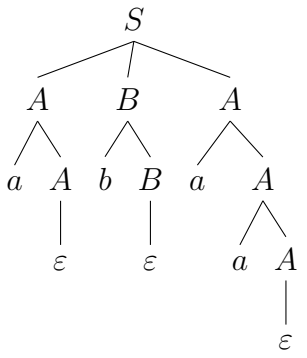
CS 360 — Introduction to the Theory of Computing
Assignment 3

University of Waterloo, Spring 2018

Due 5:00 PM, June 15, 2018.

1. (a) Here is the parse tree for $abaa$.



   (b) To see that G is ambiguous, we observe that the word aa has two leftmost derivations:

$$S \Rightarrow ABA \Rightarrow aABA \Rightarrow aaABA \Rightarrow aaBA \Rightarrow aaA \Rightarrow aa,$$

   and

$$S \Rightarrow ABA \Rightarrow BA \Rightarrow A \Rightarrow aA \Rightarrow aaA \Rightarrow aa.$$

   (c) The CFG G generates the language $\{a^i b^j a^k \mid i, j, k \geq 0\}$. To see this, note that from A, one can generate words of the form a^i with $i \geq 0$ and from B, words of the form b^j with $j \geq 0$ are generated. Then from S, there is only one production $S \to ABA$. Thus any word w with $S \Rightarrow^* w$ must be of the form $a^i b^j a^k$ for $i, j, k \geq 0$.

2. (a) The grammar G generates the language

$$L = \{w\$w^R \mid w \in \{a, b\}^*\}.$$

   First we will show that $L \subseteq L(G)$. Let $w \in L$. We will show this by induction on the length of w. The base case is $|w| = 1$, since $\varepsilon \notin L$. The only word of length 1 is \$. Thus, we have $w = \$$. There is a production $S \to \$$ in G, so $S \Rightarrow w$ and therefore $w \in L(G)$.

   For the inductive step, we consider $|w| \geq 2$ and thus, $w \neq \$$. For our induction hypothesis, we suppose that for any $w' \in L$ with $|w'| < |w|$, we have $w' \in L(G)$. In other words, $S \Rightarrow^* w'$.

Since $w \in L$ and $|w| > 1$, we have $w = x\$x^R$ for some $x \in \{a, b\}^*$. Let $\sigma \in \{a, b\}$ be the first symbol of x (and therefore, the last symbol of x^R) and write $x = \sigma y$ for $y \in \{a, b\}^*$. Then we can write $w = \sigma y\$y^R\sigma$. Clearly, $y\$y^R \in L$ and is shorter than w. By our inductive hypothesis, there is a derivation $S \Rightarrow^* y\$y^R$. Then we have the following derivation for w:

$$S \Rightarrow \sigma S\sigma \Rightarrow^* \sigma y\$y^R\sigma = w,$$

since for any choice of σ, there exists a production $S \to \sigma S\sigma$ in G. Thus, we have $w \in L(G)$ and therefore $L \subseteq L(G)$ as desired.

Now, we show $L(G) \subseteq L$. Let $w \in L(G)$. We will show that $w \in L$ by induction on k, the nubmer of steps in the derivation of w. For our base case, we have $k = 1$. Since there is only one word with a derivation of length 1, we have $w = \$$ and therefore $w \in L$.

Now, consider $k \geq 2$ and for our inductive hypothesis, we assume that every word $w' \in L(G)$ with fewer than k steps in its derivation is in L. Since $k \geq 2$, the first step in the derivation of w must be $S \to \sigma S\sigma$, where $\sigma \in \{a, b\}$. Let $w' \in L(G)$ be a word that has a derivation with fewer than k steps and let $w = \sigma w'\sigma$. By our inductive hypothesis, $w' \in L$ so we can write $w' = x\$x^R$ for some $x \in \{a, b\}^*$. Then we have $w = \sigma w'\sigma = \sigma x\$x^R\sigma$ and therefore $w \in L$ for any choice of σ. Thus, $L(G) \subseteq L$ as desired.

We have shown that $L = L(G)$.

(b) We will follow the steps of the algorithm.

i. First, add a new start symbol.

$$\begin{aligned} S_0 &\to S \\ S &\to aSa \mid bSb \mid \$ \end{aligned}$$

ii. The next step is to eliminate any ε-productions. There are none.

iii. Then, we eliminate unit productions.

$$\begin{aligned} S_0 &\to aSa \mid bSb \mid \$ \\ S &\to aSa \mid bSb \mid \$ \end{aligned}$$

iv. Next, we assign each terminal a variable unless it appears on its own on the right hand side of a production rule.

$$\begin{aligned} S_0 &\to ASA \mid BSB \mid \$ \\ S &\to ASA \mid BSB \mid \$ \\ A &\to a \\ B &\to b \end{aligned}$$

v. Finally, we split up each rule.

$$\begin{aligned} S_0 &\to AA_1 \mid BB_1 \mid \$ \\ S &\to AA_1 \mid BB_1 \mid \$ \\ A_1 &\to SA \\ B_1 &\to SB \\ A &\to a \\ B &\to b \end{aligned}$$

3. (a) The PDA A accepts the language

$$L = \{a^i b^j c^k \mid i + k = j; i, j, k \geq 0\}.$$

To see this, observe that in q_0, for each a that is read, an X is placed on the stack. To read b's, the machine must move to state q_1 and machine discards an X on the stack for every b that is read. Once there are no more X's on the stack, the machine pushes a Y onto the stack for every b that is read henceforth. Once the machine is prepared to read c's, it moves to state q_2 and reads exactly as many c's as there are Y's on the stack.

Thus, the machine reads a sequence of a's followed by b's followed by c's. Furthermore, if it reads m a's, it will read at least m b's. If more b's are read, say n of them, the machine must then read n c's. Thus, we have read a word of the form $a^m b^{m+n} c^n$ as desired.

(b) We define the grammar G as follows

$$\begin{aligned} S &\to AB \\ A &\to aAb \mid \varepsilon \\ B &\to bBc \mid \varepsilon \end{aligned}$$

To see that G generates L defined above, we observe that A generates words of the form $a^i b^i$ for $i \geq 0$, while B generates words of the form $b^j c^j$ for $j \geq 0$. Since the first step of any derivation of G must be $S \to AB$, we have that S generates words of the form $a^i b^i b^j c^j = a^i b^{i+j} c^j$.

4. Since the stack of the PDA will ever only have at most three elements on it, there are only a finite number of configurations that the stack can be in. Therefore, we can simulate the action of the stack by only using a finite number of states. We will build an ε-NFA that does exactly this.

Let $P = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$ be the PDA. We will construct an ε-NFA $A = (Q', \Sigma, \delta', q_0', F')$ as follows:

- $Q' = Q \times \Gamma^{\leq 3}$; that is, a state of A is a state of P and a string of stack symbols with length at most 3.

- $q_0' = \langle q_0, Z_0 \rangle$; the initial state of A is the start state of P and Z_0 on the stack.
- $F' = F \times \Gamma^{\leq 3}$; A will accept if upon reading a word w, the machine is in a final state of P with any contents on the stack.
- δ' is defined as follows: for each $(q', \alpha) \in \delta(q, a, X)$, where $q, q' \in Q$, $a \in \Sigma \cup \{\varepsilon\}$, $X \in \Gamma$, and $\alpha \in \Gamma^{\leq 3}$, we have

  $$\delta'(\langle q, X\beta \rangle, a) = \langle q', \alpha\beta \rangle$$

  where $\beta \in \Gamma^{\leq 3}$. In this way, the NFA keeps track of both the current state of P and the current stack contents and the transition function of A mimics the action on the stack of P. Since the stack is guaranteed never to exceed three elements, this is possible.

Therefore, A is an ε-NFA that recognizes the language of P. Thus, $L(P)$ is regular.

5. Let $L = \{w \in \{a, b, c, d\}^* \mid |w|_a = |w|_d \wedge |w|_b = |w|_c\}$. Show that L is not context-free.

   Suppose that L is context-free and let $n > 0$ be the pumping length for L. Choose $w = a^n b^n d^n c^n$. We have $|w|_a = |w|_d = n = |w|_b = |w|_c$ and therefore $w \in L$ and $|w| = 4n \geq n$ as required. Now consider factorizations of $w = uvxyz$ such that $|vxy| \leq n$ and $vy \neq \varepsilon$. There are two cases to consider.

   - $vxy = \sigma^s$, $\sigma \in \{a, b, c, d\}$ such that $0 < s \leq n$ and $vy \neq \varepsilon$. For any factorization that satisfies the above properties, we have $vy = \sigma^t$ for some $t > 0$. Then for any $i > 0$, we have $|uv^i xy^i z|_\sigma > |uv^i xy^i z|_{\overline{\sigma}}$, where $\overline{\sigma}$ is defined by

     $$\overline{a} = d \quad \overline{b} = c \quad \overline{c} = b \quad \overline{d} = a.$$

     Thus, $uv^i xy^i z \notin L$ in this case.

   - $|vxy| = a^s b^t$ such that $0 < s + t \leq n$ and $vy \neq \varepsilon$. Since $|vxy| \leq n$, vxy cannot contain any c's or d's. Therefore, choosing $i > 1$, we have either $|uv^i xy^i z|_a > |uv^i xy^i z|_d$ or $|uv^i xy^i z|_b > |uv^i xy^i z|_c$ and thus $uv^i xy^i z \notin L$. By the same argument, we can choose $vxy = d^s c^t$ and arrive at the fact that $|uv^i xy^i z|_a < |uv^i xy^i z|_d$ or $|uv^i xy^i z|_b < |uv^i xy^i z|_c$. Similarly, choosing $vxy = b^s d^t$ and following the same argument gives us $|uv^i xy^i z|_a < |uv^i xy^i z|_d$ or $|uv^i xy^i z|_b > |uv^i xy^i z|_c$.

   These are the only possible factorizations, since $|vxy| \leq n$ makes it impossible for vxy to contain more than two different symbols. Therefore, every factorization of $w = uvxyz$ fails to satisfy the pumping lemma and therefore L is not context-free as assumed.