Týden 4

Přednáška

V první části přednášky se psala první zápočtová písemka.

Ekvivalence konečných automatů; minimální automaty

Přemýšleli jsme, zda bychom uměli navrhnout (a naprogramovat) algoritmus řešící následující problém.

NÁZEV: *Evivalence konečných automatů*

VSTUP: dva konečné automaty A_1, A_2

VÝSTUP: ANO – jestliže $L(A_1) = L(A_2)$,
NE – jestliže $L(A_1) \neq L(A_2)$.

K jednomu možnému algoritmu nás přímo přivedla myšlenka, že u negativního případu, tedy $L(A_1) \neq L(A_2)$, by bylo dobré také ukázat protipříklad, tedy (co nejkratší) slovo w, které patří jen do jednoho z jazyků $L(A_1), L(A_2)$. Uvědomili jsme si, že takové slovo patří do jazyka

$$L = (L(A_1) - L(A_2)) \cup (L(A_2) - L(A_1)) = (L(A_1) \cap \overline{L(A_2)}) \cup (L(A_2) \cap \overline{L(A_1)})$$

a že díky dříve probraným algoritmům snadno sestrojíme A tak, že $L(A) = L$. Je nám tedy jasný algoritmus, který k zadaným A_1, A_2 sestrojí A tak, že

$$L(A_1) = L(A_2)) \;\Leftrightarrow\; L(A) = \emptyset\,.$$

No a napsat proceduru (algoritmus), která o zadaném konečném automatu A zjistí, zda $L(A) = \emptyset$, hravě zvládneme (když si vzpomeneme na algoritmus pro zjišťování dosažitelných stavů; stačí prostě zjistit, zda nějaký přijímající stav je dosažitelný [z počátečního]).

Ekvivalence konečných automatů se ovšem dá algoritmicky (rychle) zjišťovat i jinak. Nejprve jsme si uvedli následující přirozenou definici.

> Konečný automat A je *minimální*, jestliže neexistuje automat A', který je ekvivalentní s A (pro nějž je tedy $L(A) = L(A')$) a který má méně stavů než A.

Pak jsme si uvedli tyto věty.

Věta. Je-li automat redukovaný a nemá nedosažitelné stavy, pak je minimální.

(Poznámka. Při přednášce jsme do pojmu *redukovaný* už zahrnovali i nepřítomnost nedosažitelných stavů.)

Věta. Dva minimální automaty, které přijímají tentýž jazyk (a jsou tedy ekvivalentní), jsou izomorfní, tedy stejné až na pojmenování stavů; to také znamená, že mají stejný normovaný tvar.

Platnost vět jsme si naznačili jen intutivně, k důkazu jsme se vrátili v poslední části přednášky. Uvědomili jsme si ale, že algoritmus rozhodující ekvivalenci konečných automatů může být tedy založen na redukci a převodu do normovaného tvaru. (Dva automaty jsou ekvivalentní právě tehdy, když po provedené redukci mají stejný normovaný tvar.)

Bezkontextové gramatiky

Připomněli jsme si, že

$$(((a \cdot a) + b)^*)$$

je příklad (úplně uzávorkovaného) regulárního výrazu, který reprezentuje jazyk v abecedě $\{a, b\}$. Uvědomili jsme si, že ovšem samotný regulární výraz je prostě řetězcem symbolů abecedy

$$\Sigma = \{\emptyset, \varepsilon, a, b, +, \cdot, ^*, (,)\}\,.$$

Ne každý řetězec v Σ^* je ale regulárním výrazem. (Např. řetězec $a)++($ regulárním výrazem není.)
Snadno jsme vyvodili, že množina těchto regulárních výrazů, označovaná $RV(\{a, b\})$, není regulárním jazykem. Dá se ale generovat bezkontextovou gramatikou, např.

$$R \longrightarrow \emptyset \mid \varepsilon \mid a \mid b \mid (R + R) \mid (R \cdot R) \mid (R^*)\,.$$

Jiná gramatika generující uvedený jazyk je

$$\begin{aligned}
R &\longrightarrow C \mid L \mid (RBR) \mid (RU) \\
C &\longrightarrow \emptyset \mid \varepsilon \\
L &\longrightarrow a \mid b \\
B &\longrightarrow + \mid \cdot \\
U &\longrightarrow {}^*
\end{aligned}$$

Demonstrovali jsme si základní pojmy teorie bezkontextových gramatik. Ukázali jsme si *(levou) derivaci* slova $(((a \cdot a) + b)^*)$, příslušný *derivační strom*, apod.
Připomněli jsme definici *bezkontextové gramatiky* jako struktury

$$G = (\Pi, \Sigma, S, P)$$

a *jazyka generovaného gramatikou*

$$L(G) = \{w \in \Sigma^* \mid S \Rightarrow^* w\}\,.$$

Pak jsme se vrátili k příkladu jazyka $RV(\{a, b\})$ a upravili jej tak, že v řetězcích (regulárních výrazech) vynecháváme tečku pro zřetězení a nemusíme plně uzávorkovávat. Např. výraz $(((a \cdot a) + b)^*)$ můžeme zapsat $(aa + b)^*$. Takto upravený jazyk generuje např. gramatika

$$R \longrightarrow \emptyset \mid \varepsilon \mid a \mid b \mid R+R \mid R \cdot R \mid R^* \mid (R)\,.$$

Všimli jsme si ovšem, že např. ono slovo $aa + b$ má v této gramatice dva různé derivační stromy; tedy tato gramatika *není jednoznačná*. (Příčinou je tady fakt, že naše dohodnutá priorita operátorů není v gramatice reflektována.)

Demonstrovali jsme, že v tomto případě lze nalézt ekvivalentní gramatiku (tedy gramatiku generující tentýž jazyk), která jednoznačná je. Úvahami nad strukturou regulárních výrazů jsme postupně došli ke gramatice

$$\begin{aligned} R &\longrightarrow T+R \mid T \\ T &\longrightarrow FT \mid F \\ F &\longrightarrow F^* \mid (R) \mid C \\ C &\longrightarrow \emptyset \mid \varepsilon \mid a \mid b \end{aligned}$$

Pomohla nám úvaha, že T (Term) reprezentuje ty regulární výrazy, které nejsou ve tvaru $R_1 + R_2$ (pro dva regulární výrazy R_1, R_2), a F (Factor) reprezentuje ty výrazy, které nejsou ve tvaru $R_1 + R_2$ ani R_1R_2.

Na závěr jsme promyšlením tvarů slov celkem přirozeně zkonstruovali gramatiky pro jazyk palindromů $\{w \in \{a,b\}^* \mid w = w^R\}$

$$S \longrightarrow \varepsilon \mid a \mid b \mid aSa \mid bSb$$

a pro jazyk posloupností v abecedě $\{\,(,),[,]\,\}$, které odpovídají správnému uzávorkování

$$S \longrightarrow \varepsilon \mid SS \mid (S) \mid [S]\,.$$

Sekce pro hlubší zájemce – důkazy

Ukázali jsme si, na čem lze založit důkaz následujících dvou vět.

Věta. Je-li automat redukovaný a nemá nedosažitelné stavy, pak je minimální.

Věta. Dva minimální automaty, které přijímají tentýž jazyk (a jsou tedy ekvivalentní), jsou izomorfní, tedy stejné až na pojmenování stavů.

Uvažujme dva konečné automaty $A_1 = (Q_1, \Sigma, \delta_1, q_{01}, F_1\}$, $A_2 = (Q_2, \Sigma, \delta_2, q_{02}, F_2\}$ takové, že $L(A_1) = L(A_2) = L$; tedy $L^{toAcc}_{q_{01}} = L^{toAcc}_{q_{02}}$.

Připomeňme, že pro každé $w \in \Sigma^*$ existuje právě jeden stav $q_1 \in Q_1$ a právě jeden stav $q_2 \in Q_2$ tak, že $q_{01} \xrightarrow{w} q_1$, $q_{02} \xrightarrow{w} q_2$. (Přesněji psáno $q_{01} \xrightarrow{w}_{A_1} q_1$, $q_{02} \xrightarrow{w}_{A_2} q_2$.) Zřejmě přitom platí $w\backslash L = L^{toAcc}_{q_1} = L^{toAcc}_{q_2}$.

Ke každému dosažitelnému stavu $q_1 \in Q_1$ (v automatu A_1) tedy existuje dosažitelný stav $q_2 \in Q_2$ (v automatu A_2) tak, že $L^{toAcc}_{q_1} = L^{toAcc}_{q_2}$. Když je tedy A_1 bez nedosažitelných stavů a je redukovaný, což znamená, že $L^{toAcc}_{q_1} \neq L^{toAcc}_{q_1'}$ pro každé $q_1 \neq q_1'$ v Q_1, musí mít A_2 alespoň tolik stavů jako A_1 (tedy $|Q_1| \leq |Q_2|$). Platnost první věty je tedy jasná.

Když A_1, A_2 jsou minimální (a tedy i redukované a bez nedosažitelných stavů), tak zobrazení $f : Q_1 \to Q_2$ definované tak, že $L^{toAcc}_{q_1} = L^{toAcc}_{f(q_1)}$, je příslušným izomorfismem. (Ověřte detailně sami.)

Partie textu k prostudování

Část 3.7. (ekvivalence konečných automatů, minimální automaty), části 4.1., 4.2., 4.3. (bezkontextové gramatiky, jednoznačné gramatiky).

(Máte si udělat přinejmenším dobrou první představu a zamyslet se nad příklady, speciálně těmi plánovanými na cvičení, ať se můžete na cvičení aktivně účastnit a případné problémy si tam objasnit.)

Cvičení

Na začátku diskuse opravené první zápočtové písemky.

Prezentace referátů

Referát č. 5

K regulárnímu jazyku $L \subseteq \Sigma^*$ nazveme *kanonickým* takový KA A, $L(A) = L$, který je v normovaném tvaru (a tudíž jsou všechny stavy dosažitelné) a v němž pro každé $w \in \Sigma^*$ existuje právě jeden stav q takový, že $L_q^{toAcc} = w\backslash L$.

Vysvětlete, proč ke každému regulárnímu jazyku L existuje právě jeden kanonický automat, a proč je kanonický automat minimálním.

Referát č. 6

Vysvětlete, proč pro každé n existuje nedeterministický automat A_n s n stavy takový, že minimální deterministický konečný automat přijímající $L(A_n)$ má 2^n stavů. (Ilustrujte např. na konkrétním příkladu pro $n = 5$. Ten můžete najít např. ve starším materiálu `http://www.cs.vsb.cz/jancar/TEORET-INF/teoret-inf.pdf`.)

Příklady

Příklad 4.1

Připomeňme problém

NÁZEV: *Evivalence konečných automatů*

VSTUP: dva konečné automaty A_1, A_2

VÝSTUP: ANO – jestliže $L(A_1) = L(A_2)$,
NE – jestliže $L(A_1) \neq L(A_2)$.

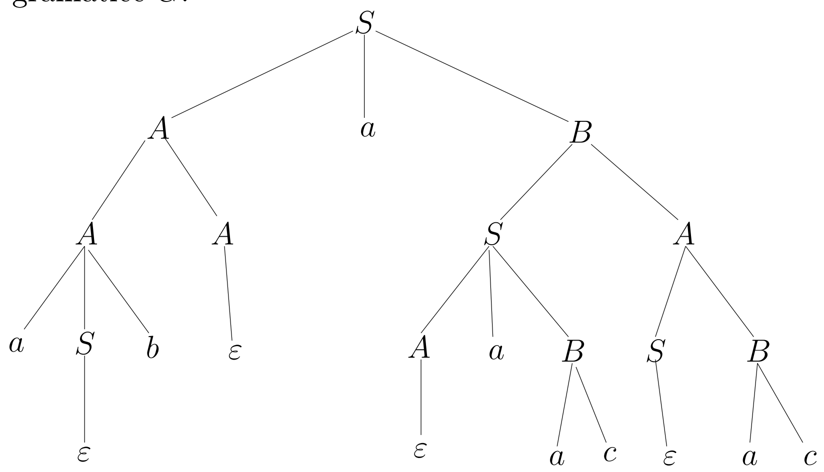
a uvažujme následující algoritmus pro jeho řešení:

> generuj postupně (všechna) slova $w_0, w_1, w_2, \ldots$ v abecedě daných automatů; pro každé w_i přitom zjisti, zda jej oba automaty přijímají či oba nepřijímají – není-li tomu tak, skonči s výsledkem $L(A_1) \neq L(A_2)$.

Je zřejmé, že běh tohoto algoritmu neskončí v případě $L(A_1) = L(A_2)$. Lze tento nedostatek opravit tím, že necháme algoritmus probírat jen slova do délky n, kde n je součtem počtu stavů A_1 a A_2, a nenajde-li se rozlišující slovo do té doby, pak algoritmus zahlásí $L(A_1) = L(A_2)$? Pokud je tomu tak (tedy odpověď algoritmu je vždy správná), v čem je tento algoritmus zřejmě horší než algoritmy probrané na přednášce?

Příklad 4.2

Na obrázku je derivační strom pro slovo $w = abaaacac$ odpovídající jisté bezkontextové gramatice G.



- Vypište všechna pravidla G, jejichž existenci můžete vyvodit z daného derivačního stromu.
- Napište levé odvození (levou derivaci) slova w podle gramatiky G.
- Najděte *menší* derivační strom pro slovo $abaaacac$ a zakreslete jej tak, že všechny listy budou na stejné úrovni (tedy odvozené slovo bude celé na „jednom řádku“).
- Najděte nejlevější větev (v onom menším stromě), která obsahuje dva výskyty neterminálu B. Využijte to k důkazu, že gramatika generuje také slovo $abaac$. Pak ukažte, že gramatika také generuje slova $aba(a)ac(ac)$, $aba(a)^2ac(ac)^2$, $aba(a)^3ac(ac)^3$, ...
- Lze z dostupné informace zjistit něco ohledně jednoznačnosti gramatiky G ?

Příklad 4.3

Uvažujme jazyk $L = \{ w \in \{a, b\}^* \mid |w| \geq 1 \text{ a } |w|_a = |w|_b \}$.
Charakterizujte slova z $L^2 = L \cdot L$. Je pravda, že $L = L^2$? Platí případně alespoň jedna z inkluzí $L \subseteq L^2$, $L^2 \subseteq L$?
Charakterizujte slova z $L - L^2$.
Na základě předešlých úvah navrhněte bezkontextovou gramatiku generující L.

Příklad 4.4

Snažte se co nejvýstižněji charakterizovat jazyk generovaný gramatikou
$S \longrightarrow bSS \mid a$