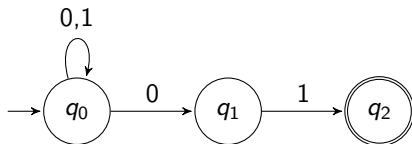


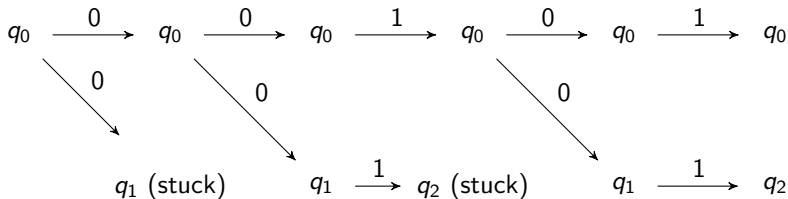
Nedeterministické konečné automaty s ϵ přechody (ϵ -NFA)

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

NFA přijímající všechna slova končící 01.



NFA zpracovává vstup 00101.



Definition 3.1 (Nedeterministický konečný automat s ϵ přechody (ϵ -NFA))

Nedeterministický konečný automat s ϵ přechody (ϵ -NFA)

$A = (Q, \Sigma, \delta, q_0, F)$ sestává z:

- ① konečné množiny **stavů**, zpravidla značíme Q
- ② konečné množiny **vstupních symbolů**, značíme Σ
- ③ **přechodové funkce**, zobrazení $\delta : Q \times (\Sigma \cup \{\epsilon\}) \rightarrow \mathcal{P}(Q)$ vracející podmnožinu Q .
- ④ **počáteční ho stavu**^a $q_0 \in Q$,
- ⑤ a **množiny koncových (přijímajících) stavů** $F \subseteq Q$.

^aalternativa: množiny počátečních stavů $S_0 \subseteq Q$

Example 3.1

Tabulka pro ϵ -NFA z předchozího slajdu $A = (\{q_0, q_1, q_2\}, \{0, 1\}, \delta, q_0, \{q_2\})$ je:

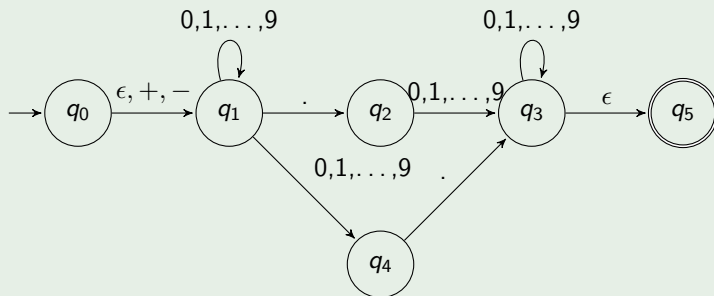
δ	ϵ	0	1
$\rightarrow q_0$	\emptyset	$\{q_0, q_1\}$	$\{q_0\}$
q_1	\emptyset	\emptyset	$\{q_2\}$
$*q_2$	\emptyset	\emptyset	\emptyset

Konečné automaty s ϵ přechody

- Nově dovolíme přechody na ϵ , prázdné slovo, tj. bez přečtení vstupního symbolu.

Example 3.2 (NFA s ϵ přechody)

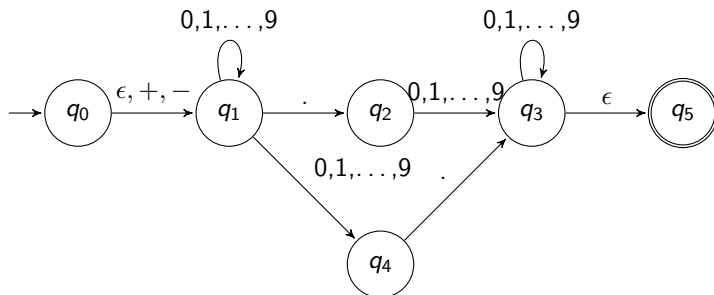
- Volitelně znaménko $+$ nebo $-$,
- řetězec číslic,
- desetinná tečka a
- další řetězec číslic. Nejméně jeden z řetězců (2) a (4) musí být neprázdný.



Example 3.3 (Přechodová funkce v tabulce)

Předešlý ϵ -NFA je: $E = (\{q_0, q_1, \dots, q_5\}, \{., +, -, 0, 1, \dots, 9\}, \delta, q_0, \{q_5\})$ s δ :

δ	ϵ	$+, -$	$.$	$0, 1, \dots, 9$
q_0	$\{q_1\}$	$\{q_1\}$	\emptyset	\emptyset
q_1	\emptyset	\emptyset	$\{q_2\}$	$\{q_1, q_4\}$
q_2	\emptyset	\emptyset	\emptyset	$\{q_3\}$
q_3	$\{q_5\}$	\emptyset	\emptyset	$\{q_3\}$
q_4	\emptyset	\emptyset	$\{q_3\}$	\emptyset
$*q_5$	\emptyset	\emptyset	\emptyset	\emptyset



ϵ -uzávěrDefinition 3.2 (ϵ -uzávěr)

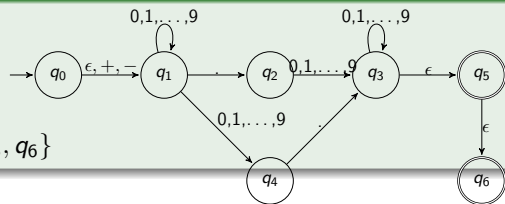
Pro $q \in Q$ definujeme ϵ -uzávěr $\epsilon\text{closure}(q)$ rekurzivně:

- Stav q je v $\epsilon\text{closure}(q)$.
- Je-li $p \in \epsilon\text{closure}(q)$ a $r \in \delta(p, \epsilon)$ pak i $r \in \epsilon\text{closure}(q)$.

Pro množinu stavů $S \subseteq Q$ definujeme $\epsilon\text{closure}(S) = \bigcup_{q \in S} \epsilon\text{closure}(q)$.

Example 3.4 (ϵ uzavěr)

- $\epsilon\text{closure}(q_0) = \{q_0, q_1\}$
- $\epsilon\text{closure}(q_1) = \{q_1\}$
- $\epsilon\text{closure}(q_3) = \{q_3, q_5, q_6\}$
- $\epsilon\text{closure}(\{q_3, q_4\}) = \{q_3, q_4, q_5, q_6\}$



Rozšířená přechodová funkce a jazyk přijímaný ϵ -NFA

Definition 3.3

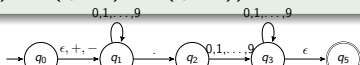
Nechť $E = (Q, \Sigma, \delta, q_0, F)$ je ϵ -NFA. Rozšířenou přechodovou funkci δ^* definujeme následovně:

- $\delta^*(q, \epsilon) = \epsilon\text{closure}(q)$.
- Indukční krok: $v = wa$, kde $w \in \Sigma^*$, $a \in \Sigma$.

$$\delta^*(q, wa) = \epsilon\text{closure} \left(\bigcup_{p \in \delta^*(q, w)} \delta(p, a) \right).$$

Example 3.5

$$\begin{aligned} \delta^*(q_0, \epsilon) &= \epsilon\text{closure}(q_0) &= \{q_0, q_1\} \\ \delta^*(q_0, 5) &= \epsilon\text{closure}(\bigcup_{q \in \delta^*(q_0, \epsilon)} \delta(q, 5)) = \epsilon\text{closure}(\delta(q_0, 5) \cup \delta(q_1, 5)) &= \{q_1, q_4\} \\ \delta^*(q_0, 5.) &= \epsilon\text{closure}(\delta(q_1, .) \cup \delta(q_4, .)) &= \{q_2, q_3, q_5, q_6\} \\ \delta^*(q_0, 5.6) &= \epsilon\text{closure}(\delta(q_2, 6) \cup \delta(q_3, 6) \cup \delta(q_5, 6)) &= \{q_3, q_5, q_6\} \end{aligned}$$



Jazyk přijímaný ϵ NFA

Definition 3.4 (Jazyk přijímaný nedeterministickým konečným automatem s přechody)

Mějme NFA $A = (Q, \Sigma, \delta, q_0, F)$, pak

$$L(A) = \{w \mid \delta^*(q_0, w) \cap F \neq \emptyset\}$$

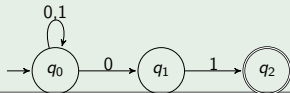
je jazyk přijímaný automatem A .

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

Example 3.6

Automat z předchozího slajdu přijímá jazyk $L = \{w \mid w \text{ končí na } 01, w \in \{0, 1\}^*\}$.
Důkaz indukcí konjunkce tvrzení:

- $\delta^*(q_0, w)$ obsahuje q_0 pro každé slovo w .
- $\delta^*(q_0, w)$ obsahuje q_1 iff w končí 0.
- $\delta^*(q_0, w)$ obsahuje q_2 iff w končí 01.



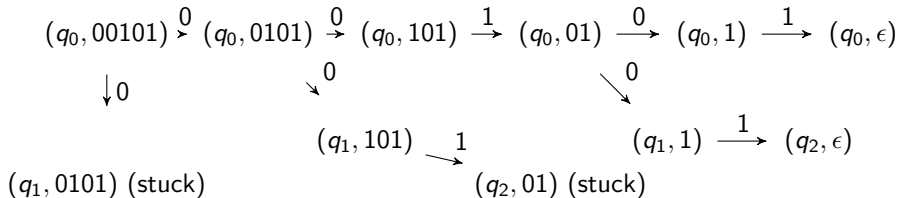
Konfigurace automatu, Výpočetní graf

Definition 3.5 (Konfigurace DFA, ϵ NFA)

Mějme ϵ NFA $A = (Q, \Sigma, \delta, q_0, F)$, $q \in Q$, $v \in \Sigma^*$, pak dvojice (q, v) označuje **konfiguraci** konečného automatu, nacházejícího se ve stavu q s nepřčteným vstupem v .

Definition 3.6 (Výpočetní strom, graf ϵ NFA)

Mějme NFA $A = (Q, \Sigma, \delta, q_0, F)$ a vstupní slovo $w \in \Sigma^*$. Uzly **výpočetního grafu** jsou konfigurace A nad slovem w , orientované hrany značí možný přechod mezi konfiguracemi, tj. z (p, au) vede hrana do (q, u) právě když $q \in \delta(p, a)$.



Podmnožinová konstrukce (s ϵ -přechody)

Theorem 3.1 (Podmnožinová konstrukce (s ϵ -přechody))

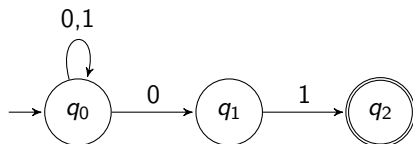
Jazyk L je rozpoznatelný ϵ -NFA právě když je L regulární.

Algorithm: **!** Podmnožinová konstrukce (s ϵ -přechody)

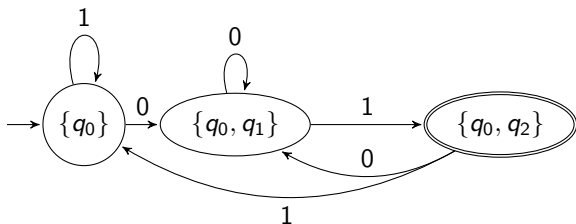
Pro libovolný ϵ -NFA $N = (Q_N, \Sigma, \delta_N, q_0, F_N)$ zkonstruujeme DFA $D = (Q_D, \Sigma, \delta_D, q_D, F_D)$ přijímající stejný jazyk jako N .

- Nové stavy jsou ϵ uzavřené podmnožiny Q_N .
 $Q_D \subseteq \mathcal{P}(Q_N), \forall S \subseteq Q_N : \epsilon\text{closure}(S) \in Q_D$. V Q_D může být i \emptyset .
- Počáteční stav je ϵ uzávěr q_0 .
 $q_D = \epsilon\text{closure}(q_0)$.
- Přijímací stavy jsou všechny množiny obsahující nějaký přijímací stav.
 $F_D = \{S \mid S \in Q_D \ \& \ S \cap F_N \neq \emptyset\}$.
- Přechodová funkce sjednotí předchody z jednotlivých stavů a uzavře $\epsilon\text{closure}$.

Pro $S \in Q_D, a \in \Sigma$ definujeme $\delta_D(S, a) = \epsilon\text{closure}(\bigcup_{p \in S} \delta(p, a))$.

Příklad podmnožinové konstrukce pro $\{w.01 \mid w \in \{0, 1\}^*\}$ 

	0	1
\emptyset	\emptyset	\emptyset
$\rightarrow \{q_0\}$	$\{q_0, q_1\}$	$\{q_0\}$
$\{q_1\}$	\emptyset	$\{q_2\}$
$*\{q_2\}$	\emptyset	\emptyset
$\{q_0, q_1\}$	$\{q_0, q_1\}$	$\{q_0, q_2\}$
$*\{q_0, q_2\}$	$\{q_0, q_1\}$	$\{q_0\}$
$*\{q_1, q_2\}$	\emptyset	$\{q_2\}$
$*\{q_0, q_1, q_2\}$	$\{q_0, q_1\}$	$\{q_0, q_2\}$



Theorem 3.2 (Převod NFA na DFA)

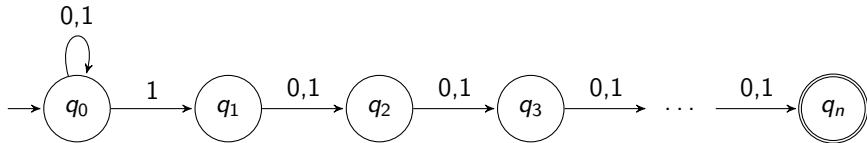
Pro DFA $D = (Q_D, \Sigma, \delta_D, q_{D_0}, F_D)$ vytvořený podmnožinovou konstrukcí z NFA $N = (Q_N, \Sigma, \delta_N, q_0, F_N)$ platí $L(N) = L(D)$.

Proof.

Indukcí dokážeme: $\delta_D^*(q_0, w) = \delta_N^*(q_{D_0}, w)$. □

Example 3.7 ('Těžký' případ pro podmnožinovou konstrukci)

Jazyk $L(N)$ slov 0's a 1's takových, že n -tý symbol od konce je 1. Intuitivně si DFA musí pamatovat n posledních přečtených symbolů.



- Aplikace hledání v textu

Množinové operace nad jazyky

Definition 3.7 (Množinové operace nad jazyky)

Mějme dva jazyky L, M . Definujme následující operace:

- (1) binární **sjednocení** $L \cup M = \{w \mid w \in L \vee w \in M\}$
 - Příklad: jazyk obsahuje slova začínající a^i nebo tvaru $b^j c^j$.
- (2) **průnik** $L \cap M = \{w \mid w \in L \ \& \ w \in M\}$
 - Příklad: jazyk obsahuje slova sudé délky končící na 'baa'.
- (3) **rozdíl** $L - M = \{w \mid w \in L \ \& \ w \notin M\}$
 - Příklad: jazyk obsahuje slova sudé délky nekončící na 'baa'.
- (4) **doplňěk (komplement)** $\bar{L} = -L = \{w \mid w \notin L\} = \Sigma^* - L$
 - Příklad: jazyk obsahuje slova nekončící na 'a'.

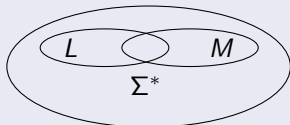
Theorem (de Morganova pravidla)

$$L \cap M = \overline{\bar{L} \cup \bar{M}}$$

Platí: $L \cup M = \overline{\bar{L} \cap \bar{M}}$

$$L - M = L \cap \bar{M}.$$

Důkaz ze vztahů $\&$, \vee , \neg .



Uzávěrové vlastnosti regulárních jazyků

Theorem 3.3 (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 $\bar{L} = \Sigma^* - L$.

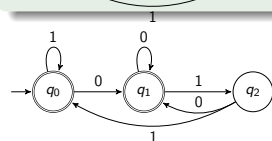
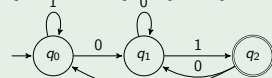
Proof: Uzavřenost RJ na doplněk

- Pokud δ není pro některé dvojice q, a definovaná, přidáme nový nepřijímající stav q_{fail} a do něj přechod pro vše dříve nedefinované plus $\forall a \in \Sigma: \delta(q_{fail}, a) = q_{fail}$.
- Pak stačí prohodit koncové a nekoncevé stavy přijímajícího deterministického FA $F_{doplňek} = Q_A - F_A$. □

Example 3.8

Jazyk

$\{w; w \in \{0, 1\}^*01\}$



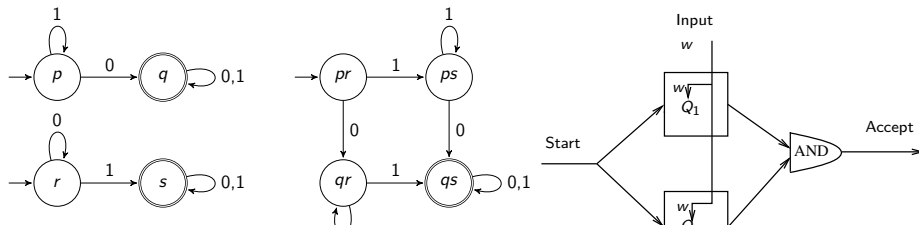
Konstrukce součinu automatů

Proof: Průnik, sjednocení, rozdíl

- Pro sjednocení a rozdíl doplníme funkci δ na totální.
- Zkonstruujeme součinnový automat,
 $Q = (Q_1 \times Q_2, \Sigma, \delta((p_1, p_2), x) = (\delta_1(p_1, x), \delta_2(p_2, x)), (q_{0_1}, q_{0_2}), F)$
- průnik: $F = F_1 \times F_2$
- sjednocení: $F = (F_1 \times Q_2) \cup (Q_1 \times F_2)$
- rozdíl: $F = F_1 \times (Q_2 - F_2)$.



Příklad součinu automatů (průnik jazyků). Slova obsahující 0,1, oboje.



Příklady na uzávěrové vlastnosti

Example 3.9

Konstruujeme konečný automat přijímající slova, která obsahují $3k + 2$ symbolů 1 a neobsahují posloupnost 11.

- Přímá konstrukce je komplikovaná.
- $L_1 = \{w \mid w \in \{0, 1\}^* \& |w|_1 = 3k + 2\}$
- $L_2 = \{w \mid u, v \in \{0, 1\}^* \& w = u11v\}$
- $L = L_1 - L_2$.

Example 3.10

Jazyk M slov s různým počtem 0 a 1 není regulární.

- Je-li M regulární, $\overline{M} = \Sigma^* - M$ je také regulární.
- O \overline{M} víme, že regulární není (pumping lemma).

Ještě jeden příklad

Example 3.11

Jazyk $L_{0 \neq 1} = \{0^i 1^j : i \neq j, i, j \in \mathbb{N}_0\}$ není regulární.

- Jazyk $L_{01} = \{0^i 1^j : i, j \in \mathbb{N}_0\}$ je regulární, umíme sestavit konečný automat.
- $L_{01} - L_{0 \neq 1} = \{0^i 1^i : i \in \mathbb{N}_0\}$
- Z uzávěrových vlastností víme, že rozdíl regulárních jazyků je regulární.
- Jazyk L_{01} regulární je.
- Předpokládejme, že $L_{0 \neq 1}$ je regulární. Pak by i $\{0^i 1^i : i \in \mathbb{N}_0\}$ musel být regulární, což není - SPOR.

Řetězcové operace nad jazyky

Definition 3.8 (Řetězcové operace nad jazyky)

Nad jazyky L, M definujeme následující operace:

zřetězení jazyků

$$L.M = \{uv \mid u \in L \ \& \ v \in M\}$$

$$L.x = L.\{x\} \text{ a } x.L = \{x\}.L \text{ pro } x \in \Sigma$$

mocniny jazyka

$$L^0 = \{\epsilon\}$$

$$L^{i+1} = L^i.L$$

pozitivní iterace

$$L^+ = L^1 \cup L^2 \dots = \bigcup_{i \geq 1} L^i$$

obecná iterace

$$L^* = L^0 \cup L^1 \cup L^2 \dots = \bigcup_{i \geq 0} L^i$$

$$\text{tedy } L^* = L^+ \cup \{\epsilon\}$$

otočení jazyka

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

(=zrcadlový obraz, reverze)

$$(x_1x_2 \dots x_n)^R = x_nx \dots x_2x_1$$

levý kvocient L podle M

$$M \setminus L = \{v \mid uv \in L \ \& \ u \in M\}$$

levá derivace L podle w

$$\partial_w L = \{w\} \setminus L \text{ (pozn. derivace bude i v jiném významu)}$$

pravý kvocient L podle M

$$L/M = \{u \mid uv \in L \ \& \ v \in M\}$$

pravá derivace L podle w

$$\partial_w^R L = L/\{w\}.$$

Theorem 3.4 (Uzavřenost reg. jazyků na řetězcové operace)

Jsou-li L, M regulární jazyky, je regulární i $L.M, L^, L^+, L^R, M \setminus L$ a L/M .*

Lemma ($L.M$)

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

Proof:

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

$\delta(q_0, \epsilon) = \{q_1, q_2\}$ pro $q_1 \in F_1$ tj. $\epsilon \in L(A_1)$

$\delta(q_0, \epsilon) = \{q_1\}$ pro $q_1 \notin F_1$ tj. $\epsilon \notin L(A_1)$

$\delta(q_0, x) = \emptyset$ pro $x \in \Sigma$

$\delta(q, x) = \{\delta_1(q, x)\}$ pro $q \in Q_1$ & $\delta_1(q, x) \notin F_1$ počítáme v A_1

$= \{\delta_1(q, x), q_2\}$ pro $q \in Q_1$ & $\delta_1(q, x) \in F_1$ nedet. přechod z A_1

$= \{\delta_2(q, x)\}$ pro $q \in Q_2$ počítáme v A_2 .

Pak $L(B) = L(A_1).L(A_2)$. □

Uzavřenost iterace

Lemma (L^* , L^+)

Je-li L regulární jazyk, je regulární i L^* , L^+ .

- Idea: opakovaný výpočet automatu $A = (Q, \Sigma, \delta, q_0, F)$
- realizace: nedeterministické rozhodnutí, zda pokračovat nebo restart
- speciální stav pro příjem $\epsilon \in L^0$ (pro L^+ vynecháme či $\notin F$).

Proof: Důkaz pro L^*

Vezmeme DFA $A = (Q, \Sigma, \delta, q_0, F)$, tak že $L = L(A)$.

Definujeme NFA automat $B = (Q \cup \{q_B\}, \Sigma, \delta_B, q_B, F \cup \{q_B\})$ kde:

$\delta_B(q_B, \epsilon) = \{q_0\}$ nový stav q_B pro příjem ϵ , přejdeme do q_0

$\delta_B(q_B, x) = \emptyset$ pro $x \in \Sigma$

$\delta_B(q, x) = \{\delta(q, x)\}$ pokud $q \in Q$ & $\delta(q, x) \notin F$ uvnitř A

$= \{\delta(q, x), q_0\}$ pokud $q \in Q$ & $\delta(q, x) \in F$ možný restart

Pak $L(B) = L(A)^*$ ($q_B \in F_B$), $L(B) = L(A)^+$ ($q_B \notin F_B$). □

Uzavřenost reverze

Lemma (L^R)

Je-li L regulární jazyk, je regulární i L^R .

- Zřejmě $(L^R)^R = L$ a tedy stačí ukázat jeden směr.
- idea: obrátíme šipky ve stavovém diagramu; nedeterministický FA

Proof:

Vezmeme DFA $A = (Q, \Sigma, \delta, q_0, F)$, tak že $L = L(A)$.

Definujeme nedeterministický automat $B = (Q \cup \{q_B\}, \Sigma, \delta_B, q_B, \{q_0\})$ kde:

- $\delta_B(q, x) = \{p \mid \delta(p, x) = q\}$ pro $q \in Q$
 - $\delta_B(q_B, \epsilon) = F$, $\delta_B(q_B, x) = \emptyset$.
 - Pro libovolné slovo $w = x_1x_2 \dots x_n$
 - $q_0, q_1, q_2, \dots, q_n$ je přijímající výpočet pro w v A
- \Leftrightarrow
- $q_B, q_n, q_{n-1}, \dots, q_2, q_1, q_0$ je přijímající výpočet pro w^R v B . □

Uzavřenost kvocientu

Lemma ($M \setminus L$ a L/M)

Jsou-li L, M regulární jazyky, je regulární i $M \setminus L$ a L/M .

- Idea: A_L , budeme startovat ve stavech, do kterých se lze dostat slovem z M .

Proof:

- $v \in M \setminus L$
 - $\Leftrightarrow (\exists u \in M) uv \in L$
 - $\Leftrightarrow (\exists u \in M, \exists q \in Q) \delta(q_0, u) = q \ \& \ \delta(q, v) \in F$
 - $\Leftrightarrow \exists q \in S_0 \ \& \ \delta(q, v) \in F$
 - $\Leftrightarrow v \in L(B)$

Vezmeme DFA $A = (Q, \Sigma, \delta, q_0, F)$, tak že $L = L(A)$.

Definujeme nedeterministický NFA $B = (Q, \Sigma, \delta, q_\epsilon, F)$ kde:

- a přidáme přechod $\delta(q_\epsilon, q)$ do stavů $\{q \mid q \in Q \ \& \ (\exists u \in M) q = \delta^*(q_0, u)\}$.
 - lze nalézt algoritmicky $(\{q; L(A_q) \cap M \neq \emptyset \text{ kde } A_q = (Q, \Sigma, \delta, q_0, \{q\})\})$

$$L/M = (M^R \setminus L^R)^R.$$



Regulární výrazy

Regulární výrazy (RV) jsou

- algebraickým popisem jazyků
- deklarativním způsobem, jak vyjádřit slova, která chceme přijímat.
- Schopné definovat všechny a pouze regulární jazyky.
- Můžeme je brát jako programovací jazyk, uživatelsky přívětivý popis konečného automatu.

Example 4.1

- Základní UNIX grep příkaz.
- Lexikální analyzátoři jako Lex a Flex (popis pomocí 'token'ů je vzásadě regulární výraz).
- Python knihovna re .
- Syntaktická analýza potřebuje silnější nástroj, bezkontextové gramatiky, budou následovat.

Definition 4.1 (Regulární výrazy (Regular Expression) (RegE), hodnota $\text{RegE } L(\alpha)$)

Regulární výrazy $\alpha, \beta \in \text{RegE}(\Sigma)$ nad konečnou neprázdnou abecedou $\Sigma = \{a_1, a_2, \dots, a_n\}$ a jejich hodnota $L(\alpha)$ jsou definovány induktivně:

	výraz α	pro	hodnota $L(\alpha) \equiv [\alpha]$
• Základ:	ϵ	prázdný řetězec	$L(\epsilon) = \{\epsilon\}$
	\emptyset	prázdný výraz	$L(\emptyset) = \{\} \equiv \emptyset$
	\mathbf{a}	$a \in \Sigma$	$L(\mathbf{a}) = \{a\}$.

• Indukce:

výraz	hodnota	poznámka
$\alpha + \beta$	$L(\alpha + \beta) = L(\alpha) \cup L(\beta)$	v grep, re
$\alpha\beta$	$L(\alpha\beta) = L(\alpha)L(\beta)$	můžeme značit ., ale plete se s UNIX grep.
α^*	$L(\alpha^*) = L(\alpha)^*$	
(α)	$L((\alpha)) = L(\alpha)$	závorky nemění hodnotu.

Každý regulární výraz dostaneme indukcí výše, tj. třída $\text{RegE}(\Sigma)$ je nejmenší třída uzavřená na uvedené operace.

Lemma 4.1

Jazyk $L(\epsilon) = \{\epsilon\} = \emptyset^*$, v definici jen pro význam symbolu $L(\epsilon)$.

Příklady regulárních výrazů, priorita

Example 4.2 (Regulární výrazy)

Jazyk střídajících se nul a jedniček lze zapsat:

- $(01)^* + (10)^* + 1(01)^* + 0(10)^*$
- $(\epsilon + 1)(01)^*(\epsilon + 0)$.

Jazyk $L((0^*10^*10^*1)^*0^*) = \{w \mid w \in \{0, 1\}^*, |w|_1 = 3k, k \geq 0\}$.

Definition 4.2 (priorita)

Nejvyšší prioritu má iterace $*$, nižší konkatenace (zřetězení), nejnižší sjednocení $+$.

Theorem 4.1 (varianta Kleeneho věty)

- 1 Každý jazyk reprezentovaný konečným automatem lze zapsat jako regulární výraz.
- 2 Každý jazyk popsaný regulárním výrazem můžeme zapsat jako ϵ -NFA (a tedy i DFA).

Převod RegE výrazu na ϵ -NFA automatPřevod RegE výrazu na ϵ -NFA automat.

Důkaz indukcí dle struktury R . Základ:

V každém kroku zkonstruujeme ϵ -NFA E rozpoznávající stejný jazyk $L(R) = L(E)$ se třemi dalšími vlastnostmi:

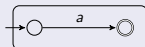
- 1 Právě jeden přijímající stav.
- 2 Žádné hrany do počátečního stavu.
- 3 Žádné hrany z koncového stavu.



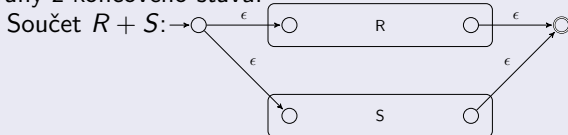
Prázdný řetězec ϵ



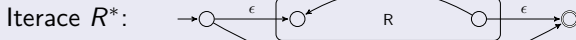
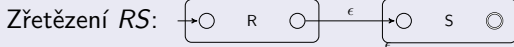
Prázdná množina \emptyset



$a \in \Sigma$: výraz a



INDUKCE:

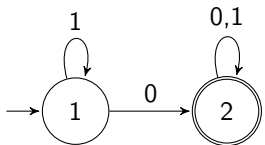


Od DFA k regulárním výrazům

Regulární výraz z DFA

- Mějme DFA A , $Q_A = \{1, \dots, n\}$ o n stavech, stavy očísľujeme od 1.
- Necht' $R_{ij}^{(k)}$ je regulární výraz, $L(R_{ij}^{(k)}) = \{w \mid \delta_{\leq k}^*(i, w) = j\}$ množina slov převádějících stav i do stavu j v A cestou, která neobsahuje stav s vyšším indexem než k .
- Budeme rekurzivně konstruovat $R_{ij}^{(k)}$ pro $k = 0, \dots, n$.
- $k = 0, i \neq j$: $R_{ij}^{(0)} = \mathbf{a}_1 + \mathbf{a}_2 + \dots + \mathbf{a}_m$ kde a_1, a_2, \dots, a_m jsou symboly označující hrany i do j (nebo $R_{ij}^{(0)} = \emptyset$ nebo $R_{ij}^{(0)} = \mathbf{a}$ pro $m = 0, 1$).
- $k = 0, i = j$: smyčky, $R_{ii}^{(0)} = \epsilon + \mathbf{a}_1 + \mathbf{a}_2 + \dots + \mathbf{a}_m$ kde a_1, a_2, \dots, a_m jsou symboly na smyčkách v i .

Příklad: Od konečného automatu k RegE

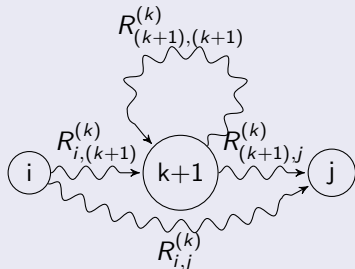


$$R_{12}^{(2)} = \mathbf{1^*0(0+1)^*}$$

$$R_{ij}^{(k+1)} = R_{ij}^{(k)} + R_{i(k+1)}^{(k)}(R_{(k+1)(k+1)}^{(k)})^*R_{(k+1)j}^{(k)}$$

$R_{11}^{(0)}$	$\epsilon + \mathbf{1}$	=
$R_{12}^{(0)}$	$\mathbf{0}$	=
$R_{21}^{(0)}$	\emptyset	=
$R_{22}^{(0)}$	$(\epsilon + \mathbf{0} + \mathbf{1})$	=
$R_{11}^{(1)}$	$(\epsilon + \mathbf{1}) + (\epsilon + \mathbf{1})(\epsilon + \mathbf{1})^*(\epsilon + \mathbf{1})$	$= \mathbf{1^*}$
$R_{12}^{(1)}$	$\mathbf{0} + (\epsilon + \mathbf{1})(\epsilon + \mathbf{1})^*\mathbf{0}$	$= \mathbf{1^*0}$
$R_{21}^{(1)}$	$\emptyset + \emptyset(\epsilon + \mathbf{1})^*(\epsilon + \mathbf{1})$	$= \emptyset$
$R_{22}^{(1)}$	$(\epsilon + \mathbf{0} + \mathbf{1}) + \emptyset(\epsilon + \mathbf{1})^*\mathbf{0}$	$= \epsilon + \mathbf{0} + \mathbf{1}$
$R_{11}^{(2)}$	$\mathbf{1^*} + \mathbf{1^*0}(\epsilon + \mathbf{0} + \mathbf{1})^*\emptyset$	$= \mathbf{1^*}$
$R_{12}^{(2)}$	$\mathbf{1^*0} + \mathbf{1^*0}(\epsilon + \mathbf{0} + \mathbf{1})^*(\epsilon + \mathbf{0} + \mathbf{1})$	$= \mathbf{1^*0(0+1)^*}$
$R_{21}^{(2)}$	$\emptyset + (\epsilon + \mathbf{0} + \mathbf{1})(\epsilon + \mathbf{0} + \mathbf{1})^*\emptyset$	$= \emptyset$
$R_{22}^{(2)}$	$(\epsilon + \mathbf{0} + \mathbf{1}) + (\epsilon + \mathbf{0} + \mathbf{1})(\epsilon + \mathbf{0} + \mathbf{1})^*(\epsilon + \mathbf{0} + \mathbf{1})$	$= (\mathbf{0} + \mathbf{1})^*$

INDUKCE. Mějme $\forall i, j \in Q R_{ij}^{(k)}$. Konstruujeme $R_{ij}^{(k+1)}$.



$$1 \quad R_{ij}^{(k+1)} = R_{ij}^{(k)} + R_{i(k+1)}^{(k)} (R_{(k+1)(k+1)}^{(k)})^* R_{(k+1)j}^{(k)}$$

- Cesty z i do j neprocházející uzlem $(k+1)$ jsou již v $R_{ij}^{(k)}$.
- Cesty z i do j přes $(k+1)$ s případnými smyčkami můžeme zapsat $R_{i(k+1)}^{(k)} (R_{(k+1)(k+1)}^{(k)})^* R_{(k+1)j}^{(k)}$.
- regulární výrazy jsou uzavřené na sčítání (sjednocení), zřetězení i iteraci, tj. $R_{ij}^{k+1} \in \text{RegE}(\Sigma)$

2 Nakonec, $\text{RegE} = \bigoplus_{j \in F_A} R_{1j}^{(n)}$ sjednocení přes přijímající stavy j .



Zjednodušení regulárních výrazů (netřeba znát)

Lemma (Další vlastnosti bez důkazu)

- Zjednodušení návrhu automatů

$$\begin{aligned}
 L.\emptyset &= \emptyset.L &= \emptyset \\
 \{\epsilon\}.L &= L.\{\epsilon\} &= L \\
 (L^*)^* &= L^* \\
 (L_1 \cup L_2)^* &= L_1^*(L_2.L_1^*)^* = L_2^*(L_1.L_2^*)^* \\
 (L_1.L_2)^R &= L_2^R.L_1^R \\
 \partial_w(L_1 \cup L_2) &= \partial_w(L_1) \cup \partial_w(L_2) \\
 \partial_w(\Sigma^* - L) &= \Sigma^* - \partial_w L.
 \end{aligned}$$

Shrnutí převodů mezi reprezentacemi regulárních jazyků

Převod NFA na DFA

- ϵ uzávěr v $O(n^3)$ – prohledává n stavů násobeno n^2 hran pro ϵ přechody.
- Podmnožinová konstrukce, DFA s až 2^n stavy. Pro každý stav, $O(n^3)$ času na výpočet přechodové funkce.

Převod DFA na NFA

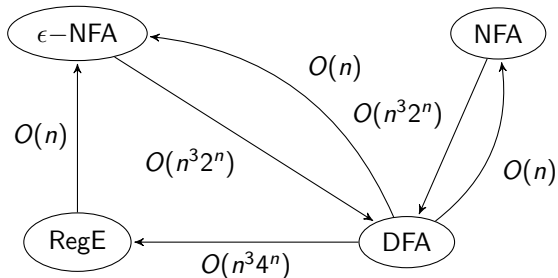
- Přidat množinové závorky k přechodové funkci a přechody pro ϵ u ϵ -NFA.

Převod automatu DFA an RegE regulární výraz

- $O(n^3 4^n)$

RegE výraz na automat

- V čase $O(n)$ vytvoříme ϵ -NFA.



Shrnutí

- Podmnožinová konstrukce DFA z ϵ NFA
- Regulární výrazy
- Kleeneho věta
 - Jazyk je přijímaný konečným automatem právě když lze napsat jako regulární výraz,
 - tj. z \emptyset a $\{a\}$ pro $a \in \Sigma$
 - a konečného počtu aplikací iterace, zřetězení a sjednocení.
- Uzávěrové vlastnosti
 - dnes jen 'regulární' sloupec.

jazyk	regulární (RL)	bezkontextové	deterministické CFL
sjednocení	ANO	ANO	NE
průnik	ANO	NE	NE
\cap s RL	ANO	ANO	ANO
doplňěk	ANO	NE	ANO
homomorfizmus	ANO	ANO	NE
inverzní hom.	ANO	ANO	ANO

Substitute jazyků

Definition 4.3 (Substitute jazyků)

Mějme konečnou abecedu Σ . Pro každé $x \in \Sigma$ budiž $\sigma(x)$ jazyk v nějaké abecedě Y_x . Dále položme

$$\sigma(\epsilon) = \{\epsilon\}$$

$$\sigma(u.v) = \sigma(u).\sigma(v)$$

- Zobrazení $\sigma : \Sigma^* \rightarrow P(Y^*)$, kde $Y = \bigcup_{x \in \Sigma} Y_x$ se nazývá **substitute**.
- Pro jazyk L definujeme: $\sigma(L) = \bigcup_{w \in L} \sigma(w)$, podobně sjednocení.
- **nevypouštějící substitute** je substitute, kde žádné $\sigma(x)$ neobstahuje ϵ .

Example 4.3 (substitute)

- 1) $\Sigma = \{k, p, m, c, t\}$, $L = (kmp)(ckmp)^*t$,
 k slovník křestních jmen, p slovník příjmení, m mezera, c čárka, t tečka.
- 2) Pokud $\sigma(0) = \{a^i b^j, i, j \geq 0\}$, $\sigma(1) = \{cd\}$
 tak $\sigma(010) = \{a^i b^j c d a^k b^l, i, j, k, l \geq 0\}$.

Homomorfismus a inverzní homomorfismus jazyků

Definition 4.4 (homomorfismus (jazyků), inverzní homomorfismus)

Homomorfismus h je speciální případ substituce, kde obraz je vždy jen jednoslovný jazyk (vynecháváme u něj závorky), tj. $(\forall x \in \Sigma) h(x) = w_x$.

Pokud $\forall x : w_x \neq \epsilon$, jde o **nevypouštějící homomorfismus**.

Inverzní homomorfismus $h^{-1}(L) = \{w | h(w) \in L\}$.

Example 4.4 (homomorfismus)

- Znak μ nahradíme $\backslash mu$ a podobně.
- Homomorfismus h definujeme: $h(0) = ab$, a $h(1) = \epsilon$. Pak $h(0011) = abab$.
Pro $L = \mathbf{10^*1}$ je $h(L) = (ab)^*$.

Theorem 4.2 (uzavřenost na homomorfismus)

Je-li jazyk L i $\forall x \in \Sigma$ jazyk $\sigma(x), h(x)$ regulární, pak je regulární i $\sigma(L), h(L)$.

Uzavřenost na substituci, homomorfismus.

Strukturální indukci 'probubláváním' algebraickým popisem jazyka základních, sjednocení, zřetězení a iterace. Pro sjednocení a zřetězení z definice substitute a uzavřenosti regulárních jazyků na sjednocení a zřetězení.

$$\begin{aligned}\underline{\sigma}(\alpha + \beta) &= \sigma(L(\alpha)) \cup \sigma(L(\beta)) \\ \underline{\sigma}(\alpha\beta) &= \{w \mid \exists u \in L(\alpha) \exists v \in L(\beta) : \sigma(u)\sigma(v) = w\}\end{aligned}$$

Pro iteraci rozložíme na nekonečné sjednocení, pro každý konkrétní počet iterací σ aplikované na konečné zřetězení.

$$\begin{aligned}\sigma(L(\alpha)^*) &= \sigma(L(\alpha)^0) \cup \sigma(L(\alpha)^1) \cup \dots \cup \sigma(L(\alpha)^n) \cup \dots \\ &= \underline{\sigma}(\alpha)^0 \cup \underline{\sigma}(\alpha)^1 \cup \dots \cup \underline{\sigma}(\alpha)^n \cup \dots \\ &= L(\underline{\sigma}(\alpha)^*).\end{aligned}$$



Inverzní homomorfismus

Definition ((4.4) Inverzní homomorfismus)

Nechť h je homomorfismus abecedy T do slov nad abecedou Σ . Pak $h^{-1}(L)$ 'h inverze L ' je množina řetězců

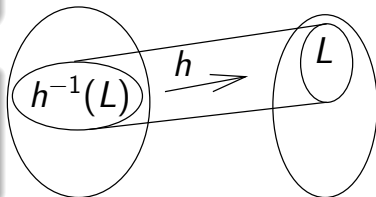
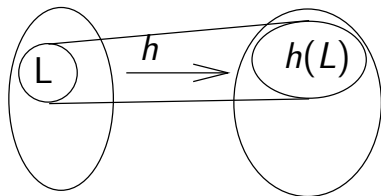
$$h^{-1}(L) = \{w \mid w \in T^*; h(w) \in L\}.$$

Example 4.5

Nechť $L = (\{00\} \cup \{1\})^*$, $h(a) = 01$ a $h(b) = 10$.

Pak $h^{-1}(L) = (\{ba\})^*$.

Důkaz: $h((\{ba\})^*) \in L$ snadno.
Ostatní w generují izolované 0 (rozbor případů).

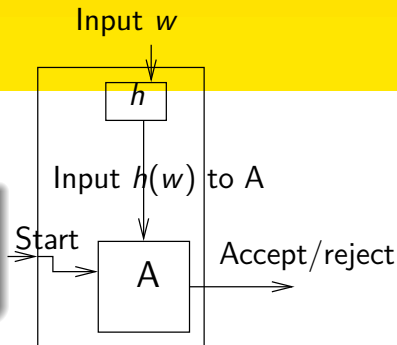


Homomorfismus aplikovaný
dopředu a zpětně.

Inverzní homomorfizmus DFA

Theorem 4.3

Je-li h homomorfizmus abecedy T do abecedy Σ a L je regulární jazyk abecedy Σ , pak $h^{-1}(L)$ je také regulární jazyk.



Proof:

- pro L máme DFA $A = (Q, \Sigma, \delta, q_0, F)$
- $h : T \rightarrow \Sigma^*$
- definujeme ϵ -NFA $B = (Q', T, \delta', [q_0, \epsilon], F \times \{\epsilon\})$ kde

$$Q' = \{[q, u] \mid q \in Q, u \in \Sigma^*, \exists(a \in T) \exists(v \in \Sigma^*) h(a) = vu\}$$

$$\delta'([q, \epsilon], a) = [q, h(a)]$$

$$\delta'([q, bv], \epsilon) = [p, v] \text{ kde } \delta(q, b) = p$$

u je buffer
naplňuje buffer
čte buffer.

□

Příklad: Navštív všechny stavy

Example 4.6

Nechť $A = (Q, \Sigma, \delta, q_0, F)$ je DFA. Definujme jazyk $L = \{w \in \Sigma^*; \delta^*(q_0, w) \in F\}$ a pro každý stav $q \in Q$ existuje prefix x_q slova w tak, že $\delta^*(q_0, x_q) = q$. Tento jazyk L je regulární.

M Označme $M = L(A)$.

T Definujme novou abecedu T trojic $\{[paq]; p, q \in Q, a \in \Sigma, \delta(p, a) = q\}$.

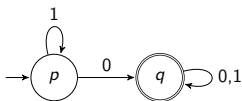
h Definujme homomorfismus $(\forall p, q, a) h([paq]) = a$.

L_1 Jazyk $L_1 = h^{-1}(M)$ je regulární, protože M je regulární (DFA inverzní homomorfismus).

- $h^{-1}(101)$ obsahuje $2^3 = 8$ řetězců, např.

$$[p1p][q0q][p1p] \in \{[p1p], [q1q]\}\{[p0q], [q0q]\}\{[p1p], [q1q]\}.$$

- Dále zkonstruujeme L z L_1 (další slide).



L_2 Vynutíme začátek q_0 . Definujeme

$$E_1 = \bigcup_{a \in \Sigma, q \in Q} \{[q_0 a q]\} =$$

$$E_1 = \{[q_0 a_1 q_0], [q_0 a_2 q_1], \dots, [q_0 a_m q_n]\}.$$

$$\text{Pak } L_2 = L_1 \cap L(E_1 \cdot T^*).$$

L_3 Vynutíme stejné sousedící stavy.

Definujeme ne-odpovídající dvojice

$$E_2 = \bigcup_{q \neq r, p, q, r, s \in Q, a, b \in \Sigma} \{[p a q][r b s]\}.$$

$$\text{Definujeme } L_3 = L_2 - L(T^* \cdot E_2 \cdot T^*),$$

- Končí v přijímajícím stavu, protože jsme začali z jazyku M přijímaném DFA A .

L_4 Všechny stavy. $\forall q \in Q$ definujeme E_q jako regulární výraz sjednocení všech symbolů T takových, že q není ani na první, ani na poslední pozici. Odečteme $L(E_q^*)$ od L_3 . $L_4 = L_3 - \bigcup_{q \in Q} \{E_q^*\}.$

L Odstraníme stavy, necháme symboly.
 $L = h(L_4)$. Tedy L je regulární.

Přehled:

$$M = L(A)$$

Inverzní homom.

$$L_1 \quad h^{-1}(M) \subseteq \{[qap]\}^*$$

průnik RJ

$$L_2 \quad + q_0$$

rozdíl RJ

$$L_3 \quad + \text{sousední stavy rovný}$$

rozdíl RJ

$$L_4 \quad + \text{všechny stavy}$$

homomorfismus

$$L \quad h([qap]) = a$$

Rozhodovací problémy pro regulární jazyky

Lemma (Test ne-prázdnoti regulárního jazyka)

Lze algoritmicky rozhodnout, zda jazyk přijímaný DFA, NFA, ϵ -NFA je prázdný.

Jazyk je prázdný právě když žádný z koncových stavů není dosažitelný.
Dosažitelnost lze testovat $O(n^2)$.

Lemma (Test náležitosti do regulárního jazyka)

Pro daný řetězec w ; $|w| = n$ a regulární jazyk L . Lze algoritmicky rozhodnout, zda je $w \in L$.

- DFA: Spust automat; pokud $|w| = n$, při dobré reprezentaci a konstatním čase přechodu $O(n)$.
- NFA o s stavech: čas $O(ns^2)$. Každý vstupní symbol aplikujeme na všechny stavy předchozího kroku, kterých je nejvýš s .
- ϵ -NFA - nejdříve určíme ϵ -uzávěr. Pak aplikujeme přechodovou funkci a ϵ -uzávěr na výsledek.

Shrnutí 4

Definition (4.1 RJ – algebraický popis jazyků)

Pro konečnou neprázdnou abecedu Σ označme $RJ(\Sigma)$ nejmenší třídu jazyků, která:

- obsahuje prázdný jazyk \emptyset
- pro každé písmeno $x \in \Sigma$ obsahuje jazyk $\{x\}$
- je uzavřená na sjednocení $A, B \in RJ(\Sigma) \Rightarrow A \cup B \in RJ(\Sigma)$
- je uzavřená na zřetězení $A, B \in RJ(\Sigma) \Rightarrow A.B \in RJ(\Sigma)$
- je uzavřená na iteraci $A \in RJ(\Sigma) \Rightarrow A^* \in RJ(\Sigma)$.

Theorem (4.1 Kleene)

Libovolný jazyk je rozpoznatelný konečným automatem právě když je ve třídě RJ.

Třída regulárních jazyků je uzavřená na

- sjednocení, průnik, doplněk
- zřetězení, iteraci, substituci, homomorfizmus, inverzní homomorfizmus
- reverzi, levý i pravý kvocient.

Přehled kapitol

- 1 Úvod, Iterační lemma pro reg. jazyky
- 2 Redukovaný DFA a ekvivalence automatů, stavů
- 3 Nedeterministické ϵ -NFA, Operace zachovávající regularitu
- 4 Regulární výrazy, Kleeneova věta, Substitute, Homomorfizmus
- 5 Dvousměrné FA, Mealy a Moore stroje
- 6 Gramatiky, Chomského hierarchie, víceznačnost
- 7 Chomského NF, Pumping Lemma pro CFL
- 8 CYK – náležení do CFL
- 9 Zásobníkové automaty, Deterministické PDA
- 10 Uzávěrové vlastnosti, Dykovy jazyky
- 11 Turingův stroj, rozšíření
- 12 Lineárně omezené automaty, Univerzální TM, Diagonální jazyk
- 13 Nerozhodnutelné problémy, Postův korespondenční p.
- 14 Časová složitost