# 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`.
- Grutor office hours in Platt are **Thursdays 7–9pm**, Sundays 8–10pm and Mondays 9–11pm; Prof. Stone's office hours in Olin 1251 are MW 4–5pm and TR 3–4pm and by appointment.
- The usual collaboration rules apply. You may *discuss* an exercise with any other student(s) currently taking CS 81 as long as:
  - You contribute equally;
  - You come away from this discussion only with *understanding in your head* — no written materials or computer notes may be retained;
  - Your submission is authored solely by you, on a separate occasion.
- You should refer only to materials from this semester of CS 81 (lecture notes, handouts, textbooks, grutors, profs, etc.).
- Bring a writeup/printout to class on the due date. Illegible answers will get no credit.
- Make sure your submission includes your name!

---

1. Read pp. 109–118 and 123–127 (and optionally 119–122) of Sipser. Come up with (at least) two questions about the reading 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. We know several ways of specifying regular languages, including regular expressions, DFAs, and NFAs. Carefully describe strategies for solving the following problems, assuming the regular languages in question might be given in any one of these forms. You will probably want to use previously-seen algorithms as building blocks (the subset construction, DFA minimization, turning a DFA for a language into a DFA for prefixes of that languages, and so on).

(a) Given a regular language $L$, decide whether the empty string is in $L$.

(b) Given a regular language $L$, decide whether it is empty or not.

(c) Given a regular language $L$, decide whether it is finite or infinite.

(d) Given regular languages $L_1$ and $L_2$, decide whether they have at least one string in common.

(e) Given regular languages $L_1$ and $L_2$, decide whether $L_1 \subseteq L_2$.

(f) Given regular languages $L_1$ and $L_2$, decide whether $L_1 = L_2$.

3. 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.

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 2.16, page 129. (You might also want to think about Exercise 2.15.)

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

7. **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 $\epsilon$ 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):

$$
\begin{array}{rcl}
S & \to & \mathsf{if}\ E\ \mathsf{then}\ S\ \mathsf{else}\ S \\
 & | & \mathsf{begin}\ \mathsf{end} \\
 & | & \mathsf{begin}\ L\ \mathsf{end} \\
 & | & \mathsf{print}\ E \\
L & \to & S \\
 & | & L\ ;\ S \\
E & \to & \mathsf{true} \\
 & | & \mathsf{false}
\end{array}
$$

(b) The grammar

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

contains $\epsilon$ 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

$$
\begin{array}{rcl}
S & \to & Aa \\
A & \to & a \\
 & | & \epsilon
\end{array}
$$

satisfies Condition 1 but is not 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 $\epsilon$, there is a slightly more complicated Condition 2 that needs to be checked before we can be sure that a grammar is LL(1).]