COMP 598 Summer (May) 2020
Assignment 3
Due Date: 25^{th} May 2020

Prakash Panangaden

18^{th} May 2020

Please do all ten questions. Some are very easy and some are very hard. They are all worth the same however.

Question 1[10 points]

A sequence of parentheses is a sequence of `(` and `)` symbols or the empty sequence. Such a sequence is said to be *balanced* if it has an equal number of `(` and `)` symbols **and** in *every* prefix of the sequence, the number of left parentheses is greater than or equal to the number of right parentheses. Thus `((())())` is balanced but, for example, `())(` is not balanced even though there are an equal number of left and right parentheses. The empty sequence is considered to be balanced.

Consider the grammar

$$S \to (S)|SS|\varepsilon.$$

This grammar is *claimed to* generate balanced parentheses.

1. Prove that any string generated by this grammar is a properly balanced sequence of parentheses.
2. Prove that every sequence of balanced parentheses can be generated by this grammar.

Question 2[10 points] Consider the following context-free grammar

$$S \longrightarrow aS \mid aSbS \mid \varepsilon$$

This grammar generates all strings where *in every prefix* the number of a's is greater than or equal to the number of b's. Show that this grammar is ambiguous by giving an example of a string and showing that it has two different derivations.

Question 3[10 points] Prove that the grammar in the previous question generates only strings with the stated property and *all* strings with the stated property.

Question 4[10 points] We define the language $PAREN_2$ inductively as follows:

1. $\varepsilon \in PAREN_2$,
2. if $x \in PAREN_2$ then so are (x) and $[x]$,
3. if x and y are both in $PAREN_2$ then so is xy.

Describe a PDA for this language which accepts by empty stack. Give all the transitions.

Question 5[10 points] Consider the language $\{a^n b^m c^p | n \leq p \textbf{ or } m \leq p\}$. Show that this is context free by giving a grammar. You need not give a *formal* proof that your grammar is correct but you must explain, at least briefly, why it works.

Question 6[10 points] A *left-linear* grammar is one in which every rule has exactly one terminal and one non-terminal, with the terminal to the left of the non-terminal, on the right hand side or just a single terminal on the right hand side or the empty string. Here is an example

$$S \to aS|a|bB; \;\; B \to bB|b.$$

1. Prove that any regular language can be generated by a left-linear grammar. I will be happy if you show me how to construct a grammar from a DFA; if your construction is clear enough you can skip the straightforward proof that the language generated by the grammar and the language recognized by the DFA are the same.
2. If I allow the terminal on the right hand side to occur on the left or the right of the non-terminal I have what is called a linear grammar. Is every language generated by a linear grammar regular? If your answer is "yes" you must give a proof. If your answer is "no" you must give an example.

Question 7[10 points] The language $\{a^i b^j c^k | i \neq j \vee j \neq k\}$ is easily seen to be context-free. Give an outline of a PDA to recognize it. Show, however, that it is *not* a deterministic CFL.

Question 8[10 points] Show that a language over a *one-letter* alphabet is context free if and only if it is regular. Clearly any regular language is context free so what you are being asked to prove is that any context-free language has to be regular. Hint: Use the pumping lemma for context-free languages to express the language as a finite union of regular languages.

Question 9[10 points] Give an example of a context-free language L such that $\mathsf{lefthalf}(L)$ is **not** context free. The definition of $\mathsf{lefthalf}(L)$ is just as in the regular case. Note the contrast with regular languaages.

Question 10[10 points] For each of the following assertions give *brief* arguments indicating whether they are true of false. In each case I am talking about sets of positive integers.

a. For each $n \in \mathbb{N}$ we have a computable set C_n. The set $\bigcup_n C_n$ is computable. We assume that the collection of computable sets is *effectively given*: this means that there is an algorithm that reads a natural number n as input and outputs a description of a Turing machine that decides the set C_n. When you Google for the answer to this question you will get the following argument *which I will not accept*: Take any non-computable set S. It is the countable union of its singletons and each sinpleton is regular so clearly computable; hence the statement is false. I don't accept this argument because the family of singletons is *not effectively given* in the sense above.

b. For each $n \in \mathbb{N}$ we have a computably enumerable set C_n effectively given as described above. The set $\bigcup_n C_n$ is computably enumerable.