CS2200: Languages, Machines, and Computation
Class test 3
April 16, 2024

Max Marks: 15
Duration: 1 hour

1. Consider the following context-free grammar G.

$$S \to aB \mid bA \mid \varepsilon$$
$$A \to aS \mid bAA$$
$$B \to bS \mid aBB$$

(a) **[2 marks]** Show that G is ambiguous.

Solution: Give a string w that has at least two distinct leftmost derivations.

Consider the string $aababb$. It has the following two distinct leftmost derivations.

$$S \to aB \to aaBB \to aabSB \to aabaBB \to aababSB \to aababB \to aababbS \to aababb$$
$$S \to aB \to aaBB \to aabSB \to aabB \to aabaBB \to aababSB \to aababB \to aababbS \to aababb$$

(b) **[1 mark]** Does there exist a string $w \in L(G)$ that has a unique parse tree? Justify your answer.

Solution: $S \to \varepsilon$ has a single leftmost derivation, and hence a unique parse tree.

2. **[3 marks]** A string x is said to be a *subsequence* of y if we can obtain x from y by deleting symbols in y and keeping the remaining in the same order. For instance, 123 is a subsequence of 91442763 since we can delete 9, the two 4s, 7 and 6 to obtain 123.

Construct a CFG for the language given below.

$$L = \{w\#x \mid w^R \text{ is a subsequence of } x \text{ where } w, x \in \{0,1\}^*\}.$$

Solution:

This is similar to the question in the problem set for $w\#x$ where w^R is a substring of x

Consider the following grammar.

$$S \to AB \mid \#B$$
$$A \to 0S0 \mid 1S1$$
$$B \to 0B \mid 1B \mid \varepsilon$$

Firstly, the strings generated by the non-terminal B is precisely $\{0,1\}^*$.

Now, any string of the form $\#x$ must be accepted, and $S \longrightarrow \#B$ generates such strings. For a string of the form $w\#x$ where $w \neq \varepsilon$, and w^R is a subsequence of x, we can say that it is of the form $\sigma u\#v\sigma z$ where $\sigma \in \{0,1\}$ and u^R is a subsequence of v - this follows from the definition of a subsequence. Thus, $S \longrightarrow AB \longrightarrow \sigma S\sigma B \xrightarrow{\star} \sigma u\#v\sigma B \xrightarrow{\star} \sigma u\#v\sigma z$.

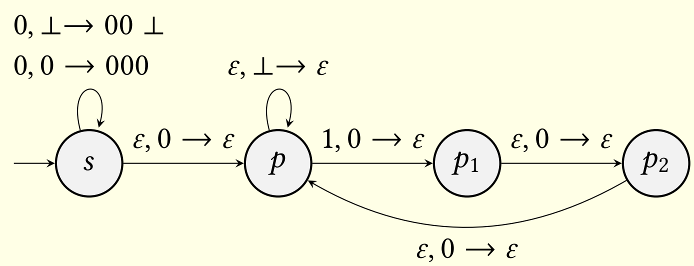
Now, if $S \longrightarrow \#B$ is the first step of a derivation, then the string generated is in the language. If $S \longrightarrow AB$ is the first step of the derivation, if we take the rightmost derivation, we have $S \longrightarrow AB \xrightarrow{\star} Az$ where $z \in \{0,1\}^*$. Looking at the production rules for A, we have $S \longrightarrow AB \xrightarrow{\star} Az \longrightarrow \sigma S\sigma z$. Thus, we can see that the string that is generated is in L (technically a proof on the length of the derivation).

3. **[3 marks]** Construct a PDA with stack alphabet $\Gamma = \{0, \perp\}$ for the language

$$L = \{0^i 1^j \mid 2i = 3j + 1 \text{ where } i, j \geq 0\}.$$

Draw the state transition diagram as discussed in class. Briefly describe why your construction is correct.

Solution: For each 0 that is read, push two 0s on to the stack. Now, non-deterministically guess that 0s have finished, and pop one 0 from the stack. Now, for each 1 that you see, pop three 0s from the stack. If the stack is empty after seeing all the 1s accept, else reject. Here the acceptance is via the empty stack.



4. **[6 marks]** Consider the following two languages.

$$L_1 = \{a^\ell b^m c^n \mid \ell < m \text{ and } m < n\}$$
$$L_2 = \{a^\ell b^m c^n \mid \ell < m \text{ or } m < n\}$$

Which of the following statements about L_1 and L_2 is correct.

A. Both L_1 and L_2 are context-free

B. L_1 is context-free and L_2 is not context-free

C. L_1 is not context-free and L_2 is context-free

D. Both L_1 and L_2 are not context-free

Solution: Option (C) is correct.

$L_2 = \{a^\ell b^m c^n \mid \ell < m\} \cup \{a^\ell b^m c^n \mid m < n\}$. The grammar for the first language is as follows; the grammar for the second is also similar.

$$
\begin{aligned}
S &\longrightarrow AB \\
A &\longrightarrow aAb \mid \varepsilon \\
B &\longrightarrow bCD \\
C &\longrightarrow bC \mid \varepsilon \\
D &\longrightarrow cD \mid \varepsilon
\end{aligned}
$$

To prove that L_1 is not context-free, choose $z = a^k b^{k+1} c^{k+2}$. Now for any choice of the prover, one of the following must happen.

- v or w contain two different symbols: Choose $i > 1$ and the string will no longer be in the form $a^\ell b^m c^n$ and hence not in L_1.
- w contains only b: Then v contains only a or only b. Choose $i > 1$, and we have $n < m$.
- w contains only c: Then v contains only c, or v contains only b. In both cases choose $i = 0$ - either $\ell > m$ or $m > n$.
- w contains only a: Then v contains only a, and hence choose $i > 1$ to have $\ell > m$.