Assignment 9: From Regular to Context-Free Languages

Due: 1:15pm, Tuesday, November 9

- Emails about this assignment should be directed to `cs81help@cs.hmc.edu`.
- Bring a writeup/printout to class. Illegible answers will get no credit.
- Make sure your submission includes your name!

---

1. Read pp. 109–118 (Section 2.2, up though Example 2.25) and 123–127 (Section 2.3) of Sipser. If you're interested, you can also look at 119–122 (the end of Section 2,2 proving that the language of every PDA is context free).

   Come up with (at least) two questions about the reading; these should be questions where you're not sure of the answer. These may relate to points where the book is confusing, or simply to some related question or conjecture that occurs to you while doing the reading.

2. Prove (by induction) that every string produced by the context-free grammar

   $$S \to a \mid Sa \mid aS \mid bSS \mid SSb \mid SbS$$

   contains more a's than b's. (There are several ways to structure the inductive proof; since every string produced by the grammar has at least one parse tree, one approach would be induction on the structure or size of parse trees.)

3. Prove that the following languages are context free, by giving a context-free grammar for each.

   (a) $\{ a^k a^j b^j c^k \mid j, k \geqslant 0 \}$

   (b) $\{ a^i b^{i+k} c^k \mid i, k \geqslant 0 \}$

   (c) $\{ a^i b^j \mid i \neq j \wedge i, j > 0 \}$

4. Prove that the following languages are *not* context free:

   (a) $\{ a^n b^{2n} c^n \mid n \geqslant 0 \}$

   (b) $\{ a^{\max\{m,n\}} b^m c^n \mid n, m \geqslant 0 \}$

5. Do Sipser Exercise 2.15, page 129.

6. Do Sipser Exercise 2.16, page 129.

7. Do Sipser Exercise 2.2, page 128. (Obviously, the first step is to prove that A and B are context free!)

8. **LL(1) Grammars**.

   For any $z \in (\Sigma \cup V)^*$ (i.e, z is a sequence of terminals and nonterminals) we can define the set $\textsc{First}(z) \subseteq \Sigma$ as follows:

   $$\textsc{First}(z) := \{ \sigma \in \Sigma \mid \exists w \in \Sigma^*. z \Rightarrow^* \sigma w \}$$

   That is, $\textsc{First}(z)$ contains all *terminals* that can begin a string derivable from z.

   In class, we defined an $\mathrm{LL}(1)$ grammar to be one in which predictive parsing can be made to work without guessing, making unambiguous predictions by peeking ahead only at 1 upcoming input symbol.

   It turns out that a grammar that does not mention ε is $\mathrm{LL}(1)$ if and only if the following condition holds:

   Condition 1 *For every nonterminal* A *and every pair of distinct productions* $A \to z_1$ *and* $A \to z_2$ *we have* $\textsc{First}(z_1) \cap \textsc{First}(z_2) = \emptyset$.

   (a) Give two reasons that the following grammar is not LL(1):

$$\begin{array}{lcl} S & \to & \texttt{if}\ E\ \texttt{then}\ S\ \texttt{else}\ S \\ & \mid & \texttt{begin}\ \texttt{end} \\ & \mid & \texttt{begin}\ L\ \texttt{end} \\ & \mid & \texttt{print}\ E \\ L & \to & S \\ & \mid & L\,;S \\ E & \to & \texttt{true} \\ & \mid & \texttt{false} \end{array}$$

(b) The grammar

$$\begin{array}{lcl} S & \to & E\,; \\ E & \to & T\,E' \\ E' & \to & +\,T\,E' \\ & | & -\,T\,E' \\ & | & \varepsilon \\ T & \to & 0 \mid 1 \end{array}$$

contains ε and is LL(1). Demonstrate this by building a 2-D table of predictions indexed by nonterminals in $\{S, E, E', T\}$ and terminals in $\{;, +, -, 0, 1\}$. For each combination of nonterminal V and terminal x, state which grammar rule we should use as our prediction when (1) we expect the next segment of the input to be derivable from V, and (2) we peek ahead and see that the upcoming input character is x. (E.g., if we are expecting T and see 1, we probably want to predict $T \to 1$.) In some cases (e.g., we expect T, the next character is $+$) no prediction can work regardless of what follows the first character of the input; the answer for such combinations should be "error."

(c) The grammar

$$\begin{array}{lcl} S & \to & Aa \\ A & \to & a \\ & | & \varepsilon \end{array}$$

satisfies Condition 1 but is not $\mathrm{LL}(1)$. Show that knowing only the first nonterminal in the input is not always enough to guarantee you are making the "right" prediction.

[For grammars containing ε, there is a more complicated condition that needs to be checked before we can be sure that a grammar is $\mathrm{LL}(1)$.]