Problem 1 - 7.2.1. (b) Show that $B = \{a^n b^n c^i | i \leq n\}$ is not context free.

Proof: Let $z = a^k b^k c^k$, and suppose $z = uvwxy$ such that $vx \neq \epsilon$ and $|vwx| \leq k$. There are several ways we can do this:

(1) v and x contain only a or only b. In this case, take $r = uv^2wx^2y = a^l a^{(k-l)k} b^k c^k$, or a similar formulation for b. Since there are more as than bs, $r \notin B$.

(2) v and x contain both as and bs. Take $r = uv^0wx^0y$. Since there are k bs, and the length of vwx is bounded by k, if vwx contains as, it cannot also contain cs. Hence, r has fewer as and fewer bs than cs, and therefore $r \notin B$.

(3) v and x contain both bs and cs. By the same reasoning as above, vwx can only contain bs and cs. Take $r = uv^0wx^0y$, which removes some of the bs, but keeps just as many as. Since there are more as than bs, r is not an element of B.

(4) finally, v and x can contain only cs. In this case, we take $r = uv^2wx^2y$. Since v and x are composed of only cs, we increase the number of cs, keeping the number of as and bs constant, so r has more cs than as or bs. Hence, $r \notin B$.

Since we can always choose some i such that $uv^iwx^iy \notin B$, we have proved by the pumping lemma that B is not context free.

(c) Show that $C = \{c^p | p \text{ is prime}\}$ is not context free.

Proof: for any given k, let $z = 0^p$, where p is a prime greater than $k + 2$, so $|z| > k$. Suppose $z = uvwxy$ such that $|vx| \neq 0$ and $|vwx| \leq k$. Then let $i = p - |v| - |x|$. We claim that $r = uv^iwx^iy \notin C$.

First, let us define $|u| = a, |v| = b, |w| = c, |x| = d$ and $|y| = e$. The length of r is

$$\begin{aligned} |uv^iwx^iy| &= a + c + e + (p - b - d)(b + d) \\ &= p - b - d + (p - b - d)(b + d) \\ &= (p - b - d)(b + d + 1) \end{aligned}$$

Since $vw \neq \epsilon$, we know that $(b+d+1) > 1$. We chose $p > k+2$, so $b+d \leq k < k+2 = p$, which means that $p-b-d \geq k+2-b-d > 1$. Hence $|r|$ has two factors, neither of which is 1, and so r must be composite, and therefore not in C. We conclude that, by the pumping lemma, C is not context free.

(f) Show that $F = \{ww^Rw \,|\, w \text{ is binary}\}$ is not context free.

Proof: let $w = 1^k0^k$. We have three ways of dividing $z = ww^Rw = 1^k0^{2k}1^{2k}0^k$ into $uvwxy$.

- First, suppose vwx is part of the first block of ones (that is, it contains no zeroes). If we take $r = uv^2wx^2y$, then we have increased the number of ones in the first block. We know that our new w' must end in a zero, since r ends in a zero, so the first block of zeroes separates w' from $(w')^Rw'$. There is no way of forming $w'(w')^Rw'$ from this, so we know that $r \notin B$.
- A similar argument works if vwx is part of any of the other three contiguous blocks of ones or zeroes.
- Suppose vwx lies on the boundary between two blocks. Again, because of length restrictions, vwx can only lie on one boundary, and can't span three blocks. Let $r = uv^0wx^0y$. We know that w' must begin with a one and end in a zero, so we still have $w' = 1^m0^n$. However, r is no longer of the form $1^a0^{2a}1^{2a}0^a$, so r cannot be in F.

Hence, by the pumping lemma, F is not context free.

Question 2 (7.2.5)

(a) Show $\{0^i 1^j 0^k | j = max(i,k)\}$ is not a CFL using Ogden's lemma:
We begin selecting $z = 0^{2n}1^{2n}0^n$ and marking all the last 0's (for example, $z = 0^{2n}1^{2n}\hat{0}^n$). If we select v or x to have both 0's and 1's in it, we can instantly see that our syntax is no longer correct. If we select v to be 0 or 1 then we can also see that our original assumption that $j = max(i,k)$ no longer holds because one of the numbers will grow while the other will remain the same. Our only choice is to have v and x exclusively contain $\hat{0}$. When we start pumping it, at some point the number of $\hat{0}$ (corresponding to k) will grow to be larger than i and j, therefore once again breaking our condition that $j = max(i,k)$. Therefore, the language is not a CFL.

(b) Show $\{a^n b^n c^i | i \neq n\}$ is not a CFL using Ogden's lemma:
Consider the case where we choose z to be $a^n b^n c^{n!+n}$ (where n is the constant from Ogden's lemma). We mark all the a's and all the b's. We first note that if v or x contain a mix of a's and b's, then we can see that with $i = 2$, the structure of the resulting grammar is no longer correct. We now look at the case where $v = a^\alpha$ and $x = b^\beta$. If $\alpha \neq \beta$ then we can also see that the number of a's and b's will be different, therefore $\alpha = \beta$. We can now call $\gamma = \alpha = \beta$ and see that our final string will be of the form

$$a^{n+\gamma(i-1)} b^{n+\gamma(i-1)} c^{n!+n}$$

Therefore, if we set the exponents of a or b equal to c, we get that

$$n + \gamma(i-1) = n! + n$$

$$\gamma(i-1) = n!$$

$$i - 1 = \frac{n!}{\gamma}$$

Since $\gamma \leq n$ we know that the right side divides evenly and therefore we can pick an i that satisfies this constraint, therefore our original constraint of $\{a^n b^n c^i | i \neq n\}$ is not satisfied, therefore we do not have a CFL.

CS 381/481 Homework 9

7.4.5

Let N_{ijA} denote the number of distinct parse trees for substring $a_i \dots a_j$ of the input w, starting from variable A (i.e., with A as the root of the parse tree). Note that we are using A here as a metavariable, not any particular variable in G that might have been named A. N_{1nS}, where $n = |w|$ and S the starting variable of G, is the value we are interested in. We can augment the CYK algorithm to compute each N_{ijA} as we compute the corresponding X_{ij}. That is, after computing X_{ij} in CYK, we proceed to compute N_{ijA} for each variable A.

Initially, we set all N_{ijA} to 0.

For the base case, we can compute the first row of N as follows. N_{iiA} is 1 if $A \rightarrow a_i$ is a production of G. Otherwise, N_{iiA} remains 0.

To compute N_{ijA}, $j - i > 0$, we look at each of the pairs $(X_{ii}, X_{i+1,j}), \dots, (X_{i,j-1}, X_{jj})$ the same way plain CYK did. For each pair, we look at each element of the cross product of that pair. That is, for $(X_{ik}, X_{k+1,j})$, we consider all pairs (B, C) such that $B \in X_{ik}$ and $C \in X_{k+1,j}$. If $A \rightarrow BC$ is a production, we increment N_{ijA} by $N_{ikB} \times N_{k+1,j,C}$.

When the algorithm completes, N_{1nS} would contain the solution.

For the special case when $w = \varepsilon$, this algorithm won't work, but the answer is easy. It's 1 if $S \rightarrow \varepsilon$ is a production, 0 otherwise.

Problem 4 (a). The set $\{a^ib^jc^i | i, j \geq 0\}$ is a DCFL because we can construct a DPDA M accepting it. Let $M = (\{q_S, q_A, q_B, q_C\}, \{a, b, c\}, \{A, Z_0\}, \delta, q_S, Z_0, \emptyset)$ be a DPDA that accepts by empty stack. We define δ as follows:

$$\begin{aligned}
\delta(q_S, \epsilon, Z_0) &= (q_A, \epsilon) \\
\delta(q_A, a, A) &= (q_A, AA) \\
\delta(q_A, b, A) &= (q_B, A) \\
\delta(q_B, b, A) &= (q_B, A) \\
\delta(q_B, c, A) &= (q_C, \epsilon) \\
\delta(q_C, c, A) &= (q_C, \epsilon)
\end{aligned}$$

Intuitively, the DPDA pushes an A onto the stack for each a it sees at the beginning of the input. Then it reads the bs without altering the stack, and then for each c it reads, it pops an A off the stack. If the number of as equals the number of cs, then the stack will be empty at the end of the input.

Problem 4 (b). Assume WLOG that L is a DCFL accepted by a DPDA $M = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$, which accepts by final state. Then the DPDA $M' = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, Q - F)$ accepts $\overline{L}$. To see this, notice that running M on any input string w will put the machine in exactly one state q_w, since it is deterministic. Thus, after running M' on input w it will also end in state q_w. This state is final in M iff it is not final in M', and so w is in $L(M)$ iff it is not in $L(M')$. So $L(M') = \overline{L(M)} = \overline{L}$.

Problem 4 (c). We have shown that $L_1 = \{a^ib^jc^i | i, j \geq 0\}$ is a DCFL, and we can similarly show that $L_2 = \{a^ib^ic^j | i, j \geq 0\}$ is a DCFL. However, we know that $L_1 \cap L_2 = \{a^ib^ic^i | i \geq 0\}$ is *not* a CFL, and thus it is not a DCFL either (since DCFLs are a subset of CFLs).

Problem 4 (d). We can show that $L_1 = \{a^ib^jc^k | i, j, k \geq 0, i \neq j\}$ and $L_2 = \{a^ib^jc^k | i, j, k \geq 0, j \neq k\}$ are both DCFLs. Assume for a contradiction that $L_1 \cup L_2$ is a DCFL. It follows by (b) that its complement, $\overline{L_1} \cap \overline{L_2}$ is a DCFL, and thus is a CFL. Since CFLs are closed under intersection with regular sets, we have that $\{a^*b^*c^*\} \cap (\overline{L_1} \cap \overline{L_2})$ is also a CFL. But this set is equal to $\{a^ib^ic^i | i \geq 0\}$, which we already know to be not context-free.