CS533: Homework 5

Chapter 2: CFG

Question 1: Question 2.15

This question is not well-written. Consider if we add the rules $S \rightarrow SS \mid \epsilon$. Does this properly generate the star of the language?

Question 2: Question 2.16

In textbook question 2.2, you already proved that CFL’s are closed under union. So, just do concatenation and star.

Chapter 2: PDA

Question 3

Give a state diagram for a PDA that ensures that the first and last character of the string is the same (Question 2.4 b), the string has length at least 2, and the alphabet is \{0,1\}.

Also give a DFA (deterministic finite automata). Although you could give the same answer for both (as a DFA is a trivial PDA), make sure your PDA makes use of the stack in a meaningful way, and so should have fewer states than your DFA.

Question 4: Question 2.10

See the answer for Question 2.7 on page 160 for the level of detail expected in an ‘informal description’.

Question 5: Question 2.11

Show Languages are CF by constructing a CFG or PDA

Question 6: Question 2.25

Question 7: Question 2.20

In homework 4, you proved this where A is regular and B is regular. Make sure that you thoroughly understand the answer.

Do this proof by using a PDA for language A, and a DFA for language B. You will construct a PDA for $A/B$. 
Question 8: Convert PDA to CFG

![PDA Diagram]

Part a:
What language does this recognize? Hint: it is a language that was presented in the class slides.

Part b:
Give a grammar for this language. Note that the grammar that you will generate with Lemma 2.27 will be similar to your grammar, but might not be as concise.

Part c:
To convert a PDA to a CFG, lemma 2.27 requires it be first converted into a special form. State what that form is. Is the PDA above in this form?

Part d:
In Lemma 2.27, you need to add a set of rules corresponding to Figure 2.28 (2nd part). How many rules will this create? Explain your answer.

Write the rules that will actually be useful. Use $S_{ab}$ to format $S_{ab}$.

Part e:
Write all of the rules that correspond to the 3rd part (the epsilon rules).

Which ones will be useful?

Part f:
Write all of the rules that correspond to Figure 2.29 (first part). For each rule, explain why you were able to add it.

Part g:
What is the start variable?

Prelude to Pumping Lemma

Question 9: Question 2.35

This is a simpler version of the question in the textbook. In the next four steps, you will show that if $G$ generates a string $s$ with a derivation having at least $2^b$ steps, $s$ has a parse tree of height at least $b + 2$. Note: I am defining the height of a tree as the number of nodes (not edges) on the longest path from a leaf node to the start symbol.

Part a
First, prove that a string $s$ has $|s| \times 2 - 1$ steps in its derivation.

Part b
Instead of focusing on a sequence of derivations, we can arrange the derivations into a parse tree. A parse tree has nodes that are variables, and its leaves are the terminals. Since \( G \) is in Chomsky normal form, each node will either go to one leaf, or to two other nodes. The shortest parse tree has a height of 2 (start node goes to a terminal node).

For the following grammar, the parse tree will be very skinny.

\[
S \rightarrow AS | a \\
A \rightarrow a
\]

How high will the parse tree of the string \( a^n \) be?

**Part c**

Rather than thinking of really skinny parse trees, think of how bushy they can be. This of course will depend on the grammar.

We want to determine how short a parse tree can be for a string, for any grammar. In other words, we want to determine a minimum bound.

What is the longest string that a parse tree of height 2, 3, 4, 5 can generate? How about a parse tree of height \( n \).

**Part d**

Now argue that if \( G \) generates a string \( s \) with a derivation having at least \( 2^b \) steps, \( s \) has a parse tree of height at least \( b + 2 \).