**Theory of Computation**

**CS-312** **Fall 2023**

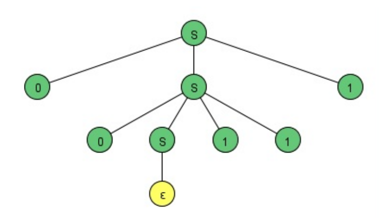
**Exam 2** **Solution Set**

---

1. *Consider the context-free grammar given by the following ruleset:*

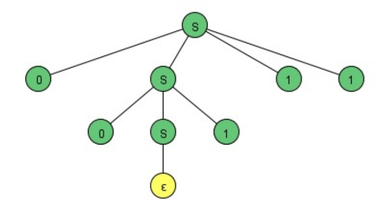
$$S \to 0S1 \mid 0S11 \mid \epsilon$$

   (a) *5 points. Give a parse tree showing that* 00111 *can be generated by this grammar.*




   (b) *5 points. Describe (in English) the language generated by this grammar.*
   $\{0^n1^m : n \leq m \leq 2n\}$

   (c) *5 points. Show that this grammar is ambiguous.*
   The following is a different parse tree for 00111; this, combined with the answer to part (a), gives two different parse trees for the same input string.




2. *10 points each. Give context-free grammars which generate the following languages:*

   (a) $\{0^n1^m : m \geq n, m - n \text{ is even}\}$.
   (Note: $m - n$ may be a negative even number or 0.)

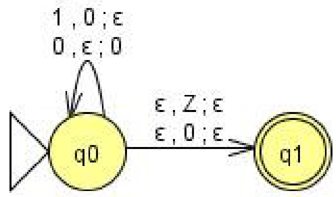
$$\begin{aligned} S &\to 0S1 \mid A \\ A &\to 00A \mid A11 \mid \epsilon \end{aligned}$$

   (b) $\{w \in \{0,1\}^* :\ w \text{ has odd length, and the first, middle, and last symbols are identical}\}$

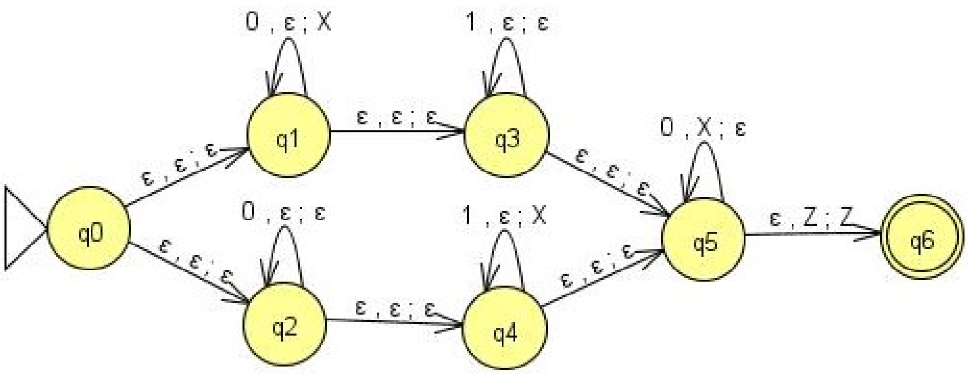
$$\begin{aligned} S &\to 0 \mid 1 \mid 0A0 \mid 1B1 \\ A &\to 0A0 \mid 0A1 \mid 1A0 \mid 1A1 \mid 0 \\ B &\to 0B0 \mid 0B1 \mid 1B0 \mid 1B1 \mid 1 \end{aligned}$$

3. *10 points each. Give pushdown automata which accept the following languages (by final state):*

   (a) $\{w \in \{0,1\}^* :$ every prefix of $w$ has at least as many 0s as 1s$\}$




   (b) $\{0^i 1^j 0^k : i = k \text{ or } j = k\}$




4. *15 points. Show that the class of context-free languages is closed under the reversal operator. That is: for any string $w \in \Sigma^*$, let $w^R \in \Sigma^*$ be the string obtained by reversing the order of the letters in $w$. (For example, $(0010)^R = 0100$. For a language $L \subseteq \Sigma^*$, let $L^R = \{w^R : w \in L\}$. Suppose that $L$ is a context-free language. Show that $L^R$ is a context-free language.*

   If $L$ is a context-free language, $L$ has a context-free grammar $G = (V, \Sigma, S, P)$ which generates $L$. Construct a new grammar $G' = (V, \Sigma, S, P')$, where every rule in $P'$ is a reversal of a rule in $P$. That is, if $A \to \alpha$ is a rule in $P$, $A \to \alpha^R$ is a rule in $P'$. This grammar generates $L^R$.

5. *15 points. jFLAP supports two different models of pushdown automata. The model we used throughout the course ("multiple character input") allows automata to push multiple characters at once onto a stack during a push operation. The other common model ("single character input") allows at most one character to be pushed at a time onto the stack.*

   *Show that these computational models of pushdown automata are equivalent. That is, show that any language accepted by an automata of one type can be accepted by an automata of the other type.*

   *(Hint: this is two proofs. One is easier than the other.)*

   Every single-character-input pushdown automata is already a multiple-character-input pushdown automata; it simply does not take advantage of the added features. Thus, any language accepted by a single-character-input pushdown automata is triviallly accepted by a multiple-character-input pushdown automata.

For the other direction, we have some work to do. Let $\delta(p, a, b_0) = (q, b_1 b_2 \dots b_k)$ be a transition in a multiple-character-input pushdown automaton. That is, on state $p$, reading input $a$ and popping stack symbol $b_0$, we move to state $q$ and push the symbols $b_1 \dots b_k$ onto the stack.

Modify the pushdown automaton as follows. Create new states $s_1, s_2, \dots, s_{k-1}$, remove the old transition $\delta(p, a, b_0) = (q, b_1 b_2 \dots b_k)$, and add the following transitions:

$$\begin{aligned}
&\delta(p, a, b_0) = (s_{k-1}, b_k)\\
&\delta(s_{k-1}, \epsilon, \epsilon) = (s_{k-2}, b_{k-1})\\
&\delta(s_{k-2}, \epsilon, \epsilon) = (s_{k-3}, b_{k-2})\\
&\vdots\\
&\delta(s_2, \epsilon, \epsilon) = (s_1, b_2)\\
&\delta(s_1, \epsilon, \epsilon) = (q, b_1)
\end{aligned}$$

That is, the new machine will take $k$ steps to push the characters $b_k, b_{k-1}, \dots, b_2, b_1$ onto the stack individually.

Performing this transformation on all transitions of the original automata will create an equivalent single-character-input pushdown automaton.

6. *15 points. Consider the language $L = \{0^a 1^b 0^c : a > b > c\}$. Prove that this language $L$ is not context-free.*

Suppose that $L$ is not context-free. By the Pumping Lemma, there is a constant $n$ such that for any string $z \in L$ with $|z| \geq n$, there are substrings $u, v, w, x, y$ such that $z = uvwxy$, $|vwx| \leq n$, $vx \neq \epsilon$, and $uv^i wx^i y \in L$ for any $i \geq 0$.

Consider the string $z = 0^{n+2} 1^{n+1} 0^n$, where $n$ is the constant from the Pumping Lemma. (That is, $a = n+2$, $b = n+1$, and $c = n$.) There are several cases for $v$ and $x$ to consider.

- $v$ or $x$ contain more than one symbol. Then $uv^2wx^2y$ does not have the form $0^*1^*0^*$, and therefore is not in the language.
- $vx$ contains only 0s from the first block of 0s. Then $uv^0wx^0y = 0^{n+2-k}1^{n+1}0^n$ for some $k > 0$. Since $a = n+2-k \leq n+1 = b$, this means $a \not> b$, and $z \notin L$.
- $vx$ contains only 1s. Then $uv^2wx^2y = 0^{n+2}1^{n+1+k}0^n$ for some $k > 0$. Since $a = n+2 \leq n+1+k = b$, this means $a \not> b$, and $z \notin L$.
- $vx$ contains only 0s from the last block of 0s. Then $uv^2wx^2y = 0^{n+2}1^{n+1}0^{n+k}$ for some $k > 0$. Since $b = n+1 \leq n+k = c$, this means $b \not> c$, and $z \notin L$.
- $vx$ contains 0s from the first block of 0s and 1s. Then $uv^0wx^0y = 0^{n+2-j}1^{n+1-k}0^n$ for some $j, k > 0$. Since $b = n+1-k \leq n = c$, this means $b \not> c$, and $z \notin L$.
- $vx$ contains 1s and 0s from the last block of 0s. Then $uv^2wx^2y = 0^{n+2}1^{n+1+j}0^{n+k}$ for some $j, k > 0$. Since $a = n+2 \leq n+1+j = b$, this means $a \not> b$, and $z \notin L$.

In all cases, we can pump $z$ to create strings not in $L$. This is our desired contradiction.