CS 320
Computer Language Processing
Exercises: Week 3

March 7, 2025

Exercise 1 Recall the pumping lemma for regular languages:

For any language $L \subseteq \Sigma^*$, if L is regular, there exists a strictly positive constant $p \in \mathbb{N}$ such that every word $w \in L$ with $|w| \geq p$ can be written as $w = xyz$ such that:

- $x, y, z \in \Sigma^*$
- $|y| > 0$
- $|xy| \leq p$, and
- $\forall i \in \mathbb{N}.\ xy^i z \in L$

Consider the language $L = \{w \in \{a\}^* \mid |w| \text{ is prime}\}$. Show that L is not regular by using the pumping lemma.

Exercise 2 For each of the following languages, give a context-free grammar that generates it:

1. $L_1 = \{a^n b^m \mid n, m \in \mathbb{N} \land n \geq 0 \land m \geq n\}$
2. $L_2 = \{a^n b^m c^{n+m} \mid n, m \in \mathbb{N}\}$
3. $L_3 = \{w \in \{a, b\}^* \mid \exists m \in \mathbb{N}.\ |w| = 2m + 1 \land w_{(m+1)} = a\}$ (w is of odd length, has a in the middle)

Exercise 3 Consider the following context-free grammar G:

$$
\begin{aligned}
A &::= -A \\
A &::= A - id \\
A &::= id
\end{aligned}
$$

1. Show that G is ambiguous, i.e., there is a string that has two different possible parse trees with respect to G.

2. Make two different unambiguous grammars recognizing the same words, G_p, where prefix-minus binds more tightly, and G_i, where infix-minus binds more tightly.

3. Show the parse trees for the string you produced in (1) with respect to G_p and G_i.

4. Produce a regular expression that recognizes the same language as G.

Exercise 4 Consider the two following grammars G_1 and G_2:

$$\begin{aligned}
&G_1 : \\
&\quad S ::= S(S)S \mid \epsilon \\
&G_2 : \\
&\quad R ::= RR \mid (R) \mid \epsilon
\end{aligned}$$

Prove that:

1. $L(G_1) \subseteq L(G_2)$, by showing that for every parse tree in G_1, there exists a parse tree yielding the same word in G_2.

2. (Bonus) $L(G_2) \subseteq L(G_1)$, by showing that there exist equivalent parse trees or derivations.

Exercise 5 Consider a context-free grammar $G = (A, N, S, R)$. Define the reversed grammar $rev(G) = (A, N, S, rev(R))$, where $rev(R)$ is the set of rules is produced from R by reversing the right-hand side of each rule, i.e., for each rule $n ::= p_1 \ldots p_n$ in R, there is a rule $n ::= p_n \ldots p_1$ in $rev(R)$, and vice versa. The terminals, non-terminals, and start symbol of the language remain the same.

For example, $S ::= abS \mid \epsilon$ becomes $S ::= Sba \mid \epsilon$.

Is it the case that for every context-free grammar G defining a language L, the language defined by $rev(G)$ is the same as the language of reversed strings of L, $rev(L) = \{rev(w) \mid w \in L\}$? Give a proof or a counterexample.