

Regulární výrazy

Definice 2.58. Množina **regulárních výrazů** nad abecedou Σ , označovaná $RE(\Sigma)$, je definována induktivně takto:

- 1 ε , \emptyset a a pro každé $a \in \Sigma$ jsou regulární výrazy nad Σ (tzv. *základní regulární výrazy*).
- 2 Jsou-li E, F regulární výrazy nad Σ , jsou také $(E.F)$, $(E + F)$ a (E^*) regulární výrazy nad Σ .
- 3 Každý regulární výraz vznikne po konečném počtu aplikací kroků 1-2.

Každý regulární výraz E nad abecedou Σ **popisuje** (jednoznačně určuje) **jazyk** $L(E)$ nad abecedou Σ podle těchto pravidel:

$$L(\varepsilon) \stackrel{\text{def}}{=} \{\varepsilon\}$$

$$L(\emptyset) \stackrel{\text{def}}{=} \emptyset$$

$$L(a) \stackrel{\text{def}}{=} \{a\} \text{ pro každé } a \in \Sigma$$

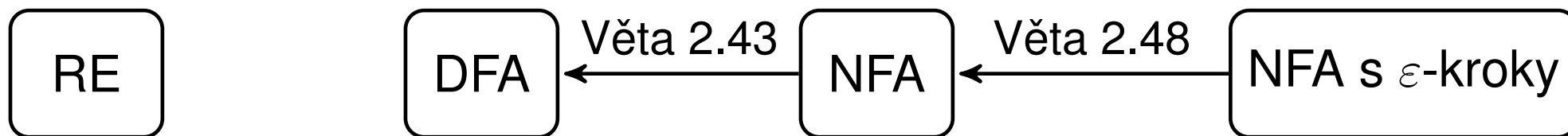
$$L(E.F) \stackrel{\text{def}}{=} L(E).L(F)$$

$$L(E + F) \stackrel{\text{def}}{=} L(E) \cup L(F)$$

$$L(E^*) \stackrel{\text{def}}{=} L(E)^*$$

Příklady

Převod regulárních výrazů na konečné automaty



Věta 2.60. Necht' E je regulární výraz. Pak existuje konečný automat rozpoznávající $L(E)$.

Důkaz. Pro libovolnou abecedu Σ lze zkonstruovat konečné automaty, které rozpoznávají jazyky popsané základními regulárními výrazy, tj. \emptyset , $\{\varepsilon\}$ a $\{a\}$ pro každé $a \in \Sigma$. Tvrzení věty pak plyne z uzavřenosti jazyků rozpoznatelných konečnými automaty vůči operacím sjednocení, zřetězení a iteraci. □

Příklad

Regulární přechodový graf

Definice 2.64. Regulární přechodový graf \mathcal{M} je pětice $(Q, \Sigma, \delta, I, F)$, kde

- Q je neprázdňá konečňá množina stavů,
- Σ je vstupní abeceda,
- $\delta : Q \times Q \rightarrow RE(\Sigma)$ je parciální přechodová funkce,
- $I \subseteq Q$ je množina počátečních stavů,
- $F \subseteq Q$ je množina koncových stavů.

Příklad

Slovo $w \in \Sigma^*$ je **akceptováno** grafem \mathcal{M} , právě když

- existuje posloupnost stavů q_0, \dots, q_n , kde $n \geq 0$, $q_0 \in I$, $q_n \in F$
- a $\delta(q_{i-1}, q_i)$ je definováno pro každé $0 < i \leq n$

taková, že

- w lze rozdělit na n částí $w = v_1 \dots v_n$ tak, že
- $v_i \in L(\delta(q_{i-1}, q_i))$ pro každé $0 < i \leq n$.

Slovo ε je akceptováno také tehdy, je-li $I \cap F \neq \emptyset$.

Převod regulárního přechodového grafu na NFA

Motivace

Věta 2.65. Pro libovolný regulární přechodový graf $\mathcal{M} = (Q, \Sigma, \delta, I, F)$ existuje ekvivalentní NFA \mathcal{M}' s ε -kroky.

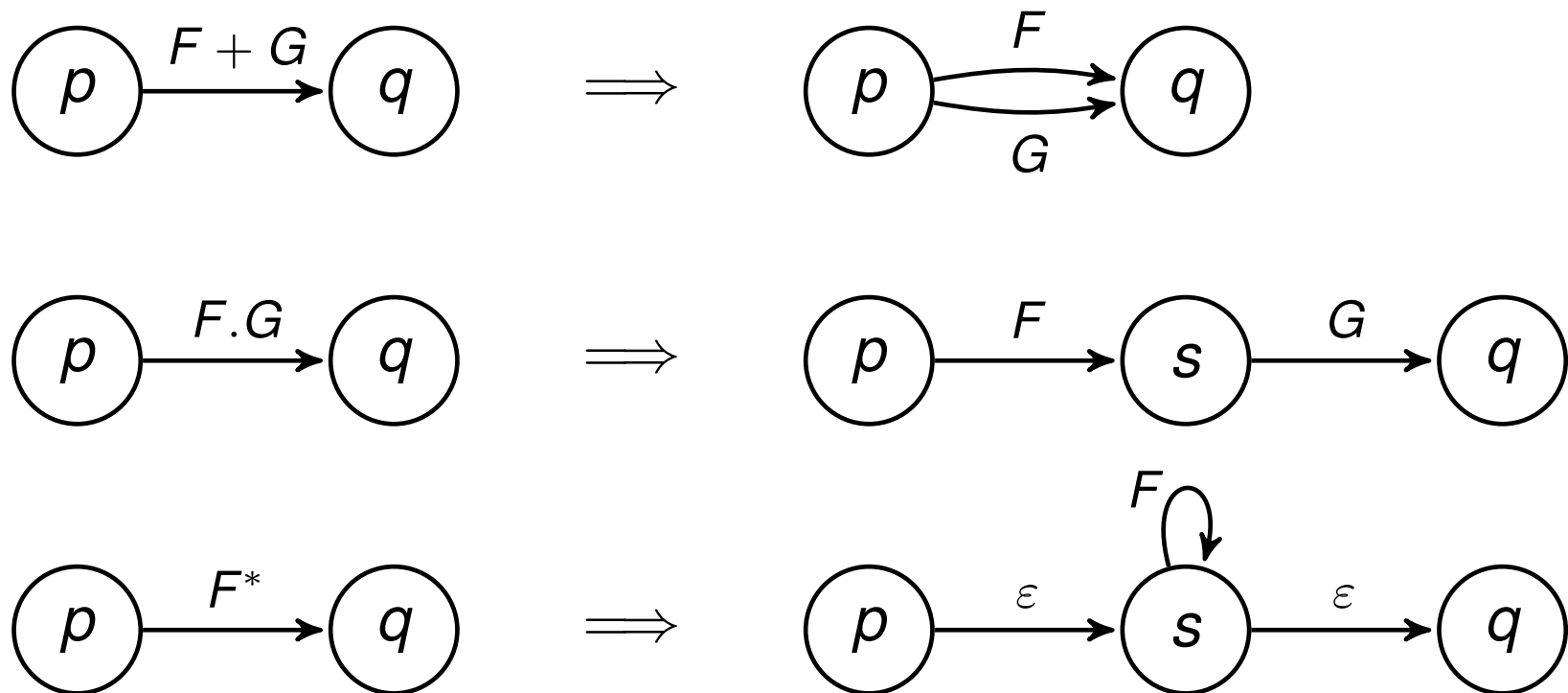
Důkaz. Algoritmus konstrukce NFA \mathcal{M}' s ε -kroky.

Krok 1: Ke grafu \mathcal{M} přidáme nový stav q_0 a hranu $q_0 \xrightarrow{\varepsilon} q$ pro každé $q \in I$. Stav q_0 bude (jediným) počátečním stavem automatu \mathcal{M}' , prvky F jeho koncovými stavy.

Krok 2: Opakovaně realizuj kroky **(a)** a **(b)** dokud přechodový graf obsahuje alespoň jednu hranu ohodnocenou symbolem, který nepatří do $\Sigma \cup \{\varepsilon\}$, tedy je tvaru $F + G$, $F.G$, F^* nebo \emptyset .

(a) Odstraň všechny hrany, které jsou ohodnoceny symbolem \emptyset .

(b) Vyber libovolnou hranu $p \xrightarrow{E} q$, kde $E \notin \Sigma \cup \{\varepsilon\}$, odstraň ji a proved' následující:



Konečnost algoritmu \mathcal{M} obsahuje pouze konečně mnoho hran, tj. konečně mnoho regulárních výrazů. Každý regulární výraz obsahuje jen konečně mnoho výskytů $+$, $.$ a $*$. V každém kroku 2(b) jeden výskyt odstraníme.

Korektnost algoritmu

- Výsledný graf je přechodovým grafem nedeterministického konečného automatu \mathcal{M}' s ε -kroky.
- Přímo z definice regulárního přechodového grafu se snadno ověří, že kroky 1 a 2 popsaného transformačního algoritmu zachovávají ekvivalenci automatů, proto $L(\mathcal{M}) = L(\mathcal{M}')$. □

Převod DFA na regulární výraz

Motivace

Věta 2.66. Pro každý regulární přechodový graf $\mathcal{M} = (Q, \Sigma, \delta, I, F)$ existuje ekvivalentní přechodový graf $\mathcal{M}' = (\{x, y\}, \Sigma, \delta', \{x\}, \{y\})$, kde δ' může být definováno pouze pro dvojici (x, y) .

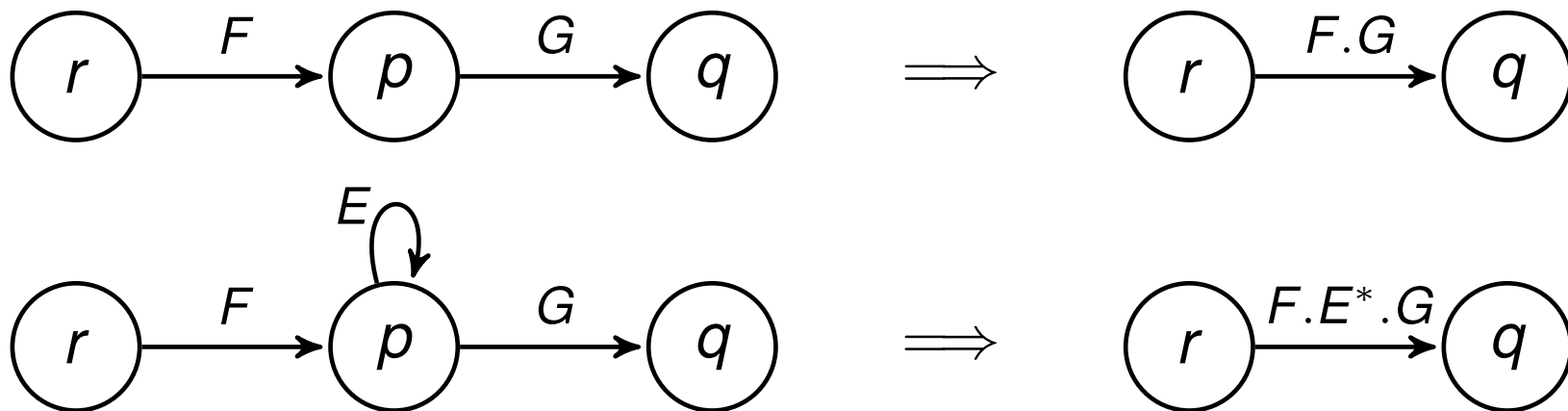
Důkaz. Algoritmus transformace

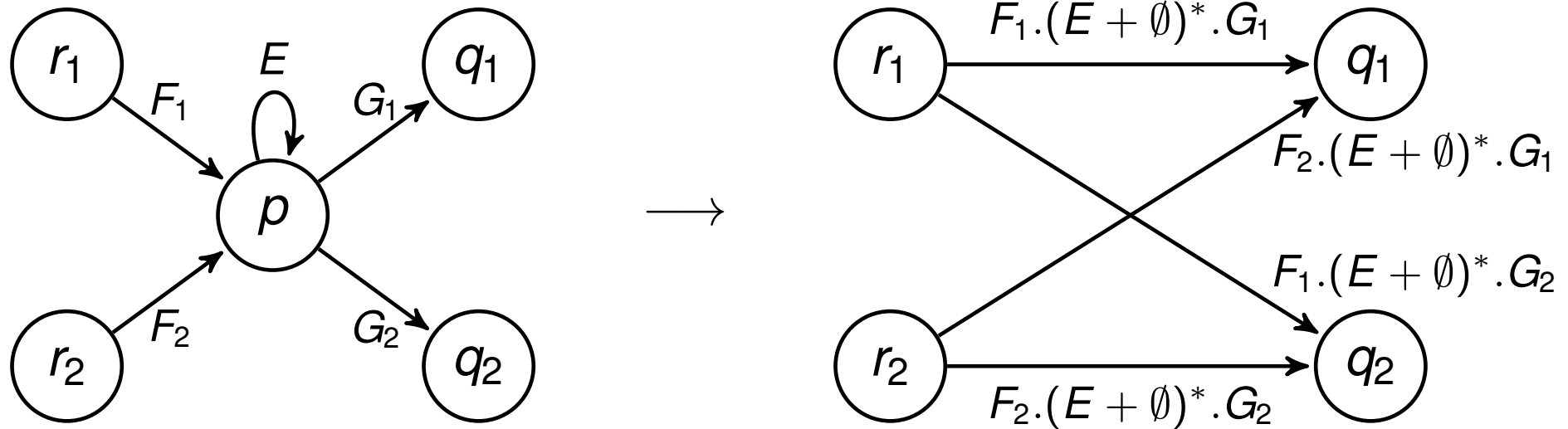
Krok 1: Ke grafu \mathcal{M} přidáme nový počáteční stav x a nový koncový stav y . Přidáme také hrany $x \xrightarrow{\varepsilon} q$ pro každé $q \in I$ a $r \xrightarrow{\varepsilon} y$ pro každé $r \in F$.

Krok 2: Každý stav p různý od x, y nyní odstraníme spolu s hranami, které do p vcházejí nebo z p vycházejí.

Pokud do p nevede hrana z jiného uzlu, je nedosažitelný z počátečního stavu. Pokud z p nevede hrana do jiného uzlu, nelze z p dosáhnout koncový stav. V obou případech p odstraníme bez náhrady.

Pro každou dvojici vstupní hrany vedoucí do p z jiného uzlu a výstupní hrany vedoucí z p do jiného uzlu přidáme přímý přechod. Pak p odstraníme.





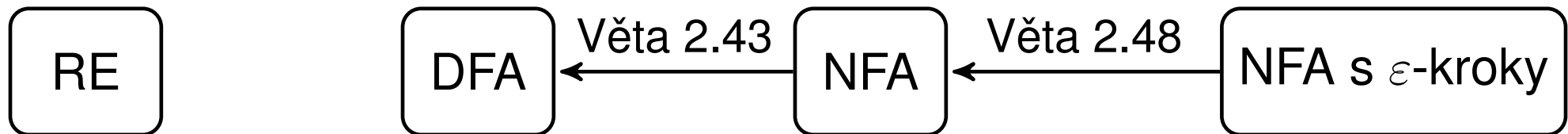
Po odstranění všech stavů různých od x a y zůstanou tyto dva stavy spolu s (žádnou nebo jednou) hranou z x do y .

Konečnost algoritmu Každým krokem 2 snížíme počet stavů.

Korektnost algoritmu Z definice regulárního přechodového grafu přímo ověříme, že kroky 1 i 2 zachovávají ekvivalenci. □

Na regulární výraz lze stejně převést každý konečný automat (nejen DFA).

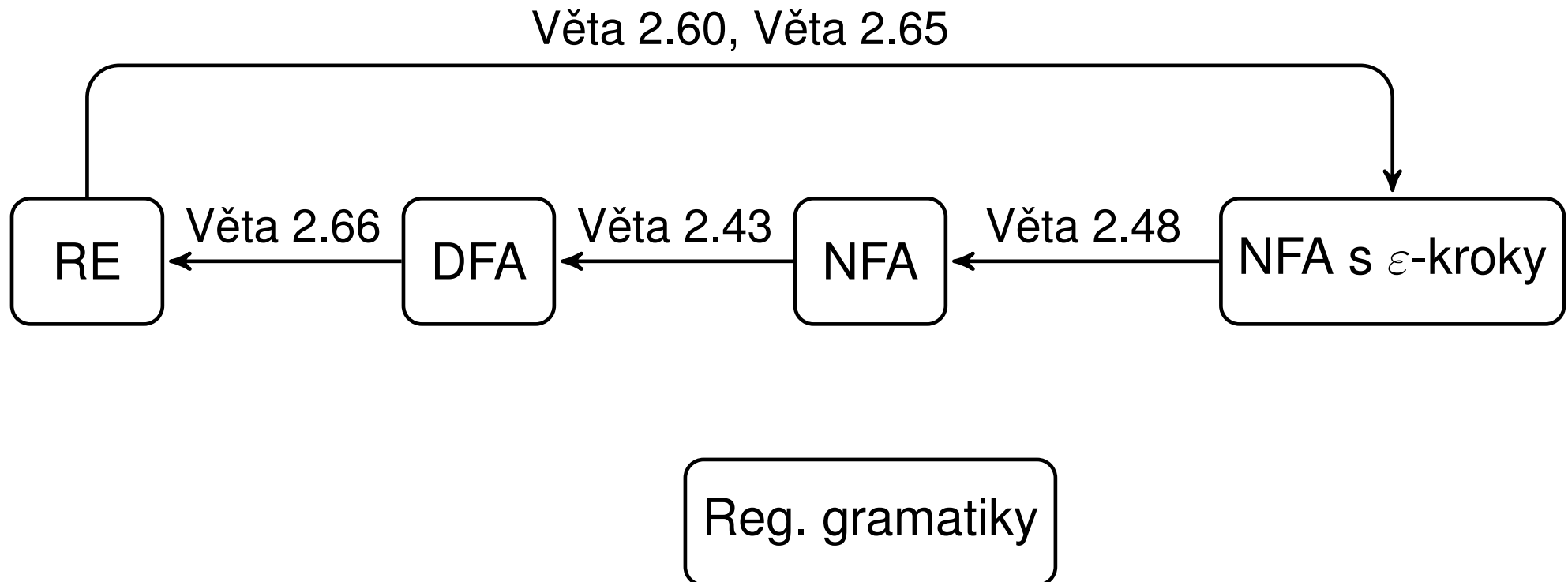
Ekvivalence konečných automatů a reg. výrazů



Kleeneho věta 2.63. Libovolný jazyk je popsateľný regulárním výrazem právě když je rozpoznateľný konečným automatem.

Ekvivalentní formulace: Třída jazyků rozpoznateľných konečnými automaty je nejmenší třída, která obsahuje všechny konečné množiny a je uzavřena na sjednocení, zřetězení a iteraci.

Ekvivalence konečných automatů a regulárních gramatik



Pojem regulárního jazyka byl definován dvakrát – nejprve pomocí regulární gramatiky a pak ještě jednou pomocí konečného automatu.

Převod regulární gramatiky na konečný automat

Lemma 2.69. Ke každé regulární gramatice $\mathcal{G} = (N, \Sigma, P, S)$ existuje nedeterministický konečný automat $\mathcal{M} = (Q, \Sigma, \delta, q_0, F)$ takový, že $L(\mathcal{G}) = L(\mathcal{M})$.

Důkaz.

Konstrukce konečného automatu $\mathcal{M} = (Q, \Sigma, \delta, q_0, F)$

- $Q = \{\bar{A} \mid A \in N\} \cup \{q_f\}$, kde $q_f \notin N$
- $q_0 = \bar{S}$
- δ je nejmenší funkce $Q \times \Sigma \rightarrow 2^Q$ splňující:
 - pokud $A \rightarrow aB$ je pravidlo v P , pak $\bar{B} \in \delta(\bar{A}, a)$
 - pokud $A \rightarrow a$ je pravidlo v P , kde $a \neq \varepsilon$, pak $q_f \in \delta(\bar{A}, a)$
- $F = \begin{cases} \{\bar{S}, q_f\} & \text{pokud } S \rightarrow \varepsilon \text{ je pravidlo v } P \\ \{q_f\} & \text{jinak} \end{cases}$

Korektnost Nejprve indukcí vzhledem ke k dokážeme, že pro každé $a_1, \dots, a_k \in \Sigma$ a $B \in N$ platí:

$$S \Rightarrow^* a_1 \dots a_k B \iff \bar{B} \in \hat{\delta}(\bar{S}, a_1 \dots a_k)$$

■ **Základní krok $k = 0$:** Z definice $\hat{\delta}$ plyne $\hat{\delta}(\bar{S}, \varepsilon) = \{\bar{S}\}$ a proto:

$$S \Rightarrow^* B \iff B = S \iff \bar{B} = \bar{S} \iff \bar{B} \in \hat{\delta}(\bar{S}, \varepsilon)$$

■ **Indukční krok:**

$$S \Rightarrow^* a_1 \dots a_{k+1} B$$

$$\iff \exists C \in N \text{ takové, že } S \Rightarrow^* a_1 \dots a_k C \Rightarrow a_1 \dots a_{k+1} B$$

$$\iff \exists C \in N \text{ takové, že } \bar{C} \in \hat{\delta}(\bar{S}, a_1 \dots a_k) \wedge C \rightarrow a_{k+1} B$$

$$\iff \exists C \in N \text{ takové, že } \bar{C} \in \hat{\delta}(\bar{S}, a_1 \dots a_k) \wedge \bar{B} \in \delta(\bar{C}, a_{k+1})$$

$$\iff \bar{B} \in \hat{\delta}(\bar{S}, a_1 \dots a_{k+1}).$$

Dokázali jsme: $S \Rightarrow^* a_1 \dots a_k B \iff \bar{B} \in \hat{\delta}(\bar{S}, a_1 \dots a_k)$

Ukážeme, že $w \in L(\mathcal{G}) \iff w \in L(\mathcal{M})$:

■ $w = \varepsilon$:

$$\varepsilon \in L(\mathcal{G}) \iff S \rightarrow \varepsilon \in P \iff \bar{S} \in F \iff \varepsilon \in L(\mathcal{M})$$

■ $w = va$, kde $v \in \Sigma^*$, $a \in \Sigma$:

$$\begin{aligned} va \in L(\mathcal{G}) &\iff S \Rightarrow^* vB \Rightarrow va \\ &\iff S \Rightarrow^* vB \wedge B \rightarrow a \in P \\ &\iff \bar{B} \in \hat{\delta}(\bar{S}, v) \wedge q_f \in \delta(\bar{B}, a) \\ &\iff q_f \in \hat{\delta}(\bar{S}, va) \iff va \in L(\mathcal{M}) \end{aligned}$$

□

Převod konečného automatu na regulární gramatiku

Lemma 2.71 Pro každý konečný automat $\mathcal{M} = (Q, \Sigma, \delta, q_0, F)$ existuje regulární gramatika $\mathcal{G} = (N, \Sigma, P, S)$ taková, že $L(\mathcal{M}) = L(\mathcal{G})$.

Důkaz.

Bez újmy na obecnosti předpokládejme, že \mathcal{M} je nedeterministický.

- $N = \{\bar{q} \mid q \in Q\} \cup \{S\}$, kde $S \notin Q$.
- P je nejmenší množina pravidel splňující:
 - pokud $p \in \delta(q, a)$, je $\bar{q} \rightarrow a\bar{p}$ pravidlo v P
 - pokud $p \in \delta(q, a)$ a $p \in F$, je $\bar{q} \rightarrow a$ pravidlo v P
 - pokud $p \in \delta(q_0, a)$, je $S \rightarrow a\bar{p}$ pravidlo v P
 - pokud $p \in \delta(q_0, a)$ a $p \in F$, je $S \rightarrow a$ pravidlo v P
 - pokud $q_0 \in F$, je $S \rightarrow \varepsilon$ pravidlo v P

Gramatika $\mathcal{G} = (N, \Sigma, P, S)$ je zřejmě regulární.

Platí: $\hat{\delta}(q_0, a_1 \dots a_k) \cap F \neq \emptyset$, kde $k \geq 0$, $a_1, \dots, a_k \in \Sigma$

$$\iff S \Rightarrow^* a_1 \dots a_k$$



Rozhodnutelné problémy pro třídu reg. jazyků

Regulární jazyk – popsán některým z uvažovaných formalismů.

Otázky: Máme-li dány konečné automaty \mathcal{M} a \mathcal{M}' nad Σ

- **ekvivalence:** jsou \mathcal{M} a \mathcal{M}' ekvivalentní? (platí $L(\mathcal{M})=L(\mathcal{M}')$?)
- **inkluze** (jazyků): platí $L(\mathcal{M}) \subseteq L(\mathcal{M}')$?
- **příslušnost** (slova k jazyku): je-li dáno $w \in \Sigma^*$, platí $w \in L(\mathcal{M})$?
- **prázdnot** (jazyka): je $L(\mathcal{M}) = \emptyset$?
- **univerzalita** (jazyka): je $L(\mathcal{M}) = \Sigma^*$?
- **konečnost** (jazyka): je $L(\mathcal{M})$ konečný jazyk?

Věta 2.74 Problém **prázdnosti** ($L(\mathcal{M}) \stackrel{?}{=} \emptyset$) a problém **univerzality** ($L(\mathcal{M}) \stackrel{?}{=} \Sigma^*$) jsou rozhodnutelné pro regulární jazyky.

Důkaz. $L(\mathcal{M})$ je prázdný, právě když mezi dosažitelnými stavy automatu \mathcal{M} není žádný koncový stav.

Univerzalita: $L(\mathcal{M}) = \Sigma^* \iff co-L(\mathcal{M}) = \emptyset.$ □

Věta 2.77 Problém **ekvivalence** je rozhodnutelný pro regulární jazyky.

Důkaz. Pro libovolné L_1, L_2 platí:

$$(L_1 = L_2) \iff (L_1 \cap co-L_2) \cup (co-L_1 \cap L_2) = \emptyset.$$

Pro L_1, L_2 zadané automaty lze uvedené operace algoritmicky realizovat.
Alternativně: minimalizace a kanonizace. □

Věta 2.76 Problém, zda jazyk L zadaný automatem \mathcal{M} je **konečný**, resp. **nekonečný**, je rozhodnutelný.

Důkaz. Nechť \mathcal{M} je DFA. L je nekonečný právě když \mathcal{M} akceptuje alespoň jedno slovo $w \in \Sigma^*$ s vlastností $n \leq |w| < 2n$, kde $n = \text{card}(Q)$.

(\implies) Je-li L nekonečný, pak existuje $u \in L$ takové, že $|u| \geq n$. Je-li $|u| < 2n$, jsme hotovi. Nechť $|u| \geq 2n$. Z lemma o vkládání plyne, že $u = xyz$, kde $1 \leq |y| \leq n$ a $xz \in L$. Platí $|xz| \geq n$. Pokud $|xz| \geq 2n$, celý postup opakujeme.

(\impliedby) Je-li $|w| \geq n$, pak \mathcal{M} při čtení w musí projít dvakrát stejným stavem. Proto $w = xyz$ tak, že $|y| \geq 1$ a platí $xy^i z \in L$ pro každé $i \in \mathbb{N}_0$ (viz důkaz lemmatu o vkládání), tedy L je nekonečný.

Existenci $w \in L$ takového, že $n \leq |w| < 2n$, lze algoritmicky ověřit (slov je konečně mnoho, „vyzkoušíme“ každé z nich). □

Aplikace reg. jazyků a konečných automatů

- **Vyhledávání vzorů (pattern matching)** v textu (editory, textové systémy), DNA sekvencích, ...
Například v Unixu:
 - `grep` - vyhledávání podle zadaného regulárního výrazu
 - `egrep` - vyhledávání podle zadaného rozšířeného regulárního výrazu
 - `fgrep` - vyhledávání podle zadaného řetězce
- **Zpracování lexikálních jednotek** například při automatizované konstrukci překladačů (`lex`, `flex`)
- **Zpracování obrazů (image processing)**
- **Konečné automaty nad nekonečnými slovy**
- **Specifikace a verifikace** konečně stavových systémů
- **Konečné automaty s výstupem**