CMPU 240 · Spring 2026

Assignment 5

Submissions due: 24 March, 1:30 p.m.
Corrections due: 26 March, 1:30 p.m.

Exercise 1

Consider the language

$$L_1 = \{w \in \Sigma^* \mid n_{\mathtt{a}}(w) = n_{\mathtt{b}}(w)\},$$

where $\Sigma = \{\mathtt{a}, \mathtt{b}\}$, i.e., the language of all strings with an equal number of as and bs in any order.

In class, we identified four context-free grammars that failed to generate this language. For this exercise, design a CFG that does! Your answer shouldn't be a complicated grammar, but test it carefully.

You can test your CFGs using the online CFG editor.

Exercise 2

On Assignment 4, we introduced a language over $\Sigma = \{1, +, =\}$ that encodes unary addition:

$$ADD = \{1^m+1^n=1^{m+n} \mid m, n \in \mathbb{N}_0\}$$

For example:

3 + 4 = 7 would be encoded as 111+1111=1111111 $\in L$

7 + 1 = 8 would be encoded as 1111111+1=11111111 $\in L$

0 + 1 = 1 would be encoded as +1=1 $\in L$

You used the Pumping Lemma to prove that *ADD* isn't a regular language, but it turns out that it *is* context-free. Prove this by designing a CFG that generates *ADD*.

This requires no more than two variables and four rules, but it may require some experimentation to ensure your grammar *only* generates strings with the correct sum. Think about how you can control the *order* in which the string is generated.

Exercise 3

Here is a simple – but ambiguous – context-free grammar, G:

$$S \rightarrow \mathtt{a}S \mid \mathtt{b}S \mid S\mathtt{a} \mid \varepsilon$$

a. Give a concise, informal description of $L(G)$.

b. Demonstrate the grammar's ambiguity: For a single string in $L(G)$, give

- two leftmost derivations (that is, steps going from the start symbol to the final string, replacing the leftmost variable in each step) and
- the corresponding parse trees.

Keep it simple! Your string can be *very* short.

c. Give an unambiguous grammar for the same language.