Poznámky k přednáškám TID

Petr Zemek

Fakulta informačních technologií VUT v Brně, Česká Republika,
email: izemek@fit.vutbr.cz

9. ledna 2011

Abstrakt V tomto textu přináším některé své poznámky k přednáškám předmětu TID v roce 2010, které vycházejí z materiálů z roku 2007. Tyto materiály byly zpracovány Tomášem Masopustem, Alexanderem Medunou a Jirkou Techetem v rámci FRVŠ projektu. Poznámky vychází jak z výkladu prof. Meduny, tak z mého vlastního výzkumu.

Poznámky k jednotlivým přednáškám

Přednášky, ke kterým se tento text vztahuje, lze nalézt v [1]. Občas neuvádím reference, odkud danou informaci mám, takže v takovým případech mi buď budete muset věřit, nebo si to zjistit sami :). Čísla v následujících sekcích jsou čísla slajdů, uvedená v pravém dolním rohu každé přednášky. Pokud s některou poznámkou nesouhlasíte, dejte mi, prosím, vědět a pokusíme se to vyřešit. Díky.

1 Chomsky Hierachy

7 Zde definovaná gramatika bývá často označována jako *monotonous/non-erasing grammar*. Kontextová gramatika má obvykle pravidla tvaru $\alpha A\beta \to \alpha y\beta$, kde α a β jsou libovolné řetězce, A je neterminál, a y je libovolný neprázdný řetězec.

7 Byly zavedeny také *k-linear grammars* (viz např. stránka 72 v [6]), které, kromě jediného pravidla tvaru $S \to A_1A_2 \cdots A_k$, kde A_i jsou neterminály, obsahují pouze lineární pravidla (S se pak nesmí vyskytovat na pravé straně žádného pravidla). Ty, co jsou pak uvedeny na slajdech, jsou vlastně 1-*linear grammars*.

2 Context-Free Grammars

8 Uvedené pumping lemma je pouze nutnou podmínkou pro to, aby jazyk byl bezkontextový. Jinými slovy, existují jazyky, které toto lemma splňují, ale nejsou bezkontextové. Není známo, zda existuje nějaká nutná a zároveň postačující podmínka pro to, aby jazyk byl bezkontextový (takové existují pro regulární jazyky, viz např. sekce 4.1 v kapitole 2 v [3]).

3 Normal Forms

3 Pokud $n \leq 1$, dostaneme regulární gramatiku. Pokud $n = 0$, tak si všimněte, že generovaný jazyk může být maximálně T.

7 Všimněte si, že u

$$CA_3 \cdots A_m \to B_2 \cdots B_n \text{ to } R$$

bude vždy platit $A_3 \cdots A_m = \varepsilon$ a $|B_2 \cdots B_n| = 2$.

7 V určitých případech bude nutné po tomto kroku aplikovat krok 4.

10 Všimněte si, že oproti Kurodově normální formě gramatika v (první) Geffertově normální formě obsahuje vždy pouze **jediné** kontextové pravidlo, nikoliv několik pravidel tohoto tvaru. Taktéž obsahuje **pouze** 4 neterminály. Tyto skutečnosti činí tuto normální formu opravdu zajímavou. Obdobně u dalších dvou forem.

4 Matrix Grammars

15 M je ve výsledku konečná množina posloupností, kde každá posloupnost (= řetězec) je tvaru $(r_1, s_1)(r_2, s_2) \cdots (r_n, s_n)$, kde každé r_i je pravidlo a s_i je buď $+$ nebo $-$. Samozřejmě, každá posloupnost může být různě dlouhá.

18 Tento jazyk nelze generovat žádnou maticovou gramatikou **bez appearance checking** (vyplývá to z výsledků v [5]).

5 Random Context Grammars

Jen něco k terminologii. *Permitting grammar = random context grammar. Forbidding grammar* je taková *random context grammar with appearance checking*, kde je splněna podmínka ze slajdu 5. Tudíž, každá permitting grammar i forbidding grammar je vlastně speciálním případem random context grammar with appearace checking.

6 Programmed Grammars

2 Pokud do definice přidám počáteční pravidlo či koncová pravidla, tak nijak nezměním sílu programovaných gramatik.

2 Pokud budu požadovat, aby R byla funkce, tak takovou programovanou gramatikou nelze vygenerovat žádný nekonečný jazyk.

7 Regulated Rewriting Hierarchy

3 $\mathscr{L}(RC) = \mathscr{L}(Per)$ vyplývá přímo z definice (viz výše).

3 V [9] bylo dokázáno, že $\mathscr{L}(RC, \varepsilon) = \mathscr{L}(RC)$.

5 Tomuto homomorfismu se také říká *coding* (zobrazuje symboly pouze na jiné symboly).

8 L-Systems

2 Pěkná a velmi krátká definice $0L$ systému je následující [2]. $0L$ *systém* je trojice $G = (T, h, w)$, kde T je abeceda, h je konečná substituce (viz [8]) z T do konečných podmnožin T^*, a $w \in T^+$. Jazyk je pak definovaný jako $L(G) = \bigcup_{i \geq 0} h^i(w)$. Pokud je h tzv. ε-free substituce, pak se jedná o $P0L$ *systém*. Pokud je h homomorfismus, pak se jedná o $D0L$ *systém*.

13 Náznak důkazu neuzavřenosti vůči zadaným operacím:
- homomorfismus: Nechť φ je homomorfismus nad $\{a\}$ definovaný jako $\varphi(a) = a^5$. Jazyk $K = \{\varepsilon, a, aa\}$ lze generovat $0L$ systémem, ale $\varphi(K) = \{\varepsilon, a^5, a^{10}\}$ již ne;
- průnik: $\{a\}$ je $0L$ jazyk, $\{b\}$ je $0L$ jazyk, ale $\{a\} \cap \{b\} = \emptyset$ není $0L$ jazyk;
- konkatenace: $\{a\}$ je $0L$ jazyk, $\{\varepsilon, aa\}$ je $0L$ jazyk, ale $\{a\}\{\varepsilon, aa\} = \{a, aaa\}$ není $0L$ jazyk;
- doplněk: $\{a\}^*$ je $0L$ jazyk, ale $\overline{\{a\}^*} = \emptyset$ není $0L$ jazyk (předpokládejme, že se bavíme o jazyku nad abecedou $\{a\}$);

13 Uzavřenost vůči reverzi lze ukázat tak, že se obrátí počáteční řetězec (axiom) a všechny pravé strany pravidel.

14 Důkaz první věty je uveden v [6] (strana 239, Věta 13.2). Důkaz zbylých dvou vět je celkem jednoduchý (kdyby měl někdo zájem, tak mi napište).

15 $\{b^n c d^n \mid n \geq 1\}$ lze generovat pomocí $0L$ systému obsahující pravidla $c \to bcd$, $b \to b$, $d \to d$, kde axiom je c.

23 $\mathscr{L}(ET0L) \subseteq \mathscr{L}(CS)$ vyplývá z toho, že $\mathscr{L}(ET0L) = \mathscr{L}(EPT0L)$ (vymazávací pravidla nezvyšují sílu ET0L systémů, viz [2]) a pak z Workspace theorem (věta III.10.1 v [6]). Striktnost této inkluze pak vyplývá z bodu (2) ve slajdech.

9 CD Grammar Systems

9 Co je zde zajímavé, tak je první a poslední věta. První říká, že i přesto, že můžeme mít libovolný počet komponent, tak za určitých derivačních módů sílu vůbec nezvýšíme a zůstaneme na bezkontextových jazycích. U té poslední je to pak skutečnost, že pro dvě komponenty jsme pořád na úrovni bezkontextových jazyků, ale pro tři komponenty už jsme na úrovni ET0L jazyků.

10 PC Grammar Systems

2 Symbolům z K se říká *komunikační symboly*.

3 Oněm n-ticím se říká *konfigurace*.

4 Může nastat jakýsi „deadlock“, a to tehdy, pokud všechny komponenty obsahují komunikační symboly a chtějí provést vkopírování.

7 Těmto gramatickým systémům se říká *s návratem* a *bez návratu*.

12 Co je zde zajímavé, tak je to, že v žádném z těchto případů se nedostaneme na úroveň Turingových strojů.

11 Scattered Context Grammars (SCG)

8 Druhá věta říká, že k vygenerování libovolného rekurzivně spočetného jazyka nám stačí SCG se třemi neterminály. Třetí věta říká, že každý rekurzivně spočetný jazyk lze generovat pomocí SCG, která obsahuje pouze dvě „kontextová pravidla“ (= pravidla, jejichž délka je dvě a více).
9 První věta říká, že každý rekurzivně spočetný jazyk lze generovat pomocí SCG, jejíž pravidla jsou nejvýše délky dva (čili $n \leq 2$ u všech pravidel).
10 Místo $|M| = |V| + 5$ má být $|V| = |M| + 5$.
12 Zde je dobré si uvědomit, že se nepožaduje, aby ona SCG G byla propagating (jinak by $|x_1 \cdots x_n| \geq n$ bylo triviálně splněno).
13 Místo „permuation“ má být „permutation“.
14 Druhý otevřený problém byl vyřešen v [4]. Tudíž, druhá věta ze slajdu 8 platí i pro $|V - T| = 2$.

12 Multi-Grammars

2 K může obecně obsahovat nekonečné množiny (což je možno vidět jako částečný spor s častým požadavkem, aby všechny komponenty formalismů byly konečné).
10 O aktivaci se zde již nic nepíše, protože v každém kroku jsou aktivovány všechny symboly.
12 Přísně matematicky by K měla být jednoprvková množina. V takových případech se ale často zápis zjednodušuje tak, že se složené závorky vynechají.

13 Other Grammars

2 Přidáním koncových stavů se síla stavových gramatik nezmění (viz [7]).
5 Pokud se dostanu do koncového stavu, tak končím (z něho již nelze provést žádný derivační krok). Z toho důvodu je zbytečné, aby koncových stavů bylo více (viz Filipova poznámka na přednášce).

14 Turing Machines

9 Otevřený problém „Jsou deterministické lineárně omezené automaty stejně silné jako nedeterministické?“ lze, s využitím teorie složitosti, přeformulovat do následující podoby: Platí $\mathrm{NSPACE}(\mathcal{O}(n)) = \mathrm{DSPACE}(\mathcal{O}(n))$?

15 Transducers

2 Transducer se obvykle překládá jako *převodník*.
2 Někdy (např. v [3]) se konečný převodník definuje tak, že $a \in I^*$.
16 I a O nemusí být disjunktní.

Reference

[1] T. Masopust a A. Meduna a J. Techet. TID: Materials for lectures [online]. Poslední aktualizace 2011-01-07. [cit. 2011-01-07]. Dostupné na URL: <http://www.fit.vutbr.cz/~meduna/work/doku.php?id=lectures:phd:tid:tid#materials_for_lectures_frvstechet_masopust_meduna_2007>.

[2] G. Rozenberg a A. Salomaa. *Mathematical Theory of L Systems*. Academic Press, 1980.

[3] G. Rozenberg a A. Salomaa, editor. *Handbook of Formal Languages, Vol. 1: Word, Language, Grammar*. Springer, 1997.

[4] E. Csuhaj-Varjú a G. Vaszil. Scattered context grammars generate any recursively enumerable language with two nonterminals. *Information Processing Letters*, 110:902–907, 2010.

[5] D. Hauschildt a M. Jantzen. Petri net algorithms in the theory of matrix grammars. *Acta Informatica*, 31(9):719–728, 1994.

[6] A. Salomaa. *Formal Languages*. Academic Press, 1973.

[7] P. Zemek. On state grammars with final states [online]. Poslední aktualizace 2010-11-04. [cit. 2011-01-07]. Dostupné na URL: <http://www.stud.fit.vutbr.cz/~xzemek02/articles/PZ_-_On.State.Grammars.With.Final.States_-_2010.pdf>.

[8] P. Zemek. Substituce a morfismy jednoduše [online]. Poslední aktualizace 2010-07-31. [cit. 2011-01-07]. Dostupné na URL: <http://www.stud.fit.vutbr.cz/~xzemek02/articles/substituce-a-morfismy.pdf>.

[9] G. Zetzsche. On erasing productions in random context grammars. In *ICALP'10: Proceedings of the 37th International Colloquium on Automata, Languages and Programming*, pages 175–186. Springer, 2010.