Problem Set 7 Solutions

Problem 1.

a. *Prove that* $L_a = \{a^i b^j c^k : j = \max\{i, k\}\}$ *is not context free.*

Suppose for contradiction that L_a were context free. Let N be the "N" of the pumping lemma for context free languages. Consider the string $w = a^N b^N c^N$. Suppose $w = uvxyz$, where $|vxy| \leq N$ and $|vy| \geq 1$. If vy contains only a's or vy contains only c's, then pump up: the string $uv^2xy^2z \notin L_a$. Suppose vy contains only b's. Then we can pump either way to get a string not in L_a. Suppose v contains two different letters or y contains two different letters. Then uv^2xy^2z is not even of the form $a^*b^*c^*$, so certainly it is not in L_a. Finally, suppose ($v \in a^+$ and) $y \in b^+$, or $v \in b^+$ (and $y \in c^+$). Then we can pump down and there will be too few b's. By $|vwy| \leq N$, these are all the possible cases. So in all cases there is some i for which $uv^ixy^iz \notin L$, a contradiction.

b. *Prove that* $L_b = \{b_i\#b_{i+1} : b_i$ *is* i *in binary,* $i \geq 1\}$ *is not context free.*

Suppose for contradiction that L_b were context free. Let N be the "N" of the pumping lemma for context free languages. Consider the string $w = 1^N0^N\#1^N0^{N-1}1 \in L_b$. Suppose $w = uvxyz$, where $|vxy| \leq N$ and $|vy| \geq 1$. If vxy does not contain a #, then pumping either way will cause a contradiction (increasing or decreasing one of the numbers without touching the other). If the # is contained in v or y, then pumping either way leads to a string not even in $(a \cup b)^*\#(a \cup b)^*$, i.e., a string definitely outside of L_b. Because of the $|vxy| \leq N$ condition, the only remaining possibility is for $v = 0^i$ and $y = 1^j$, $i, j \geq 1$, to fall on opposite sides of the "#." But pumping up in this case means multiplying the left hand number by some power of two, while it never means multiply the right hand number by some power of two. Thus the pumped-up strings will not remain with the right number one more than the left.

Problem 2. *If* A *and* B *are languages define* $A \Diamond B = \{xy | x \in A$ *and* $y \in B$ *and* $|x| = |y|\}$.

a. *Show that if* A *and* B *are regular then* $A \Diamond B$ *might not be regular.*

Let $A = L(a^*)$ and let $B = L(b^*)$. Then $A \Diamond B = \{a^n b^n : n \geq 0\}$, which is not regular.

b. *Show that if* A *and* B *are regular then* $A \Diamond B$ *is context free.*

Let M_A be a DFA accepting A and let M_B be a DFA accepting B. We construct a PDA M for $A \Diamond B$ as follows. The machine M starts out in the start state of M_A. It reads in the input, processing it by M_A. As it reads each character of input it pushes a character onto the stack. At some point in time the machine M nondeterministically *guesses* that it is at the midpoint of the input string. At this point—assuming that M is currently in a *final* state of M_A—it *jumps* to the start state of M_B. The machine M now reads in the remainder of the input, processing it according to machine M_B. With each character read in it pops one character off the stack. The machine M accepts if the input is exhausted, the stack is empty, and the machine is now in a final state of M_B.

Here is the construction written out more formally. Let $M_A = (Q_A, \Sigma, \delta_A, q_A, F_A)$ be a DFA for A and let $M_B = (Q_B, \Sigma, \delta_B, q_B, F_B)$ be a DFA for B. Assume Q_A and Q_B are disjoint (rename states to make them disjoint if they are not already so). We construct a PDA $M = (Q_A \cup Q_B, \Sigma, \{\$\}, \delta, q_A, F_B)$ by asserting that $\delta(q, a, \varepsilon) = (\delta_A(q, a), \$)$ for each $q \in Q_A$ and $a \in \Sigma$ (read an input character, process it by machine M_A, and push an \$ onto the stack); $\delta(q, \varepsilon, \varepsilon) = (q_B, \varepsilon)$ for each $q \in F_A$ (guess that we're at the middle of the input and jump to the start state of machine M_B); and $\delta(q, a, \$) = (\delta_B(q, a), \varepsilon)$ for each $q \in Q_B$ and $a \in \Sigma$. We assume the empty-stack convention for the PDA: it only accepts if the stack is empty.

Problem 3. *Formally specify (draw a transition diagram for) a Turing machine that, when started on an initially empty, two-way infinite tape, will eventually visit any cell. Make your machine be as simple and have as few states as you can.*