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 $F = \{c_1, \ldots, c_n\}$, kde $L = \bigcup_{i \in [n]} c_i$. Přechodová funkce $\delta([u], x) = [ux]$ (korektní z definice pravé kongruence). □

Příklad

$L = \{u|u = a^+b^ic^i \wedge u = b^ic^j \wedge i, j \in \mathbb{N}_0\}$ 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 $S = \{ab, abb, abbb, \ldots, ab^{m+1}\}$. Existují dvě slova (Dirichletův princip) $i \neq j$, která padnou do stejné třídy. $ab^i \sim ab^j \Leftrightarrow ab^ic^i \sim ab^jc^i$, ale $ab^ic^i \in L \wedge ab^jc^i \notin L$. ↯. □

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 $h(q_{10}) = q_{20}$, $h(\delta_1(q, x)) = \delta_2(h(q), x)$ 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 $d*$. □

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. □

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) $F_1 \times (Q_2 - F_2)$. □

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 $w = 0^n10^n$. $w = xyz$, 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 $\Longrightarrow\ ^*$)

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

Říkáme, že α se přepíše na ω (píšeme $\alpha \overset{*}{\underset{G}{\Longrightarrow}} \omega$ nebo $\alpha \Longrightarrow {}^*\omega$), jestliže ω vznikne z α 'aplikováním' konečně mnoha pravidel. Posloupnost β_i, že $\alpha = \beta_1 \Longrightarrow \beta_2 \Longrightarrow \ldots \Longrightarrow \beta_n = \omega$ 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 $L(A) = \{w \in T^* | A \Longrightarrow {}^*w\}$.

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.

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

$$\Leftrightarrow (q_0 \implies a_1 q_1 \implies \dots \implies a_1 \dots a_n q_n \implies a_1 \dots a_n) \text{ je derivace } \Leftrightarrow a_1 \dots a_n \in L(G).$$

□

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 $A \to a_1 Y_1 \to a_1 a_2 Y_2 \to \dots \to a_1 \dots a_n[B]$. 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^n1^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 $X_1, X_2, \ldots, X_n$ a otec A, pak $(A \to X_1X_2 \ldots X_n) \in P$.

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) ($\Longrightarrow_{lm}$, $\overset{*}{\Longrightarrow}_{lm}$) 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 $A \overset{*}{\Longrightarrow}_{lm} w$ v G. Viz prezentace. □

6 Zásobníkový automat (PDA)

Definice 6.1

Zásobníkový automat (PDA) je $P = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$, kde Q je konečnámnožína stavů, Σ je neprázdná konečná množina vstupních symbolů, Γ neprázdná konečná zásobníková abeceda, δ je přechodová funkce $\delta : Q \times (\sigma \cup \{\lambda\}) \times \Gamma \to P(FIN(Q \times \Gamma^*))$, $\delta(p, a, X) \ni (q, \gamma)$, kde q je nový stav a γ je řetězec zásobníkových symbolů, který nahradí X na vrcholu zásobníku. $q_0 \in Q$ je počáteční stav, $Z_0 \in \Gamma$ je počáteční zásobníkový symbol (víc na začátku na zásobníku není), F je množina přijímajících (koncových) stavů; může být nedefinovaná.

Definice 6.2 (Přechodový diagram pro zásobníkový automat)

Uzly odpovídají stavům PDA, šipka odnikud ukazuje počátek, dvojité kruhy jsou přijímající stav, hrana odpovídá přechodu PDA.

Definice 6.3 (Situace zásobníkového automatu)

Situaci zásobníkového automatu reprezentujeme trojicí (q, w, γ), kde q je stav, w je zbývající vstup a γ je obsah zásobníku (vrchol je vlevo). Situaci značíme ID z anglického instantaneous description.

Definice 6.4 (Posloupnost situací, jazyk)

Intuitivně. Jen u jazyka můžeme mít přijímání koncovým stavem, nebo prázdným zásobníkem.

TODO

Věta 6.1

Následující tvrzení jsou ekvivalentní:

- *Jazyk L je bezkontextový, tj. generovaný CFG.*
- *Jazyk L je přijímaný nějakým zásobníkovým automatem koncovým stavem.*
- *Jazyk L je přijímaný nějakým zásobníkovým automatem prázdným zásobníkem.*

Lemma 6.2 (Od přijímajícího stavu k prázdnému zásobníku)

Mějme $L = L(P_F)$ pro nějaký PDA. Pak existuje PDA P_N takový, že $L(P_N)$.

Důkaz
Odsimulujeme P_F a ze všech přijímajících stavů dovolíme přejít do uklízecího stavu, kde automat vše vyhází. (Pozor, musíme přidat „neznámý“ prvek na zásobník, aby nedošlo k vyprázdnění v nepřijímajícím stavu.) □

Lemma 6.3 (Od prázdného zásobníku ke koncovému stavu)

Pokud $L = N(P_n)$*, pak* $L = L(P_F)$.

Důkaz
Odsimulujeme a pak se, pokud máme prázdný zásobník, dostaneme do stavu, kde přijmeme. □

Definice 6.5 (Algoritmus CYK)

TODO

Definice 6.6 (Chomského normální forma)

Bezkontextová gramatika, kde jsou pravidla pouze typu $A \to BC$ nebo $A \to A$.

Definice 6.7 (Zbytečný, užitečný, generující a dosažitelný sympbol)

Symbol X je užitečný v gramatice G, pokud existuje derivace tvaru $S \implies {}^*\alpha X\beta \implies {}^*w$. Pokud X není užitečný, říkáme, že je zbytečný.

X je generující, pokud $X \implies {}^*w$. X je dosažitelný, pokud $S \implies {}^*\alpha X\beta$.

Definice 6.8 (Eliminace zbytečných symbolů)

Nechť G je CFG a $L(G) \neq \emptyset$. Zkonstruujeme G_1 tak, že eliminujeme negenerující symboly a pravidla je obsahující a poté eliminujeme všechny nedosažitelné symboly. Pak G_1 nemá zbytečné symboly a $L(G_1) = L(G)$.

Definice 6.9 (Nulovatelný neterminál)

Neterminál je nulovatelný, pokud $A \implies {}^*\lambda$.

Definice 6.10 (Jednotkové pravidlo)

Jednotkové pravidlo je $A \to B \in P$, kde A, B jsou oba neterminály.

Definice 6.11 (Jednotkový pár)

Dvojici $A, B \in V$ takovou, že $A \implies {}^*B$ pouze jednotkovými pravidly nazýváme jednotkový pár (jednotková dvojice).

Definice 6.12 (Chomského normální tvar)

O bezkontextové gramatice $G(V, T, P, S)$ bez zbytečných symbolů, kde jsou všechna pravidla v jednom ze dvou tvarů $A \to BC$ nebo $A \to a$, $A, B, C \in V, a \in T$, říkáme, že je v Chomského normálním tvaru.

Lemma 6.4 (Gramatika v normálním tvaru, redukovaná)

Mějme bezkontextovou gramatiku G, $L(G) \setminus \{\lambda\} \neq \emptyset$. Pak existuje CFG G_1 taková, že $L(G_1) = L(G) \setminus \{\lambda\}$ a G_1 neobsahuje λ-pravidla …

Důkaz
Prostě postupně odstraníme vše, co překáží. □

Věta 6.5 (ChNF)

Mějme bezkontextovou gramatiku G, $L(G) \setminus \{\lambda\} \neq \emptyset$. Pak existuje CFG G_1 v Chomského normálním tvaru taková, že $L(G_1) = L(G) \setminus \{\lambda\}$.

Lemma 6.6 (Velikost derivačního stromu gramatiky v CNF)

Mějme derivační strom podle gramatiky $G = (V, T, P, S)$ v Chomského normálním tvaru, který dává slovo w. Je-li délka nejdelší cesty n, pak $|w| \leq 2^{n-1}$.

Důkaz
Indukcí. $|a| = 1 = 2^0$. Pokud pro $n-1$ tvrzení platí, pak se na začátku podívám, že S generuje nejvýše 2 neterminály, čímž pokračujeme do 2 stromů s cestou $n-1$, tedy slovo je délky nejvýše $2 \cdot 2^{(n-1)-1} = 2^{n-1}$. □

Věta 6.7 (Lemma o vkládání (pumping) pro bezkontextové jazyky)

Mějme bezkontextový jazyk L. Pak existuje $n \in \mathbb{N}$ takové, že každé $z \in L, |z| > n$ lze rozložit na $z = uvwxy$, kde: $|vwx| \leq n$, $vx \neq \lambda$, $\forall i \geq 0 : uv^i wx^i y \in L$.

Důkaz
Vezmeme derivační strom pro z. Najdeme nejdelší cestu a na ní dva stejné neterminály nejdále od kořene. Tyto neterminály určí dva podstromy, z nichž spodní můžeme nahradit horním. Jednoduše ověříme, že délky fungují (z minulého lemmatu). □

Věta 6.8

CFL jsou uzavřené na sjednocení, konkatenaci, iteraci, positivní iteraci, homomorfismus a zrcadlový obraz.

Důkaz
Sjednocení: pokud $V_1 \cap V_2 \neq \emptyset$, přejmenujeme neterminály. Přidáme nový startovní symbol S_n a přidáme pravidlo $S_n \to S_1|S_2$. Zřetězení stejně, akorát pravidlo bude $S_n \to S_1|S_2$.

Pozitivní a obecná iterace obdobně. (Pravidla $S_n \to SS_n|S$ a $S_n \to SS_n|\lambda$.)

Zrcadlový obraz: obrátíme pravou stranu pravidel. □

Pozor
CFL nejsou uzavřené na průnik! (Viz $\{0^n1^n2^i|n,i \geq 1\}$ a $\{0^i1^n2^n|n,i \geq 1\}$)

Věta 6.9 (CFL i DCFL jsou uzavřené na průnik s regulárním jazykem)

Mějme L bezkontextový a R regulární jazyk. Pak $L \cap R$ je bezkontextový jazyk.

Mějme L deterministický CFL a R regulární jazyk. Pak $L \cap R$ je deterministický CFL.

Důkaz
Vezmeme kartézský součin stavů a zjevným způsobem doplníme přechody. □

Věta 6.10 (CFL jsou uzavřené na substituci)

Mějme CFL jazyk L nad Σ a substituci σ na Σ takovou, že $\sigma(a)$ je CFL pro každé $a \in \Sigma$. Pak je i $\sigma(L)$ CFL (bezkontextový).

Důkaz
Vezmeme všechny gramatiky substitucí $a \in \Sigma$ a gramatiku L. Neterminály a terminály budou sjednocením neterminálů a terminálů, k pravidlům navíc přidáme lambda přechod na start příslušných automatů. □

Důsledek (Homomorfismus)
Bezkontextové jazyky jsou uzavřeny na homomorfismus.

Věta 6.11 (CFL jsou uzavřené na inverzní homomorfismus)

Mějme CFL (resp. deterministický CFL) jazyk L a homomorfismus h. Pak $h^{-1}(L)$ je bezkontextový (resp. deterministický CFL).

Důkaz
Přes buffer. □

Lemma 6.12

Bezkontextové jazyky jsou uzavřené na levý (pravý) kvocient s regulárním jazykem. (Na přednášce přeskočeno.)

Věta 6.13 (Rozdíl s regulárním jazykem)

Mějme bezkontextový jazyk L a regulární jazyk R. Pak $L - R$ je CFL.

Důkaz

$$L - R = L \cap \overline{R}.$$

□

Pozor
CFL nejsou uzavřené na doplněk.

Lemma 6.14

Doplněk deterministického CFL je opět deterministický CFL.

Důkaz
„Prohodíme koncové a nekoncové stavy.“ Nedefinované kroky ošetříme „podložkou“ na zásobníku, cyklus odhalíme pomocí čítače, až po přečtení slova procházíme koncové a nekoncové stavy. □

Pozor
DCFL nejsou uzavřené na sjednocení, viz $\{a^i b^j c^k | i \neq j \vee j \neq k \vee i \neq k\}$.

Pozor
DCFL nejsou uzavřené na homomorfismus.

TODO Dyckovy jazyky.

Definice 6.13 (Jednoznačnost a víceznačnost CFG)

Bezkontextová gramatika G je víceznačná, pokud existuje alespoň jeden řetězec, pro který můžeme najít dva různé derivační stromy.

V opačném případě je G jednoznačná.

Věta 6.15

Nechť $L = N(P)$ nebo $L(P)$ pro nějaký DPDA P. Pak L má jednoznačnou CFG.

Důkaz
Viz prezentace, nezkouší se. □

7 Turingovy stroje

Definice 7.1 (Turingův stroj)

Turingův stroj (TM) je sedmice $M = (Q, \Sigma, \Gamma, \delta, q_0, B, F)$ se složkami: Q – konečná množina stavů, Σ – konečná neprázdná množina vstupních symbolů, $\Gamma \supseteq \Sigma$ – konečná množina všech symbolů pro pásku (Chceme $Q \cap \Gamma = \emptyset$), δ – (částečná) přechodová funkce : $(Q \setminus F) \times \Gamma \to Q \times \Gamma \times \{L, R\}$, q_0 – počáteční stav, B – prázdný symbol, F – množina finálních stavů.

Definice 7.2 (Konfigurace TM (Instantaneous Description))

Konfigurace TM je řetězec $X_1X_2 \ldots X_{i-1}qX_iX_{i+1} \ldots X_n$, kde Q je stav TM, čtecí hlava je vlevo od i-tého symbolu, $X_1 \ldots X_n$ je část pásky mezi nejlevějším a nejpravějším symbolem různým od prázdného (B), s výjimkou, když hlava čte z B, pak je tam i toto B.

Definice 7.3 (Krok TM, odvození TM, přijímání jazyka TM)

Intuitivně.

Definice 7.4 (Rekurzivně spočetný jazyk)

Rekurzivně spočetný jazyk je jazyk, který je přijímán TM.

Definice 7.5 (TM se zastaví)

TM se zastaví, pokud vstoupí do stavu q, s čteným symbolem X, a není instrukce pro tuto situaci, tj. $\delta(q, X)$ není definováno.

Definice 7.6 (Rozhodování jazyka, rekurzivní jazyky)

Říkáme, že TM M rozhoduje jazyk L, pokud $L = L(M)$ a pro každé $w \in \Sigma^*$ stroj M nad w zastaví.

Jazyky rozhodnutelné TM nazýváme rekurzivní jazyky.

Definice 7.7 (Přechodový diagrem pro TM)

Přechodový diagram pro TM sestává z uzlů odpovídajícím stavům TM. Hrany $q \to p$ jsou označeny seznamem všech dvojic, kde $\delta(q, X) = (p, Y, D), D \in \{\to, \leftarrow\}$.

TODO

Věta 7.1

Každý rekurzivně spočetný jazyk je typu 0.

Důkaz
Pro Turingův stroj T najdeme gramatiku G, $L(T) = L(G)$. $G = (\{S, C, D, E\} \cup \{\underline{x}\}_{x \in \Sigma \cup \Gamma} \cup \{q_i\}_{q_i \in Q}, \Sigma, P, S)$, TODO! (Tvoříme 3 typy pravidel: vygenerování slova a jeho kopie pro výpočet, simulace automatu, úklid). □

Důkaz
Každý jazyk typu 0 je rekurzivně spočetný. □