COT 5310 Assignment 7 Key, Fall 2007

1. Present the description of a PDA (in words) that accepts L_A (see page 253 of Notes). You may assume that $[i]$ is a single symbol.

   A PCP instance over Σ consists of two vectors x, y over Σ^* where $|x| = |y| = n$. Let $I = \{1, \ldots, n\}$ and be disjoint from Σ (we use just i instead of $[i]$ to denote the index). L_A is the language generated by the grammar

$$A \to x_i\, A\, i \ \mid \ x_i\, i \quad \text{for } i \in I$$

   over the alphabet $\Sigma' = \Sigma \cup I$. So

$$L_A = \left\{ x_{i_1} x_{i_2} \ldots x_{i_{k-1}} x_{i_k} i_k i_{k-1} \ldots i_2 i_1 \ : \ 1 \le i_j \le n \text{ for } 1 \le j \le k \text{ and } i_j \in I \text{ and } x_{i_j} \in \Sigma^* \right\}.$$

   To make a PDA that accepts this language, use $\Sigma' = \Sigma \cup I$ as the stack symbols, with Z_0 as the bottom of stack marker. We'll push symbols until we reach an element $i \in I$. Then we'll pop $(x_i)^R$ from the top of the stack and repeat. If we run out of input and have an empty stack, then we're in L_A.

   1) Read in elements of Σ and push them onto the stack. When encountering some $i \in I$, push it onto the stack and move to step 2).
   2) Pop i from the stack and check if ${w_i}^R$ is on top of the stack. If it isn't, we're not in L_A so transition to a non-accepting state r).
      If ${w_i}^R$ is on top, and the bottom of stack symbol is exposed, transition to an accepting state a).
      If ${w_i}^R$ is on top, and the bottom of stack symbol isn't exposed, transition to state 3).
   3) Read in an input, if it's in I then push it onto the stack and go to state 2). If it's not in I then we're not in the language, so move to state r).

   r) Consume input without changing the stack, don't accept.

   a) Accept, unless there's input left, in which case transition to r).

   Figure 1 illustrates in more detail how the "If ${w_i}^R$ is on top" is determined. For $w \in \Sigma^*$, w_k denotes the k^{th} character of w.

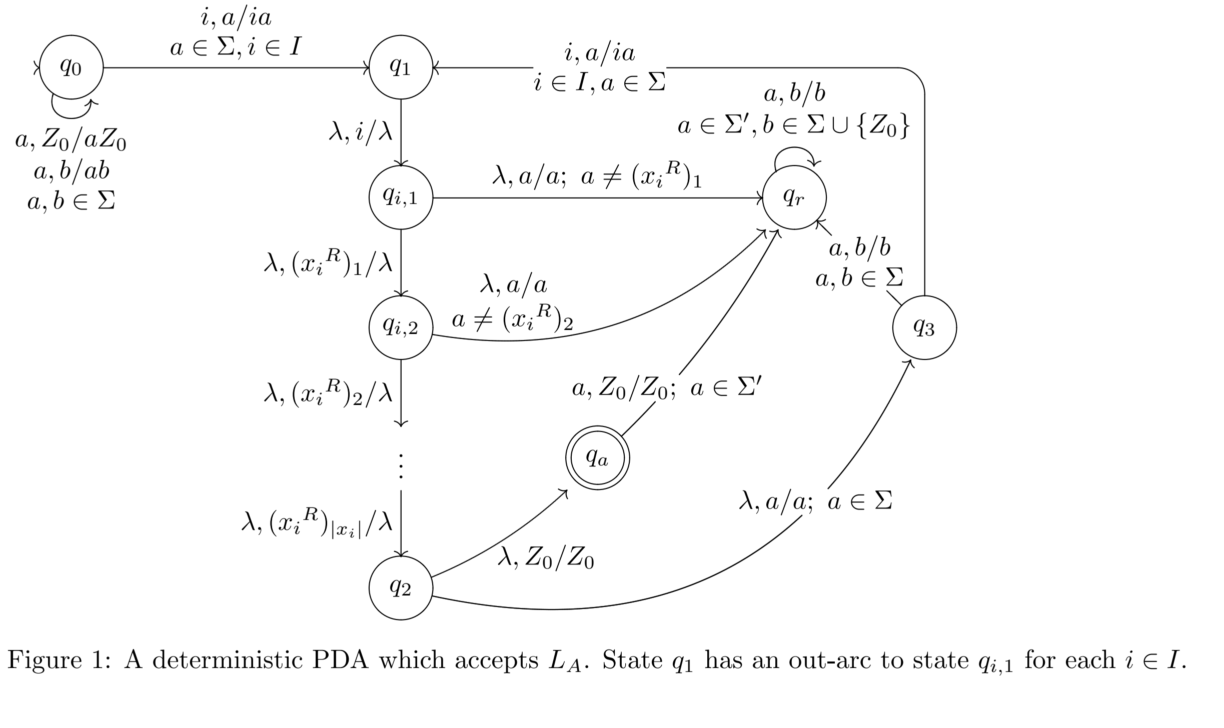

Figure 1: A deterministic PDA which accepts L_A. State q_1 has an out-arc to state $q_{i,1}$ for each $i \in I$.

2. Present the description of a PDA (in words) that accepts $\overline{L_A}$ (see page 253 of Notes).

   In problem 1. the state q_r isn't necessary because the PDA accepts by having no input left and being in an accepting state (at least that's the version of PDA's we're using). However, it helps here because all non-accepting inputs with some element of I in them end up in q_r while all accepting inputs end up in q_a, so to accept the elements of $\overline{L_A}$ which contain an element of I, it suffices to make q_r accepting and q_a non-accepting. To take care of the case where no element of I occurs, make q_0 accepting.

   This is also a deterministic PDA since $|\delta(q,a,Z)| \leq 1$ and $|\delta(q,\lambda,Z)| \leq 1$ and $\delta(q,\lambda,Z) \neq \varnothing$ implies that $\delta(q,a,Z) = \varnothing$ for all $q \in Q$, $a \in \Sigma$, $Z \in \Gamma$.

   The final PDA is $(Q, \Sigma', \Gamma, \delta, q_0, Z_0, F)$ where

   - $Q = \{q_0, q_1, q_2, q_3, q_a, q_r\} \cup \{q_{i,k} \ : \ i \in I \text{ and } 1 \leq k \leq |x_i|\}$
   - $\Sigma' = \Sigma \cup I$
   - $\Gamma = \Sigma'$
   - $\delta =$ See Figure 2
   - $F = \{q_0, q_r\}$

   Here's the transition function in graphical notation.

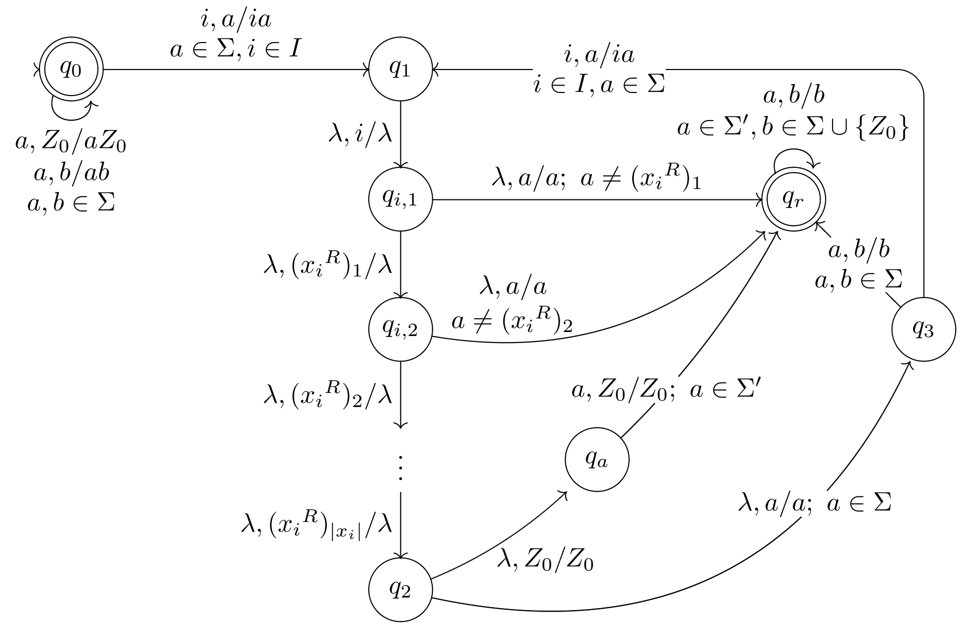

Figure 2: A deterministic PDA which accepts $\overline{L_A}$. State q_1 stil has an out-arc to state $q_{i,1}$ for each $i \in I$.

3. Use (2) to show that it is undecidable to determine of an arbitrary CFL, L, over the alphabet A, whether or not $L = A^*$.

   Parts 1. and 2. show that L_A, L_B and $\overline{L_A}, \overline{L_B}$ are all context free languages where L_A corresponds to the x vector of the PCP and L_B corresponds to the y vector of the PCP. Then

$$L_A \cap L_B = \varnothing \Leftrightarrow \overline{L_A} \cup \overline{L_B} = \Sigma^*.$$

   So if determining that an arbitrary CFL L is equal to A^* is decidable, then determining that $\overline{L_A} \cup \overline{L_B} = \Sigma^*$ would be decidable since $\overline{L_A} \cup \overline{L_B}$ is a CFL. However this would also mean determining $L_A \cap L_B = \varnothing$ is decidable, but this would let us determine if the PCP instance had a solution, an impossibility. Therefore determining $L = A^*$ is undecidable where L is a CFL.

4. Prove that Post Correspondence Systems over $\{a\}$ are decidable.

   Given a PCP system with alphabet $\{a\}$ and vectors x, y each of size n, we show an algorithm to determine if a solution exists. There are three general cases which must be considered.

   If there exists an i such that $x_i = y_i$ then using i once is a solution.

   If for all indices i, $|x_i| < |y_i|$, or for all indices i, $|x_i| > |y_i|$ then no solution exists because the lengths can never match.

   The final case is the interesting one, in which there exists an i and a j such that $|x_i| < |y_i|$ and $|x_j| > |y_j|$. Then let

$$\begin{aligned} |x_i| &= b & \qquad |x_j| &= d \\ |y_i| &= c & \qquad |y_j| &= e \\ c - b &= u & \qquad d - e &= v \end{aligned}$$

   and use v applications of index i and u applications of index j. Since there's only 1 character, we only care about the number of characters. This gives

$$\begin{aligned} bv + du &= bv + (v + e)u = bv + uv + eu & \qquad &\ldots\text{from } x \\ cv + eu &= (u + b)v + eu = bv + uv + eu & \qquad &\ldots\text{from } y \end{aligned}$$

   and thus a solution exists.