Lecture Notes
Converting a CFG into an equivalent PDA

Prof. Amit Chakrabarti
Computer Science Department
Dartmouth College

Your textbook (Sipser) states, in Lemma 2.21, that any context-free grammar (CFG) can be converted into an equivalent pushdown automaton (PDA). The proof given there takes a CFG G and constructs a certain "3-state" PDA M, and gives intuition for why $\mathcal{L}(M) = \mathcal{L}(G)$. (In fact, the number of states could be much greater than 3, once we unroll the shorthand notation that allows us to push multiple symbols on the stack in a single move.) The textbook stops short of giving a full formal proof, though. Here is a formal proof.

Theorem: For the PDA M constructed in the textbook (Figure 2.24), we have $\mathcal{L}(M) = \mathcal{L}(G)$.

Proof: First, we introduce some notation. For $y \in \Sigma^*$ and $\gamma \in (V \cup \Sigma)^*$, we let $M[y, \gamma]$ denote the statement "M can be in state q_{loop}, having read the prefix y of the input string, and with $\gamma\$$ on its stack." Note that $M[x, \varepsilon]$ iff M can make the transition to q_{accept} after reading x, i.e., iff $x \in \mathcal{L}(M)$.

Part 1: $\mathcal{L}(G) \subseteq \mathcal{L}(M)$: Suppose $x \in \mathcal{L}(G)$. Then $S \stackrel{*}{\Rightarrow} x$ in n steps for some positive integer n, via a leftmost derivation. Let $S = s_0 \Rightarrow s_1 \Rightarrow s_2 \Rightarrow \cdots \Rightarrow s_n = x$ be such a leftmost derivation. Suppose

$$s_i \;=\; y_i A_i \gamma_i \,,$$
$$\text{where } y_i \in \Sigma^*, A_i \in V, \text{ and } \gamma_i \in (V \cup \Sigma)^* \,, \quad \text{for } 0 \le i < n \,,$$
$$\text{and } y_n = x, A_n = \gamma_n = \varepsilon \,.$$

In other words, A_i denotes the leftmost variable in s_i (or ε, in the case $i = n$ when s_i has no variables). We claim that $M[y_i, A_i\gamma_i]$ for all i, $0 \le i \le n$. In particular, this proves that $M[x, \varepsilon]$, i.e., that $x \in \mathcal{L}(M)$. The proof of the claim is by induction on i.

The base case is $i = 0$. The transition out of q_{start} shows that M can be in state q_{loop} having read no input and with $S\$$ on its stack, i.e., $M[\varepsilon, S]$, Note that $y_0 = \gamma_0 = \varepsilon$ and $A_0 = S$; therefore $M[y_0, A_0\gamma_0]$.

For the induction step, suppose we have shown $M[y_i, A_i\gamma_i]$, for some i with $0 \le i < n$. The derivation step $s_i \Rightarrow s_{i+1}$ must expand the leftmost variable in s_i, i.e., A_i. Let $A_i \to \alpha_i$ be the CFG rule used in this step. Then

$$y_{i+1} A_{i+1} \gamma_{i+1} = s_{i+1} = y_i \alpha_i \gamma_i \,.$$

Since y_i is a prefix of y_{i+1}, we can write $\alpha_i\gamma_i = z_i A_{i+1}\gamma_{i+1}$ for some $z_i \in \Sigma^*$ (note, in particular, that this continues to hold even if $i + 1 = n$). This implies $y_{i+1} = y_i z_i$. Since M has a loop transition at state q_{loop} that can pop A_i and push α_i, we have $M[y_i, \alpha_i\gamma_i]$, i.e., $M[y_i, z_i A_{i+1}\gamma_{i+1}]$. Finally, since M has a loop transition at q_{loop} that can read any input character $a \in \Sigma$ while popping a off the stack, and since $y_i z_i = y_{i+1}$ is a prefix of the input x, we have $M[y_i z_i, A_{i+1}\gamma_{i+1}]$, i.e., $M[y_{i+1}, A_{i+1}\gamma_{i+1}]$. This completes the induction step and the proof of Part 1.

Part 2: $\mathcal{L}(M) \subseteq \mathcal{L}(G)$: The proof of this is similar to the proof in Part 1. The details are left to you as an exercise. (It's good practice; please try writing out the details.)

Addendum: The Lashof-Regas Lemma Here is the formal proof that Matthew had wanted to see in Lecture 17.

Let $M = (Q, \Sigma, \Gamma, \Delta, r, \{f\})$ be a PDA in normal form. Recall that we wrote $(q, s) \xrightarrow{a} (q', s')$ if $a \in \Sigma_\varepsilon$ could take M from the configuration (q, s) to the configuration (q', s'). We wanted to show that if a string $x \in \Sigma^*$ can take M from (q_0, ε) to (q_n, ε), then, for any stack symbol $b \in \Gamma$, x can also take M from (q_0, b) to (q_n, b). This is a consequence of applying the following lemma to each step of the computation chain $(q_0, s_0) \xrightarrow{a_1} (q_1, s_1) \xrightarrow{a_2} \cdots\cdots \xrightarrow{a_n} (q_n, s_n)$.

Lemma: Suppose $q, q' \in Q$, $a \in \Sigma_\varepsilon$, $b \in \Gamma$ and $s, s' \in \Gamma^*$. If $(q, s) \xrightarrow{a} (q', s')$ then $(q, sb) \xrightarrow{a} (q', s'b)$.

Proof: By definition of the "$\xrightarrow{a}$" relation, there exist $c, d \in \Gamma$ and $t \in \Gamma^*$ such that $s = ct$, $s' = dt$ and $(q', d) \in \delta(q, a, c)$. Therefore, we also have $sb = ctb$, $s'b = dtb$ and $(q', d) \in \delta(q, a, c)$. Since tb is just another string in Γ^*, this proves $(q, sb) \xrightarrow{a} (q', s'b)$.