1 Úvod

Poznámka (Historie)

- První formalizace pojmu algoritmus – Ada, Countess of Lovelace 1852.
- Intenzivnější vývoj s rozvojem počítačů ve 2. čtvrtině 20. století.
- Co stroje umí a co ne? – Church, Turing, Kleene (konečné automaty / neuronové sítě), Post, Markov, Chomsky (zásobníkové automaty a formální teorie konečných automatů, zkoumal Angličtinu).

Poznámka (Cíl)
Osvojit si abstraktní model počítače, vnímat jak drobné změny v definici vedou k velmi rozdílným důsledkům. Zažít skutečnost alg. nerozhodnutelných problémů a připravit se na přednášku o složitosti a NP-úplnosti.

Poznámka (Praktické využití)
Korektnost algoritmů, zpracování přirozeného jazyka, lexikální a syntaktická analýza v překladačích. Návrh, popis a verifikace hardwaru (automaty, integrované obvody, stroje). Vyhledávání v textu atd.

2

Definice 2.1 (Deterministický konečný automat (DFA))

Deterministický konečný automat $A = (Q, \Sigma, \delta, q_0, F)$ sestává z: konečné množiny stavů (Q), konečné neprázdné množiny vstupních symbolů (abecedy, Σ), přechodové funkce, tj. zobrazení $Q \times \Sigma \to Q$ (značí se hranami grafu, δ), počátečního stavu (vede do něj šipka 'odnikud', $q_0 \in Q$) a neprázdné množiny (přijímajících) stavů (značí se dvojitým kruhem / šipku 'ven', $F \subseteq Q$).

Úmluva
Přidáváme 0-2 stavy: fail (pokud je nějaký přechod nedefinován, vede sem a všechno z fail vede do fail) a final (pokud je F prázdné, všechny šipky z něj vedou zpět do něj).

Definice 2.2 (Slovo, jazyk)

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^* a množinu všech neprázdných Σ^+.

Jazyk $L \subseteq \Sigma^*$ je množina slov v abecedě Σ.

Definice 2.3 (Operace: zřetězení, mocnina, délka slova)

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

Definice 2.4 (Rozšířená přechodová funkce)

Mějme přechodovou funkci $\delta : Q \times \Sigma \to Q$. Rozšířenou přechodovou funkci $\delta^* : Q \times \Sigma^* \to Q$ (tranzitivní uzávěr δ) definujeme induktivně: $\delta^*(q, \lambda) = q$ a $\delta^*(q, wx) = \delta(\delta^*(q, w), x)$ pro $x \in \Sigma$ a $w \in \Sigma^*$.

Definice 2.5 (Jazyky rozpoznatelné konečnými automaty, regulární jazyky)

Jazykem rozpoznávaným (akceptovaným, přijímaným) konečným automatem A nazveme jazyk $L(A) = \{w | w \in \Sigma^* \wedge \delta^*(q_0, w) \in F\}$.

Jazyk je rozpoznatelný konečným automatem, jestliže existuje konečný automat A takový, že $L = L(A)$.

Třídu jazyků rozpoznatelných konečnými automaty označíme $\mathcal{F}$ a nazveme ji regulární jazyky.

Věta 2.1 (!Iterační (pumpin) 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 $y \neq \lambda \wedge |xy| \leq n \wedge \forall k \in \mathbb{N}_0$, slovo xy^kz je také v L.

Důkaz

Mějme regulární jazyk L, pak existuje DFA A s n stavy, že $L = L(A)$. Vezmeme libovolné slovo $a_1a_2 \ldots a_n \ldots a_m = w \in L$ délky $m \geq n$, $a_i \in \Sigma$. Následně definujeme $\forall i : p_i = \delta^*(q_0, a_1a_2 \ldots a_i)$. Platí $p_0 = q_0$. Z Dirichletova principu se některý stav opakuje. Vezmeme první takový, tj. $(\exists i, j)(0 \leq i < j \leq n \wedge p_i = p_j)$.

Definujeme $x = a_1a_2 \ldots a_i$, $y = a_{i+1}a_{i+2} \ldots a_j$ a $z = a_{j+1}a_{j+2} \ldots a_m$, tj. $w = xyz$, $y \neq \lambda$, $|xy| \leq n$. □

Definice 2.6 (Kongruence, konečný index)

Mějme konečnou abecedu Σ a relaci ekvivalence $\sim$ na Σ^*. Potom $\sim$ je pravá kongruence, jestliže $\forall u, v, w \in \Sigma^* : u \sim v \implies uw \sim vw$. $\sim$ je konečného indexu, jestliže rozklad $\Sigma^*/\sim$ má konečný počet tříd.

Třídu kongruence $\sim$ obsahující slovo u značíme $[u]_\sim$, resp. $[u]$.

Věta 2.2 (Myhill-Nerodova věta)

Nechť L je jazyk nad konečnou abecedou Σ. Potom L je rozpoznatelný konečným automatem právě tehdy, když existuje pravá kongruence $\sim$ konečného indexu nad Σ^* tak, že L je sjednocením jistých tříd rozkladu $\Sigma^*/\sim$.

Důkaz

$\Longrightarrow$ definujeme $u \sim v \equiv \delta^*(q_0, u) = \delta^*(q_0, v)$. Zřejmě je to ekvivalence. Je to pravá kongruence (z definice δ^*) a má konečný index (jelikož automat má konečně mnoho stavů).

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

$\Rightarrow$ abeceda automatu bude Σ. Stavy budou třídy rozkladu $\Sigma^*/\sim$. Počáteční stav je $q_0 = [\lambda]_\sim$. Koncové stavy $F = \{c_1, \dots, c_n\}$, kde $L = \bigcup_{i \in [n]} c_i$. Přechodová funkce $\delta([u], x) = [ux]$ (korektní z definice pravé kongruence). $\square$

Příklad

$L = \{u|u = a^+ b^i c^i \wedge u = b^i c^j \wedge i, j \in \mathbb{N}_0\}$ není regulární, ale vždy lze pumpovat první písmeno.

Důkaz (Sporem)

Předpokládejme, že L je regulární. Pak existuje pravá kongruence $\sim$ konečného indexu m, L je sjednocení některých tříd $\Sigma^*/\sim$. Vezmeme množinu slov $S = \{ab, abb, abbb, \dots, ab^{m+1}\}$. Existují dvě slova (Dirichletův princip) $i \neq j$, která padnou do stejné třídy. $ab^i \sim ab^j \Leftrightarrow ab^i c^i \sim ab^j c^i$, ale $ab^i c^i \in L \wedge ab^j c^i \notin L$. $\lightning$. $\square$

Definice 2.7 (Dosažitelné stavy)

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

Poznámka (Hledání dosažitelných stavů)
'Hloupé' prohledávání do šířky.

Definice 2.8 (Automatový homomorfismus)

Nechť A_1, A_2 jsou DFA se standardním označením a shodnou abecedou. Řekněme, že zobrazení $h : Q_1 \to Q_2$ je automatovým homomorfismem, jestliže $h(q_{1_0}) = q_{2_0}$, $h(\delta_1(q, x)) = \delta_2(h(q), x)$ a $q \in F_1 \Leftrightarrow h(q) \in F_2$.

Definice 2.9 (Ekvivalence automatů)

Dva konečné automaty nad stejnou abecedou jsou ekvivalentní, jestliže rozpoznávají stejný jazyk.

Věta 2.3 (O ekvivalenci automatů)

Existuje-li homomorfismus konečných automatů, pak jsou tyto automaty ekvivalentní.

Důkaz
Triviální. □

Definice 2.10 (Ekvivalence stavů)

Dva stavy jsou ekvivalentní, pokud pro všechna slova dojdeme z obou stavů buď do nepřijímajících, nebo do přijímajících stavů. Pokud dva stavy nejsou ekvivalentní, říkáme, že jsou rozlišitelné.

Poznámka (Algoritmus pro nalezení eqvivalentních stavů)
Vytvořím tabulku dvojic stavů a zaškrtám zřejmě rozlišitelné dvojice (přijímající + nepřijímající). Potom pro každou dvojici zkusím všechna písmena a pokud nějaké z nich posune ze stavů do rozlišitelné, pak i tato dvojice je rozlišitelná. Opakuji, dokud se něco mění.

Definice 2.11 (Redukovaný DFA, redukt)

DFA je redukovaný, pokud nemá nedosažitelné stavy a žádné dva stavy nejsou ekvivalentní. DFA B je reduktem A, jestliže B je redukovaný a B a A jsou ekvivalentní.

Poznámka (Algoritmus na testování ekvivalence reg. jazyků)
Najdeme jeden a druhý DFA rozpoznávající jeden a druhý jazyk. BÚNO jsou stavy disjunktní. Vytvoříme DFA sjednocením (za počáteční stav vezmeme libovolný z 2 počátečních stavů našich DFA). Potom jsou jazyky ekvivalentní, když jsou ekvivalentní počáteční stavy našich DFA.

3 NFA

Definice 3.1 (Nederministický konečný automat (NFA))

NFA je DFA, kde přechodová funkce je funkce do potenční množiny stavů. A počáteční stav může být také množina, ale existují obě alternativy.

Definice 3.2 (Rozšířená přechodová funkce)

Pro přechodovou funkci δ NFA je rozšířená přechodová funkce $\delta^* : Q \times \Sigma^* \to \mathcal{P}(Q)$ definovaná indukcí: $\delta^*(q, \lambda) = \{q\}$ a $\delta^*(q, wx) = \bigcup_{p \in \delta^*(q,w)} \delta(p, x)$.

Definice 3.3 (Jazyk přijímaný NFA)

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.

Poznámka (Algoritmus: podmnožinová konstrukce)
Začínáme 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 $L(N) = L(D)$.

Q_D je množina podmnožin Q_N ($Q_D = \mathcal{P}(Q_n)$). Počáteční stav DFA označený S_0 je prvek Q_D. $F_D = \{S | S \in \mathcal{P}(Q_n) \wedge S \cap F_N \neq \emptyset\}$. Přechodová funkce je ($S \in Q_D$, $a \in \Sigma$):

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

Důkaz
Triviální, indukcí dokážeme shodné chování $d*$. $\square$

Definice 3.4 (λ-NFA)

λ-NFA (NFA s λ přechody) je NFA, kde δ je definována pro $Q \times (\Sigma \cup \{\lambda\})$.

Definice 3.5 (λ-uzávěr)

Pro $q \in Q$ definujeme λ-uzávěr stavu q (v těchto poznámkách značeno $\overline{q}$) rekurzivně: $q \in \overline{q}$. Je-li $p \in \overline{q}$ a $r \in \delta(p, \lambda)$, pak i $r \in \overline{q}$.

Pro $S \subseteq Q$ definujeme $\overline{S} = \bigcup_{q \in S} \overline{q}$.

Definice 3.6 (Rozšířená přechodová funkce)

$\delta^*(q, \lambda) = \overline{q}$. $\delta^*(q, wa) = \overline{\bigcup_{\delta^*(q,w)} \delta(p, a)}$.

Věta 3.1

Jazyk je rozpoznatelný λ-NFA $\Leftrightarrow$ L regulární.

Důkaz
$\Leftarrow$: triviální. $\Longrightarrow$: přes podmnožinovou konstrukci. $\square$

4 Množinové operace nad jazyky

Definice 4.1 (Množinové operace nad jazyky)

Mějme jazyky L, M. Definujeme následující operace:

- binární (konečné) sjednocení $L \cup M = \{w|w \in L \vee w \in M\}$,
- průnik $L \cap M = \{w|w \in L \wedge w \in M\}$,
- rozdíl $L - M = \{w|w \in L \wedge w \notin M\}$,
- doplněk (komplement) $\overline{L} = -L = \{w|w \notin L\} = \Sigma^* - L$.

Věta 4.1 (Uzavřenost na množinové operace)

Regulární jazyky jsou uzavřené na 4 operace výše.

Důkaz

Doplněk: doplníme všechny přechody (doplníme FAIL stav). Potom prohodíme přijímající a nepřijímající stavy.

Průnik sjednoceni a rozdíl přes tzv. součinový automat $(Q_1 \times Q_2, \Sigma, \delta', (q_0, q_1), F)$, kde $\delta'((p_1, p_2), x) = (\delta_1(p_1, x), \delta_2(p_2, x))$ a F je podle toho, zda řešíme průnik, sjednocení nebo rozdíl, $F_1 \times F_2$, $(F_1 \times Q_2) \cup (Q_1 \times F_2)$ nebo (po doplnění) $F_1 \times (Q_2 - F_2)$. $\square$

Definice 4.2 (Řetězcové operace nad jazyky)

Mějme jazyky L, M. Definujeme následující operace:

- zřetězení $L.M = \{uv|u \in L \wedge v \in M\}$,
- mocninu $L^0 = \{\lambda\}, L^{i+1} = L^i.L$,
- pozitivní iteraci $L^+ = \bigcup_{i \geq 1} L^i$,
- obecnou iteraci $L^* = \bigcup_i L^i$,
- otočení (zrcadlový obraz, reverze) $L^R = \left\{u^R|u \in L\right\}$, $(x_1 x_2 \ldots x_n)^R = x_n \ldots x_2 x_1$,
- levý kvocient $M \setminus L = \{v|uv \in L \wedge u \in M\}$,
- levá derivace $\partial_w L = \{w\} \setminus L$,
- pravý kvocient $L/M = \{u|uv \in L \wedge v \in M\}$,

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

Věta 4.2 (Uzavřenost regulárních jazyků na řetězcové operace)

Regulární jazyky jsou uzavřené na 10 operací výše.

Definice 4.3 (Regulární jazyky)

Algebraický popis jazyků. Definuje pouze regulární jazyky, ale všechny.

Základ λ = prázdný řetězec (λ), $\emptyset$ = prázdný výraz ($\{\}$), písmeno abecedy ($\{a\}, a \in \Sigma$).

Zbytek vyrobíme indukcí pomocí: $\alpha + \beta$ ($L(\alpha) \cup L(\beta)$), $\alpha\beta$ ($L(\alpha)L(\beta)$), α^* ($L(\alpha)^*$), (α) ($L(\alpha)$) ($= \alpha$).

Definice 4.4 (Priorita)

Největší prioritu má $*$, potom zřetězení a nakonec sjednocení.

Věta 4.3 (varianta Kleene)

Každý jazyk reprezentovaný konečným automatem lze zapsat jako regulární výraz. A opačně.

Důkaz

$\Leftarrow$: triviální indukcí dle struktury regulárního výrazu.

$\Longrightarrow$: Zkonstruujeme induktivně (podle k) R_{ij}^k, kde k značí maximální číslo mezistavu na cestě, i je počáteční stav, j je koncový stav. R_{ij}^k tedy určuje regulární výraz všech slov, kterými se dostanu přes mezistavy $\leq k$ z i do j. Pro $k = 0$ je konstrukce zřejmá (součet všech písmen vedoucích z i do j, resp. $\emptyset$).

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

Nakonec vezmu regulární výraz, který je součtem všech R z počátečního stavu do nějakého koncového s $k = n$ (n = počet stavů). □

Definice 4.5 (Substituce jazyků)

Mějme konečnou abecedu Σ. Pro každé $x \in \Sigma$ budiž $\sigma(x)$ jazyk v nějaké abecedě Y_x. Dále položme $\sigma(\lambda) = \{\lambda\}$ a $\sigma(u.v) = \sigma(u).\sigma(v)$.

Zobrazení $\sigma : \Sigma^* \to P(Y^*)$, kde $Y = \sum_{x \in \Sigma} Y_x$ se nazývá substituce. Nevypouštějící substituce je substituce, kde žádné $\sigma(x)$ neobsahuje λ.

Definice 4.6 (Homomorfizmus)

Homomorfizmus 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 $\forall x : w_x \neq \lambda$, jde o nevypouštějící homomorfizmus.

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

Věta 4.4 (Uzavřenost na homomorfizmus)

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

Důkaz
Prezentace. (Indukcí rozebereme sjednocení, zřetězení a iteraci na základní symboly a ty proženeme σ). Nezkouší se. □

Věta 4.5

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

Důkaz
Pro L máme DFA $A = (Q, \Sigma, \delta, q_0, F)$. Definujeme λ-NFA $B = Q', T, \delta', [q_0, \lambda], F \times \{\lambda\}$, kde $Q' = \{[q, u]\}, q \in Q, u \in \Sigma^*, \exists (a \in T)\ \exists (v \in \Sigma^* : h(a) = vu)$. $\delta'([q, \lambda], a) = [q, h(a)]$ a $\delta'([q, bv], \lambda) = [p, v]$, kde $\delta(q, b) = p$ a $b \in \Sigma$. □

5 Dvousměrné (dvoucestné) konečné automaty

Definice 5.1 (Dvousměrné (dvoucestné) konečné automaty (2DFA))

Dvousměrným (dvoucestným) konečným automatem nazýváme pětici $A = (Q, \Sigma, \delta, q_0, F)$, kde Q, Σ, q_0, F jsou jako obvykle a δ je zobrazení $Q \times \Sigma \to Q \times \{-1, 1\}$ určuje přechodovou funkci rozšířenou o pohyb hlavy.

Poznámka
Někdy se uvažuje, že hlava se nemusí posunout. Tedy δ bude do $Q \times \{-1, 0, 1\}$.

Takhle je deterministický, nedeterministický nebudeme zavádět.

Definice 5.2 (Výpočet dvousměrného automatu)

Slovo w je přijato dvousměrným konečným automatem, pokud výpočet začal na prvním písmenu slova w vlevo v počátečním stavu, čtecí hlava poprvé opustila slovo w vpravo v některém přijímajícím stavu.

Poznámka
Můžeme si na kraj přidat speciální koncové znaky $\# \notin \Sigma$, abychom mohli lépe konstruovat automat. Pomocí $\delta\#$ a $\delta^R\#$ jsme schopni $\#$ odstranit (tedy přidání $\#$ nám nemění regularitu).

Věta 5.1

Jazyky přijímané dvousměrnými konečnými automaty jsou právě regulární jazyky.

Důkaz
$\Leftarrow$: Triviální. Hlavou pohybuji jen doprava.

TODO □

Definice 5.3 (Palindrom)

Palindrom je řetězec $w = w^R$.

Lemma 5.2

Jazyk L_{pal} všech palindromů není regulární.

Důkaz
Sporem. Předpokládejme, že je regulární a n je konstanta z pumping lemmatu. Uvažujme slovo $w = 0^n 1 0^n$. $w = xyz$, y obsahuje jednu nebo více z prvních n nul a neobsahuje jedničku. Zapumpováním přidáme na začátek 0, tedy to už nebude palindrom. □

Definice 5.4 (Formální (generativní) gramatika)

Formální (generativní) gramatika je $G = (V, T, P, S)$ složené z konečné množiny neterminálů (variables) V, neprázdné konečné množiny terminálních symbolů (terminálů) T, počátečního symbolu $S \in V$ a konečné množiny pravidel (produkcí) P reprezentující rekurzivní definici jazyka. Každé pravidlo má tvar:

$$\beta A \gamma \to \omega, A \in V, \beta, \gamma, \omega \in (V \cup T)^*.$$

Jazyky jsou typu $\mathcal{L}_0$

Definice 5.5 (Bezkontextová gramatika)

Gramatika, kde pravidla mají tvar $A \to \omega, A \in V, \omega \in (V \cup T)^*$.

Jazyky jsou typu $\mathcal{L}_1$

Definice 5.6 (Kontextová gramatika)

Gramatika, kde pravidla mají tvar $\gamma A\beta \to \gamma\omega\beta, A \in V, \gamma, \beta \in (V \cup T)^*, \omega \in (V \cup T)^+$ (tzv. nezkracující). Jedinou výjimkou je pravidlo $S \to \lambda$, potom se ale S nevyskytuje na pravé straně žádného pravidla (prostě přidáme nulové slovo, aniž bychom něco rozbili).

Jazyky jsou typu $\mathcal{L}_1$

Definice 5.7 (Regulární / pravé lineární gramatiky)

Gramatiky, kde pravidla jsou 2 typů: $A \to \omega B$ a $A \to \omega$, $A, B \in V, \omega \in T^*$.

Jazyky jsou typu $\mathcal{L}_3$

Pozorování
$\mathcal{L}_0 \supset \mathcal{L}_1 \supset \mathcal{L}_2 \supset \mathcal{L}_3$.

Důkaz
Neostré inkluze z definice.

Ostré později. □

Poznámka (Notace)
Terminály = malá písmena, číslice, znaky. Neterminály velká písmena. Řetězce terminálů = malá písmena z konce abecedy. Terminál nebo neterminál = velká písmena z konce abecedy. Řetězec neterminálů a terminálů = řecká písmena. Svislítko (OR) je kompaktní zápis více pravidel.

Definice 5.8 (Derivace $\implies ^*$)

Mějme gramatiku G. Říkáme, že α se přímo přepíše na ω (píšeme $\alpha \implies _G\omega$ nebo $\alpha \implies \omega$), jestliže ω vznikne z α 'aplikováním' jednoho pravidla.

Říkáme, že α se přepíše na ω (píšeme $\alpha \implies {}^*_G\omega$ nebo $\alpha \implies {}^*\omega$), jestliže ω vznikne z α 'aplikováním' konečně mnoha pravidel. Posloupnost β_i, že $\alpha = \beta_1 \implies \beta_2 \implies \ldots \implies \beta_n = \omega$ nazýváme derivací (odvozením). Pokud $\forall i \neq j : \beta_i \neq \beta_j$, pak hovoříme o minimálním odvození.

Definice 5.9 (Jazyk generovaný gramatikou)

$(L(G))$, tj. jazyk generovaný gramatikou G je množina terminálních řetězců, pro které existuje derivace ze startovního symbolu.

Jazyk neterminálu $A \in V$ je $L(A) = \{w \in T^* | A \implies {}^*w\}$.

Pozorování (Gramatika typu 3)
Každé slovo derivace obsahuje právě jeden neterminál, který je zcela vpravo. Druhým typem pravidla se derivace uzavírá, krok derivace pouze generuje symboly a změní neterminál.

Věta 5.3

Pro každý jazyk rozpoznávaný konečným automatem existuje gramatika typu 3, která ho generuje.

Důkaz
$L = L(A)$ pro DFA $A(Q, \Sigma, \delta, q_0, F)$. Definujeme gramatiku $G = (Q, \Sigma, P, q_0)$, kde P mají tvar $p \to aq$, když $\delta(p, a) = q$, $p \to \lambda$, když $p \in F$. Takováto gramatika zřejmě generuje L.

$$a_1 \dots a_n \in L(A) \Leftrightarrow \exists q_0, \dots, q_n \in Q : \delta(q_i, a_{i+1}) = q_{i+1}, q_n \in F \Leftrightarrow$$

$\Leftrightarrow (q_0 \implies a_1q_1 \implies \dots \implies a_1 \dots a_nq_n \implies a_1 \dots a_n)$ je derivace $\Leftrightarrow a_1 \dots a_n \in L(G)$.

□

Lemma 5.4

Ke každé gramatice typu 3 existuje gramatika typu 3, která generuje stejný jazyk a obsahuje pouze pravidla tvaru $A \to aB, A \to \lambda, A, B \in V, a \in T$.

Důkaz
Zkonstruujeme tak, že zavedeme dostatečný počet nových neterminálů a pravidlo $A \to a_1 \dots a_n[B]$ přepíšeme na $A \to a_1Y_1 \to a_1a_2Y_2 \to \dots \to a_1 \dots a_n[B]$. Smažeme i pravidla typu $A \to B$. □

Věta 5.5

Pro každý jazyk generovaný gramatikou typu 3 existuje konečný automat, který ho rozpoznává.

Důkaz
Najdeme λ-NFA podobně jako jsme hledali gramatiku v důkazu předchozí věty z gramatiky z předchozího lemmatu. □

Definice 5.10 (Levé lineární gramatiky)

Gramatika G je levá lineární, jestliže má pouze pravidla tvaru $A \to Bw, A \to w, A, B \in V, w \in T^*$.

Lemma 5.6

Jazyky generované levou lineární gramatikou jsou právě regulární jazyky.

Důkaz
'Otočíme pravidla' a získáme pravou lineární gramatiku, k té najdeme automat. □

Definice 5.11 (Lineární gramatika, jazyk)

Gramatika je lineární, jestliže má pouze pravidla tvaru $A \to uBw, A \to w$. Lineární jazyky jsou právě jazyky generované lineárními gramatikami.

Pozor
Platí regulární jazyky $\subset$ lineární jazyky (viz $0^n 1^n$, $S \to 0S1|01$).

Definice 5.12 (Derivační strom)

Mějme gramatiku G. Derivační strom pro G je strom, kde: kořen je označen S, každý vnitřní uzel je označen V, každý uzel je ohodnocen prvkem $V \cup T \cup \{\lambda\}$. Je-li uzel ohodnocen λ, pak je jediným synem. Pokud jsou synové označeni $X_1, X_2, \dots, X_n$ a otec A, pak $(A \to X_1 X_2 \dots X_n) \in P$.

Definice 5.13 (Slovo dané stromem)

Strom dává slovo w (yield), jestliže w je slovo složené z ohodnocení listů bráno zleva doprava.

Definice 5.14 (Levá a pravá derivace)

Levá derivace (leftmost) ($\Longrightarrow_{lm}$, $\Longrightarrow^*_{lm}$) v každém kroku přepisuje nejlevější neterminál. Analogicky pravá derivace.

Věta 5.7

Pro danou gramatiku G a $w \in T^*$ jsou následující tvrzení ekvivalentní: $A \to^* w$, $A \to^*_{lm} w$, $A \to^*_{rm} w$, existuje derivační strom s kořenem A dávající slovo w.

Důkaz
Všimneme si, že bezkontextovou gramatiku a derivační strom s kořenem A dávající slovo $w \in T^*$. Pak existuje levá derivace $A \Longrightarrow^*_{lm} w$ v G. Viz prezentace. □