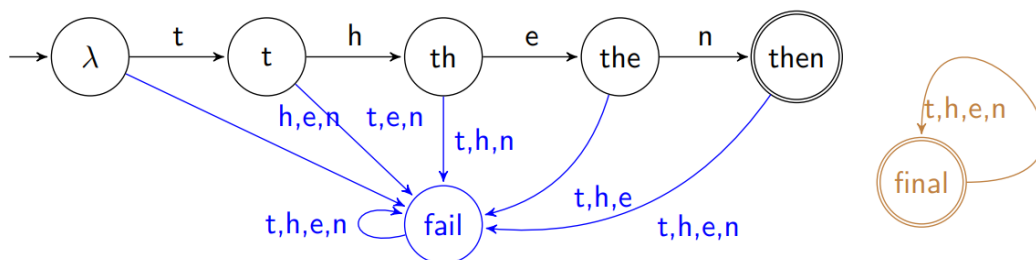
1. konečné množiny **stavů**, zpravidla značíme Q
2. konečné neprázdné množiny **vstupních symbolů (abecedy)**, značíme Σ
3. **přechodové funkce** zobrazení $Q \times X \rightarrow Q$, značíme δ , která bude reprezentovaná hranami grafu
4. **počátečního stavu** $q_0 \in Q$, vede do něj šipka 'odnikud'

5. neprázdné **množiny koncových (přijímajících) stavů** (final states) $F \subseteq Q$, označených dvojíým kruhem či šipkou 'ven'.

Poznámka:

Pokud pro některou dvojici stavu a písmene není definovaný přechod, přidáme nový stav *fail* a přechodovou funkci doplníme na totální přidáním šipek do *fail*.

Pokud je množina F prázdná, přidáme do ní i Q nový stav *final* do kterého vedou jen přechody z něj samotného $\forall s \in \Sigma : \delta(\text{final}, s) = \text{final}$.



Příklad:

Automat A přijímající $L = x01y : x, y \in \{0,1\}^*$.

Automat $A = (\{q_0, q_1, q_2\}, \{0,1\}, \delta, q_0, q_1)$

Reprezentujeme stavovým diagramem (grafem), pomocí tabulky nebo stavovým stromem

Definice (Abeceda, slova, jazyky): Mějme neprázdnou množinu symbolů Σ .

- **Slovo** je konečná (i prázdná) posloupnost symbolů $s \in \Sigma$, **prázdné slovo** se značí λ nebo ϵ
- **Množinu všech slov v abecedě** Σ značíme Σ^*
- množinu všech neprázdných slov v abecedě značíme Σ^+
- **jazyk** $L \subseteq \Sigma^*$ je množina slov v abecedě Σ

Definice (Operace na Σ^*):

1. **zřetězení slov** $u.v$ nebo uv
2. **mocnina** (počet opakování) $u^n (u^0 = \lambda, u^1 = u, u^{n+1} = u^n.u)$
3. **délka slova** $|u| (|\lambda| = 0, |auto| = 4)$
4. **počet výskytů** $s \in \Sigma$ ve slově u značíme $|u|_s (|zmrzlina|_z = 2)$.

Definice (Rozšířená přechodová funkce): Mějme přechodovou funkci $\delta : Q \times \Sigma \rightarrow Q$.

Rozšířenou přechodovou funkci $\delta^* : Q \times \Sigma^* \rightarrow Q$ (tranzitivní uzávěr δ) definujeme induktivně:

1. $\delta^*(q, \lambda) = q$,
2. $\delta^*(q, wx) = \delta(\delta^*(q, w), x)$ pro $x \in \Sigma, w \in \Sigma^*$.

Poznámka: Pokud se v textu objeví δ aplikované na slova, míní se tím δ^* .

Definice (Jazyk rozpoznávaný (přijímaný, akceptovaný) konečným automatem): Jazykem rozpoznávaným konečným automatem $A = (Q, \Sigma, \delta, q_0, F)$ nazveme jazyk $L(A) = \{w : w \in \Sigma^* \& \delta^*(q_0, w) \in F\}$.

- Slovo w je přijímáno automatem A , právě když $w \in L(A)$.
- Jazyk L je rozpoznatelný konečným automatem, jestliže existuje konečný automat A takový, že $L = L(A)$.
- Třidu jazyků rozpoznatelných konečnými automaty označíme \mathcal{F} , nazveme **regulární jazyky**.

Věta (Iterační (pumping) lemma pro regulární jazyky): *Mějme regulární jazyk L . Pak existuje konstanta $n \in \mathbb{N}$ (závislá na L) tak, že každé $w \in L; |w| \geq n$ můžeme rozdělit na tři části, $w = xyz$, že:*

1. $y \neq \lambda$
2. $|xy| \leq n$
3. $\forall k \in \mathbb{N}_0$, slovo xy^kz je také v L .

Důkaz:

- Mějme regulární jazyk L , pak existuje DFA A s n stavy, že $L = L(A)$.
- Vezměme libovolné slovo $a_1a_2a_3 \dots a_m = w \in L$ délky $m \geq n, a_i \in \Sigma$.
- Definujeme: $\forall i p_i = \delta^*(q_0, a_1a_2 \dots a_i)$. Platí $p_0 = q_0$.
- Máme $n + 1 p_i$ a n stavů, některý se opakuje, vezměme první takový, t. j. $(\exists i, j : 0 \leq i < j \leq n \wedge p_i = p_j)$.
- Definujeme $x = a_1a_2 \dots a_i, y = a_{i+1}a_{i+2} \dots a_j, z = a_{j+1}a_{j+2} \dots a_m$, t.j. $w = xyz, y \neq \lambda, |xy| \leq n$.
- pak y^k můžeme opakovat libovolněkrát a vstup je také akceptovaný.

□

Příklad (Aplikace pumping lemmatu): TODO

2 Druhá přednáška

Definice (Kongruence): Mějme konečnou abecedu Σ a relaci ekvivalence \sim na Σ^* (reflexivní, symetrická, tranzitivní). Potom:

1. \sim je pravá kongruence, jestliže $(\forall u, v, w \in \Sigma^*) u \sim v \implies uw \sim vw$.
2. je konečného indexu, jestliže rozklad Σ^* / \sim má konečný počet tříd.
3. Třidu kongruence \sim obsahující slovo u značíme $[u]_\sim$, resp. $[u]$.

Věta (Myhill-Nerodova Věta): *Nechť L je jazyk nad konečnou abecedou Σ . Potom následující tvrzení jsou ekvivalentní:*

1. L je rozpoznatelný konečným automatem,
2. \exists pravá kongruence \sim konečného indexu nad Σ^* tak, že L je sjednocením jistých tříd rozkladu Σ^* / \sim .

Důkaz:

1. \implies 2.; t.j. automat \implies pravá kongruence konečného indexu
 - definujeme $u \sim v \equiv \delta^*(q_0, u) = \delta^*(q_0, v)$.

- je to ekvivalence (reflexivní, symetrická, tranzitivní)
- je to pravá kongruence (z definice δ^*)
- má konečný index (konečně mnoho stavů)

$$L = \{w | \delta^*(q_0, w) \in F\} = \bigcup_{q \in F} \{w | \delta^*(q_0, w) = q\} = \bigcup_{q \in F} [w | \delta^*(q_0, w) = q]_{\sim}.$$

2. \implies 1.; t.j. pravá kongruence konečného indexu \implies automat

- abeceda automatu nazveme Σ
- za stavy Q volíme třídy rozkladu Σ^* / \sim
- počáteční stav $q_0 \equiv [\lambda]_{\sim}$
- koncové stavy $F = \{c_1, \dots, c_n\}$, kde $L = \bigcup_{i=1, n} c_i$
- přechodová funkce $\delta([u], x) = [ux]$ (je korektní z def. pravé kongruence).
- $L(A) = L$

$$w \in L \Leftrightarrow w \in \bigcup_{i=1, n} c_i \Leftrightarrow w \in c_1 \vee \dots \vee w \in c_n \Leftrightarrow [w] = c_1 \vee \dots \vee [w] = c_n \Leftrightarrow [w] \in F \Leftrightarrow w \in L(A)$$

□

Příklad: Sestrojte automat přijímající jazyk

$L = \{w | w \in a, b^* \& |w|_a = 3k + 2\}$, t. j. obsahuje $3k + 2$ symbolů a .

1. $|u|_x$ značí počet symbolů x ve slově u
2. definujeme $u \sim v \equiv (|u|_a \bmod 3 = |v|_a \bmod 3)$
3. třídy ekvivalence $0, 1, 2$
4. L odpovídá třídě 2
5. a - přechody do následující třídy
6. b - přechody zachovávající třídu

28. slide, doplň obrázek

Příklad (Neregulární pumpovatelný jazyk): Ne-regulární jazyk, který lze pumpovat

Jazyk $L = \{u | u = a^+ b^i c^i \vee u = b^i c^j\}$ není regulární (Myhill-Nerodova věta), ale vždy lze pumpovat první písmeno.

1. Předpokládejme, že L je regulární
2. \implies pak \exists pravá kongruence \sim_L konečného indexu m , L je sjednocení některých tříd Σ^* / \sim_L
3. vezmeme množinu slov $S = \{ab, abb, abbb, \dots, ab^{m+1}\}$
4. existují dvě slova $i \neq j$, která padnou do stejné třídy

$i \neq j$	$ab^i \sim ab^j$	
přidáme c^i	$ab^i c^i \sim ab^j c^i$	\sim je kongruence
spor	$ab^i c^i \in L \& ab^j c^i \notin L$	$s' L$ je sjednocení některých tříd Σ^* / \sim_L .

Definice (Dosažitelný stav): Mějme DFA $A = (Q, \Sigma, \delta, q_0, F)$ a $q \in Q$. Řekneme, že stav je dosažitelný, jestliže $\exists w \in \Sigma^* : \delta^*(q_0, w) = q$.

Příklad: Algoritmus na hledání dosažitelných stavů : DFS (důkaz asi není nutný)

Definice (Automatový homomorfismus): Necht' A_1, A_2 jsou DFA. Řekneme, že zobrazení $h : Q_1 \rightarrow Q_2$ na Q_2 je (automatovým) homomorfismem, jestliže:

$$\begin{array}{ll} h(q_{0_1}) = q_{0_2} & \text{'stejné' počáteční stavy} \\ h(\delta_1(q, x)) = \delta_2(h(q), x) & \text{'stejné' přechodové funkce} \\ q \in F_1 \Leftrightarrow h(q) \in F_2 & \text{'stejné' koncové stavy.} \end{array}$$

Homomorfismus prostý a na nazýváme isomorfismus.

Definice (Ekvivalence automatů): Dva konečné automaty A, B nad stejnou abecedou Σ jsou ekvivalentní, jestliže rozpoznávají stejný jazyk, t. j. $L(A) = L(B)$.

Věta (Věta o ekvivalenci automatů): *Existuje-li homomorfismus konečných automatů A_1 do A_2 , pak jsou A_1 a A_2 ekvivalentní.*

Důkaz:

1. Pro libovolné slovo $w \in \Sigma^*$ konečnou iterací

$$h(\delta_1^*(q, w)) = \delta_2^*(h(q), w)$$

2. dále

$$\begin{aligned} w \in L(A_1) &\Leftrightarrow \delta_1^*(q_{0_1}, w) \in F_1 \\ &\Leftrightarrow h(\delta_1^*(q_{0_1}, w)) \in F_2 \\ &\Leftrightarrow \delta_2^*(h(q_{0_1}, w)) \in F_2 \\ &\Leftrightarrow \delta_2^*(q_{0_2}, w) \in F_2 \\ &\Leftrightarrow w \in L(A_2) \end{aligned}$$

□

Definice (Ekvivalence stavů): Říkáme, že stavy $p, q \in Q$ konečného automatu A jsou ekvivalentní, pokud:

1. Pro všechna vstupní slova $w : \delta^*(p, w) \in F \Leftrightarrow \delta^*(q, w) \in F$.

Pokud dva stavy nejsou ekvivalentní, říkáme že jsou rozlišitelné.

Příklad: Ten example je na slide 36, nejlepší s tím obrázkem

Definice (Algoritmus hledání rozpoznatelných stavů v DFA): Následující algoritmus nalezne rozlišitelné stavy:

1. Základ: Pokud $p \in F$ (přijímající) a $q \notin F$, pak je dvojice $\{p, q\}$ rozlišitelná.
2. Indukce: Necht' $p, q \in Q, a \in \Sigma$ a o dvojici $r, s : r = \delta(p, a), s = \delta(q, a)$ víme, že jsou rozlišitelné. Pak i $\{p, q\}$ jsou rozlišitelné.
3. opakuj dokud \exists nová trojice $p, q \in Q, a \in \Sigma$.

Doplň obrázky zo slidov, 37/38

Věta: *Pokud dva stavy nejsou odlišeny předchodzím algoritmem, pak jsou tyto stavy ekvivalentní.*

Důkaz: Korektnost algoritmu

1. Uvažujme špatné páry stavů, které jsou rozlišitelné a algoritmus je nerozlišil.
2. Vezměme z nich pár p, q rozlišitelný nejkratším slovem $w = a_1 \dots a_n$.
3. Stavy $r = \delta(p, a_1), s = \delta(q, a_1)$ jsou rozlišitelné kratším slovem $a_2 \dots a_n$, takže pár není mezi špatnými.
4. Tedy jsou 'vyškrtnuté' algoritmem.
5. Tedy v příštím kroku algoritmus rozliší i p, q .

□

Čas výpočtu je polynomiální vzhledem k počtu stavů.

1. V jednom kole uvažujeme všechny páry, t.j. $O(n^2)$.
2. Kol je maximálně $O(n^2)$, protože pokud nepřidáme křížek, končíme.
3. Dohromady $O(n^4)$.

Algoritmus lze zrychlit na $O(n^2)$ pamatováním stavů, které závisí na páru $\{r, s\}$ a sledováním těchto seznamů 'zpátky'.

Definice (Redukovaný DFA): Deterministický konečný automat je redukovaný, pokud

1. nemá dosažitelné stavy,
2. žádné dva stavy nejsou ekvivalentní.

Definice (Redukt): Konečný automat B je redukem automatu A, jestliže:

1. B je redukovaný,
2. A a B jsou ekvivalentní

ADD PICS PLS (*don't shout pls*)

Věta (Algoritmus na nalezení reduktu DFA A):

1. Ze vstupního DFA A eliminujeme stavy nedosažitelné z počátečního stavu.
2. Najdeme rozklad zbylých stavů na třídy ekvivalence.
3. Konstruujeme DFA B na třídách ekvivalence jakožto stavech.
Přechodová funkce B γ , mějme $S \in Q_B$. Pro libovolné $q \in S$ označíme T třídu ekvivalence $\delta(q, a)$ a definujeme $\gamma(S, a) = T$. Tato třída musí být stejná pro všechna $q \in S$.
4. Počáteční stav B je třída obsahující počáteční stav A.
5. Množina přijímajících stavů B jsou bloky odpovídající přijímacím stavům A.

3 Třetí přednáška

Definice: Algoritmus na testování ekvivalence regulárních jazyků

Ekvivalenci regulárních jazyků L, M testujeme následovně:

1. Najdeme $DFA A_L, A_M$ rozpoznávající $L(A_L) = L, L(A_M) = M, Q_L \cap Q_M = \emptyset$.
2. Vytvoříme DFA sjednocením stavů a přechodů $(Q_L \cup Q_M, \Sigma, \delta_L \cap \delta_M, q_L, F_L \cap F_M)$; zvolíme jeden z počátečních stavů.
3. Jazyky jsou ekvivalentní právě když počáteční stavy původních DFA jsou ekvivalentní.

Nedeterministické konečné automaty (NFA)

Nedeterministický automat může být ve více stavech paralelně. Má schopnost 'uhodnout' něco o vstupu.

přidat obrázok zo slidov (61. slide)

Definice: Nedeterministický konečný automat (NFA) $A = (Q, \Sigma, \delta, S_0, F)$ sestává z:

1. konečné množiny stavů, zpravidla značíme Q ,
2. konečné množiny vstupních symbolů, značíme Σ
3. přechodové funkce, zobrazení $\delta : Q \times \Sigma \rightarrow \mathcal{P}(Q)$ vracející podmnožinu Q .
4. množiny počátečních stavů $S_0 \subseteq Q$,
5. množiny koncových (přijímajících) stavů $F \subseteq Q$.

Definice: Rozšířená přechodová funkce

Pro přechodovou funkci δ NFA je rozšířená přechodová funkce δ^*

$\delta^* : Q \times \Sigma^* \rightarrow \mathcal{P}(Q)$ definovaná indukci:

start: $\delta^*(q, \lambda) = q$. ind. indukční krok:

$$\delta^*(q, wx) = \bigcup_{p \in \delta^*(q, w)} \delta(p, x)$$

t. j. množina stavů, do kterých se mohou dostat posloupností 'správně označených'

Definice: Jazyk přijímaný nedeterministickým konečným automatem

Mějme NFA $A = (Q, \Sigma, \delta, S_0, F)$, Pak

$$L(A) = \{w : (\exists q_0 \in S_0) \delta^*(q_0, w) \cap F \neq \emptyset\}$$

je jazyk přijímaný automatem A .

Tedy $L(A)$ je množina slov $w \in \Sigma^*$ takových, že $\delta^*(q_0, w)$ obsahuje alespoň jeden přijímající stav.

Definice: Algoritmus : Podmnožinová konstrukce

Podmnožinová konstrukce začíná s NFA $N = (Q_N, \Sigma, \delta_N, S_0, F_N)$. Cílem je popis deterministického DFA $D = (Q_D, \Sigma, \delta_D, S_0, F_D)$, pro který $L(N) = L(D)$.

1. Q_D je množina podmnožin $Q_N, Q_D = \mathcal{P}(Q_N)$ (potenční množina).

Poznámka: Nedosažitelné stavy můžeme vynechat

2. Počáteční stav DFA je stav označený S_0 , t.j. prvek Q_D .
3. $F_D = \{S : S \in \mathcal{P}(Q_N) \& S \cap F_N \neq \emptyset\}$, tedy S obsahuje alespoň jeden přijímající stav N .
4. Pro každé $S \subseteq Q_N$ a každý vstupní symbol $a \in \Sigma$,

$$\delta_D(S, a) = \bigcup_{p \in S} \delta_N(p, a).$$

Věta: Převod NFA na DFA

Pro DFA $D = (Q_D, \Sigma, \delta_D, S_0, F_D)$ vytvořený podmnožinovou konstrukcí z NFA $N = (Q_N, \Sigma, \delta_N, q_0, F_N)$ platí $L(N) = L(D)$.

Důkaz: Indukcí dokážeme, že $\delta_D^*(S_0, w) = \delta_N^*(q_0, w)$. □

Můžeme přidat ještě tzv. λ -přechod.

Definice: Dovolíme přechody na λ , prázdné slovo, t.j. bez přečtení vstupního symbolu.
doplnit obrázok, 68. slide

Definice: λ -uzávěr

Pro $q \in Q$ definujeme λ -uzávěr $\lambda CLOSE(q)$ rekurzivně:

1. Stav q je $\lambda CLOSE(q)$.
2. Je-li $p \in \lambda CLOSE(q)$ a $r \in \delta(p, \lambda)$, pak i $r \in \lambda CLOSE(q)$.

Pro $S \subseteq Q$ definujeme $\lambda CLOSE(S) = \bigcup_{q \in S} \lambda CLOSE(q)$.

Definice: Rozšířená přechodová funkce a jazyk přijímaný λ -NFA

Nechť $E = (Q, \Sigma, \delta, S_0, F)$ je λ -NFA. Rozšířenou přechodovou funkci δ^* definujeme následovně:

1. $\delta^*(q, \lambda) = \lambda CLOSE(q)$.
2. indukční krok: $v = wa$, kde $w \in \Sigma^*, a \in \Sigma$.

$$\delta^*(q, wa) = \lambda CLOSE \left(\bigcup_{p \in \delta^*(q, w)} \delta(p, a) \right)$$

Věta: Eliminace λ -přechodů

Jazyk L je rozpoznatelný λ -NFA právě když je L regulární.

Důkaz: Pro libovolný λ NFA $E = (Q_E, \Sigma, \delta_E, S_0, F_E)$ zkonstruujeme DFA $D = (Q_D, \Sigma, \delta_D, q_D, F_D)$ přijímající stejný jazyk jako E .

1. $Q_D \subseteq \mathcal{P}(Q_E), \forall S \subseteq Q_E : \lambda CLOSE(S) \in Q_D$. V Q_D může být i \emptyset .
2. $q_D = \lambda CLOSE(S_0)$.
3. $F_D = \{S : S \in Q_D \& S \cap F_E \neq \emptyset\}$.
4. Pro $S \in Q_D, a \in \Sigma$ definujeme $\delta_D(S, a) = \lambda CLOSE(\bigcup_{p \in S} \delta(p, a))$.

□

Definice: Množinové operace nad jazyky

Mějme dva jazyky L, M . Definujeme následující operace:

1. binární sjednocení $L \cup M = \{w : w \in L \vee w \in M\}$

Poznámka: Příklad: jazyk obsahuje slova začínající a^i nebo tvaru $b^j c^j$.

2. průnik $L \cap M = \{w : w \in L \& w \in M\}$

Poznámka: Příklad: jazyk obsahuje slova sudé délky končící na 'baa'.

3. rozdíl $L - M = \{w : w \in L \& w \notin M\}$

4. doplněk (komplement) $\bar{L} = -L = \{w : w \in L\} = \sigma^* - L$

Poznámka: Příklad: jazyk obsahuje slova nekončící na 'a'.

Věta: *de Morganova pravidla:*

1. $L \cap M = \overline{\overline{L} \cup \overline{M}}$

2. $L \cup M = \overline{\overline{L} \cap \overline{M}}$

3. $L - M = L \cap \overline{M}$

Věta: *Uzavřenost na množinové operace*

Mějme regulární jazyky L, M . Pak jsou následující jazyky také regulární:

1. sjednocení $L \cup M$
2. průnik $L \cap M$
3. rozdíl $L - M$
4. doplněk $\bar{L} = \Sigma^* - L$.

Důkaz:

1. Pokud δ není pro některé dvojice q, a definovaná, přidáme nový nepřijímající stav q_n a do něj přechod pro vše dříve nedefinované plus $\forall a \in \Sigma \cup \lambda : \delta(q_n, a) = q_n$.
2. Pak stačí prohodit koncové a nekoncové stavy přijímajícího deterministického FA $F = Q_A - F_A$.
3. pro rozdíl doplníme funkci δ na totln. Zkonstruujeme součinový automat, $Q = (Q_1 \times Q_2, \Sigma, \delta((p_1, p_2), x) = (\delta_1(p_1, x), \delta_2(p_2, x)), (q_{01}, q_{02}, F))$
4. průnik: $F = F_1 \times F_2$
 sjednocení: $F = (F_1 \times Q_2) \cup (Q_1 \times F_2)$
 rozdíl: $F = F_1 \times (Q_2 - F_2)$.

□

Definice: Řetězcové operace nad jazyky

1. zřetězení jazyků ... $L.M = \{uv : u \in L \& v \in M\}$, $L.x = L.x$ a $x.L = x.L$ pro $x \in \Sigma$
2. mocniny jazyka ... $L^0 = \lambda$, $L^{i+1} = L^i.L$
3. pozitivní iterace ... $L^+ = L^1 \cup L^2 \dots \bigcup_{i \geq 1} L^i$
4. obecná iterace ... $L^* = L^0 \cup L^1 \cup \dots = \bigcup_{i \geq 0} L^i$, tedy $L^* = L^+ \cup \lambda$
5. otočení jazyka ... $L^R = \{u^R : u \in L\}$
6. levý kvocient L podle M ... M
 $L = \{u : uv \in L \& u \in M\}$
7. levá derivace L podle w ... $\partial_w L = \{w\}$
 L
8. pravý kvocient L podle M ... $L/M = \{u : uv \in L \& v \in M\}$
9. pravá derivace L podle w ... $\partial_w^R L = L/\{w\}$.

Věta: Jsou-li L, M regulární jazyky, je regulární i $L.M, L^*, L^+, L^R, M$
 LaL/M .

Věta: Jsou-li L, M regulární jazyky, je regulární i $L.M$.

Důkaz: Vezmeme DFA $A_1 = (Q_1, \Sigma, \delta_1, q_1, F_1)$, pak $A_2 = (Q_2, \Sigma, \delta_2, q_2, F_2)$ tak, že $L = L(A_1)$ a $M = L(A_2)$.

Definujeme Nedeterministický automat $B = (Q \cup q_0, \Sigma, \delta, q_0, F_2)$ kde:

$$Q = Q_1 \cup Q_2$$

předpokládáme různá jména stavů, jinak přejmenujeme, končíme až po přečtení slova z L_2

Pak $L(B) = L(A_1).L(A_2)$.

$\delta(q_0)$
 pro q_1
 $\delta(q_0)$
 pro q_1
 $\delta(q_0)$
 pro a
 $\delta(q_1)$
 pro $q \in Q_1 \&$
 pro $q \in Q_1 \&$
 pro q

□