Assignment 5 Solutions

Prakash Panangaden
COMP 330 Winter 2021 McGill University

March 27, 2021

Question 1[20 points] Consider the two languages below:

$$L_1 = \{a^m b^n c^k d^j \mid m, n, k, j \geq 0 \text{ and } m = n \text{ and } k = j\}$$

and

$$L_2 = \{a^m b^n c^k d^j \mid m, n, k, j \geq 0 \text{ and } m = k \text{ and } n = j\}.$$

One of these languages is context-free but the other one is not. Identify which is which. For the context-free language give a context-free grammar and for the other one give a proof using the pumping lemma that it is not context-free.

Solution The language L_1 is context free but L_2 is not. Here is a grammar for L_1

$$S \to XY \mid \varepsilon \quad X \to aXb \mid \varepsilon \quad Y \to cYd \mid \varepsilon.$$

Here is the pumping lemma proof. Suppose that the demon chooses p. I choose $s = a^p b^p c^p d^p$. Consider possible ways of decomposing the string $s = uvwxy$ with $|vx| > 0$ and $|vwx| \leq p$.

Case 1: v or x crosses a block boundary. In this case choosing $i = 2$ gets the letters out of order.

Case 2: v lies in the block of a's. In this case x cannot lie in the block of c's as this would violate the condition $|vwx| \leq p$. Thus choosing $i = 2$ will mean that we no longer have equal lengths for the a and c blocks.

Case 3: v lies in the block of b's. Then x cannot be in the block of d's. Thus, once again, choosing $i = 2$ will make the b block and the d block have different lengths.

Any other cases are trivial variations. It is pointless to make a special case out of v may be ε etc.

Question 2[20 points] Consider the language below:

$$L = \{a^i b^j c^k \mid i, j, k \geq 0 \text{ and } (i \neq j \text{ or } j \neq k)\}.$$

Prove that this language is context-free by giving a grammar but not deterministic context-free by showing that its complement is not context-free. The grammar can be rather long to write out fully so it suffices to *explain* what you are doing and reduce it to similar cases and then say, "this case is just like what we have done before." Of course, if you want to write out all the rules you are free to do so. Some of you might find that easier than writing explanations. To prove that the complement is not context free can be done by a reduction argument to a familiar language that is known to be not context free. A direct pumping lemma proof would be painful.

Solution Here is a grammar for L:

$$S \to XC|AY$$

these break the language into the union of two languages; A allows one to produce any number of a's and C allows one to produce any number of c's. The rules for X will ensure that we have unequal number of a's and b's while the rules for Y will ensure that we have unequal number of b's and c's.

$$X \to P|Q \quad P \to aPb|R \quad Q \to aQb|R' \quad R \to aR|a \quad R' \to bR'|b$$

Here P ensures that there are more a's than b's while Q ensures that there are more b's than a's; thus both possibilities are taken into account. In each case, P and Q produce equal number of a's and b's and then finally R producess excess a's and R' produces excess b's. One can write exactly similar rules for Y. The rules for A and C are obvious.

My original solution for this was wrong, I am indebted to Omar Mohammed Wagih Sadek for pointing out the mistake. This correction was uploaded on the 27th of March.

The complement of this language is a mess but if we take the intersection with the regular language $a^*b^*c^*$ we get

$$\overline{L} \cap a^*b^*c^* = \{a^n b^n c^n | n \geq 0\}.$$

This language is well known to be non-context-free. The intersection of a CFL and a regular language is always a CFL, so the fact that $\overline{L} \cap a^*b^*c^*$ is not a CFL means that $\overline{L}$ is not a CFL. Thus L cannot be a DCFL since the latter are closed under complementation.

Question 3[20 points] Suppose that we have a language L defined over the alphabet $\{a, b, c\}$ and suppose that L is context-free. We define a new language $\mathsf{perm}(L)$ to be the set of all permutations of all words in L. For example, if $L = \{abc, aab\}$ then $\mathsf{perm}(L) = \{abc, acb, bac, bca, cab, cba, aab, aba, baa\}$. Show that $\mathsf{perm}(L)$ *need not be* context-free by giving **an example** of a language L that is context-free but where $\mathsf{perm}(L)$ is not context-free. You need not give a pumping lemma proof *if your example is just like one we have seen in class.*

Solution: Consider the regular language $(abc)^*$; this is regular and hence, certainly context-free. Now in this language we have equal numbers of a's b's and c's. Now when we construct

$\mathsf{perm}(L)$ we will obtain words of the form $a^n b^n c^n$ (among lots of other words). If we intersect $\mathsf{perm}(L)$ with $a^*b^*c^*$ we get our favourite non context-free language

$$\{a^n b^n c^n | n \geq 0\}.$$

Question 4[20 points] We have seen in class that the language $L_1 = \{a^{i+j} b^{j+k} c^{k+i} | i, j, k \geq 0\}$ over the alphabet $\{a, b, c\}$ is context free. Consider the language

$$L_2 = \{a^{i+j} b^{j+k} c^{k+l} d^{i+l} | i, j, k, l \geq 0\}$$

over the alphabet $\{a, b, c, d\}$. Is this language also context free? If so give a context-free grammar for it; if not prove that it is not context-free using the pumping lemma.

Solution:

This language is context-free. We can design a grammar for it along the same lines as we did in class.

$$S \to aSd|ABC|\varepsilon$$

$$A \to aAb|\varepsilon \qquad B \to bBc|\varepsilon \qquad C \to cCd|\varepsilon$$

Question 5[20 points] We assume that the alphabet Σ has two or more letters. Prove that the complement of the set of palindromes is context-free: this is the set of words $\{w \in \Sigma^* | w \neq w^{\mathbf{rev}}\}$. You do not have to give a formal proof but you **must explain your answer**. A correct answer without an explanation will only get half the marks.

Solution:

We will design a grammar for this language. Here is the grammar:

$$S \to aSa|bSb|aSb|bSa|aTb|bTa$$

$$T \to aTa|aTb|bTa|bTb|a|b|\varepsilon$$

The idea is that S allows any kind of pairs but when it switches to T, as it *must eventually*, it produces a *mismatched* pair. Thus, at some point the word generated must have a mismatch when reading from left to right and from right to left; so it cannot be a palindrome.