Problem 1. [20 points] Solve Exercise 4.2.6(b) on page 148 of the textbook.

Problem 2. [20 points] Describe algorithms for the following decision problems. You may use any of the decision algorithms given in the class or the textbook as a building block. As usual, you may assume any convenient representation (regular expression, DFA, NFA, or ε-NFA) of the regular languages provided in the input.

a). Given regular languages \( L_1 \) and \( L_2 \), decide whether \( L_1 \subseteq L_2 \)?

b). Given a regular language \( L \) over \( \Sigma = \{0, 1\} \), does \( L \) contain all possible strings that begin with the substring 101?

Problem 3. [15 points] Consider the context-free grammar \( G = (V, T, P, S) \) with \( V = \{S, A, B, C, D\} \), \( T = \{0, 1, 2\} \), and the set of productions \( P \):

\[
\begin{align*}
S & \Rightarrow AB | CD \\
A & \Rightarrow 0A1 | \epsilon \\
B & \Rightarrow 2B | \epsilon \\
C & \Rightarrow 0C | \epsilon \\
D & \Rightarrow 1D2 | \epsilon
\end{align*}
\]

(a). Give a succinct description of the language: \( L_A = \{w \in T^* | A \Rightarrow w\} \).

(b). Give a succinct description of the language of \( G \).

(c). Show that the grammar \( G \) is ambiguous.

Problem 4. [10 points] Consider the following language over the alphabet \( \Sigma = \{0, 1\} \).

\[
L = \{0^i1^j | i \leq j \leq 2i \text{ and } i \geq 0\}
\]

This is the set of strings where all the 0's come before all the 1's, and the number of 1's is at least the number of 0's but no more than twice the number of 0's.

Provide a context-free grammar for \( L \).

Problem 5. [15 points] Solve Exercise 5.1.1(c) on page 182 of the textbook.

Problem 6. [20 points] In the book (Sections 5.4.1 and 5.4.2) we have discussed the issue of ambiguity of CFGs for arithmetic expressions written in the infix notation. In the infix notation, binary operators are written between the two operands (e.g., \( x + y \) or \( x \times y \)). There are other ways of writing operators such as the prefix notation where we would write \( +xy \) or \( x \times y \), and the postfix notation where we would write \( xy+ \) or \( xy\times \), to denote the same arithmetic expressions.
The following context-free grammar generates arithmetic expressions in the postfix notation which is used, for example, in the programming language APL. Notice the lack of parentheses in the grammar.

\[ S \rightarrow SS+ \mid SS- \mid SS* \mid x \mid y \]

(a). For the string \( xy*y \ x \ + \), find a derivation tree.
(b). For the string \( xy*y \ x \ - \), find a leftmost derivation.
(c). For the string \( xy*y \ x \ + \), find a rightmost derivation.
(d). Is this grammar ambiguous? Give a brief justification. (While a formal proof is not required, think about how you might give such a proof.)

Reading Assignment

We are done with Chapter 4. We are already into Chapter 5 and will soon be starting Chapter 6. While we will not cover Section 5.3 and Section 6.4 in class, they make for very interesting reading and are strongly recommended.