CS3133

HW#7

DUE: Monday, October 10

(20 points) Is $L=\left\{u2v \mid u,v \in \{0,1\}^* \wedge u \neq v\right\}$ a CFL? Note that $0111201111 \in L$, $0111120111 \in L$ and $010120111 \in L$. Justify your answer. Hint: In what ways can *u* be different than *v*?

CS3133
Solutions to HW#7

$L = \{u2v \mid u,v \in \{0,1\}^* \wedge u \neq v\}$ is a CFL. We note that there are two ways for $u \neq v$:

- if $|u| \neq |v|$, or,
- if $u = u_1 ... u_m$ and $v = v_1 ... v_n$ and $u_k \neq v_k$ for $k \leq \min(m,n)$,

These conditions are not exclusive. We first give a CFG to generate
$L_0 = \{u2v \mid u,v \in \{0,1\}^* \wedge |u| \neq |v|\}$ We then give a PDA to accept

$L_1 = \{u_1 ... u_m 2 v_1 ... v_n \mid u_1,...,u_m,v_1,...,v_n \in \{0,1\} \wedge (\exists k)(1 \leq k \leq \min(m,n) \wedge u_k \neq v_k)\}$.

Finally we appeal to the fact that CFLs are closed under union to establish that
$L = \{u2v \mid u,v \in \{0,1\}^* \wedge u \neq v\} = L_0 \cup L_1$ must be a CFL.

We can generate strings in L_0 by generating all strings *u*, *v* with $|u| < |v|$ and then with $|u| > |v|$. In the following grammar, *A* generates all strings *u*2*v* with |*u*| =|*v*|, and *B* generates all nonempty strings over {0, 1}.

$$S \to AB \mid BA$$
$$A \to 0A0 \mid 0A1 \mid 1A0 \mid 1A1 \mid 2$$
$$B \to 0B \mid 1B \mid 0 \mid 1$$

We accept L_1 with PDA $M=(\{s, q_0, q_1, p_0, p_1, r, t\}, \{0,1,2\}, \{\perp, 0\}, \delta, s, \perp, \{t\})$ which accepts by final state. State *s* counts until *k* by pushing *k*-1 0's onto the stack. Eventually it "guesses" that it is reading a u_k such that $u_k \neq v_k$ by effectively storing u_k in a register and jumping to state q_{u_k}.

$$\delta(s,0,\perp) = \{(s,0\perp),(q_0,\perp)\}$$
$$\delta(s,0,0) = \{(s,00),(q_0,0)\}$$
$$\delta(s,1,\perp) = \{(s,0\perp),(q_1,\perp)\}$$
$$\delta(s,1,0) = \{(s,00),(q_1,0)\}$$

From state q_{u_k} we consume 0's and 1's until encountering a 2, in which case we enter state p_{u_k}.

$$\delta(q_0,0,\perp) = \delta(q_0,1,\perp) = \{(q_0,\perp)\}$$
$$\delta(q_1,0,\perp) = \delta(q_1,1,\perp) = \{(q_1,\perp)\}$$
$$\delta(q_0,0,0) = \delta(q_0,1,0) = \{(q_0,0)\}$$
$$\delta(q_1,0,0) = \delta(q_1,1,0) = \{(q_1,0)\}$$
$$\delta(q_0,2,\perp) = \{(p_0,\perp)\}$$
$$\delta(q_0,2,0) = \{(p_0,0)\}$$
$$\delta(q_1,2,\perp) = \{(p_1,\perp)\}$$

$$\delta(q_1, 2, 0) = \{(p_1, 0)\}$$

In state p_{u_k} we consume $v_1 ... v_{k-1}$ by popping *k*-1 0's off the stack. Once $v_1 ... v_{k-1}$ has been read (is on the top of the stack), p_{u_k} compares v_k to u_k. If $v_k \neq u_k$, then the machine enters state *r*, which consumes the rest of the input.

$$\delta(p_0, 1, \perp) = \{(r, \perp)\}$$

$$\delta(p_1, 0, \perp) = \{(r, \perp)\}$$

In state *r* we assure that the unconsumed part of the input string is all 0's and 1's, and then we enter the final state *t*.

$$\delta(r, 1, \perp) = \delta(r, 0, \perp) = \{(r, \perp), (t, \perp)\}$$