COMP 3803 — Solutions Assignment 3

Question 1: Give context-free grammars that generate the following languages. For each case, justify your answer.
(1.1) $\{0^{2n}1^n | n \geq 0\}$. The set Σ of terminals is equal to $\{0, 1\}$.
(1.2) $\{w | w$ starts and ends with different symbols$\}$. The set Σ of terminals is equal to $\{0, 1\}$.
(1.3) $\{w |$ w is a palindrome$\}$. The set Σ of terminals is equal to $\{0, 1\}$.
A *palindrome* is a string w having the property that $w = w^R$, i.e., reading w from left to right gives the same result as reading w from right to left. For example, the four binary strings ϵ, 1, 0110, and 10101 are palindromes.
(1.4) $\{a^m b^n c^n | m \geq 0, n \geq 0\}$. The set Σ of terminals is equal to $\{a, b, c\}$.

Solution: First, we do

$$L_1 = \{0^{2n}1^n | n \geq 0\}.$$

Any string in L_1 is either

- empty or
- starts with 00, followed by an arbitrary string in L_1, and ends with 1.

This leads to the context-free grammar $G = (V, \Sigma, R, S)$, where $V = \{S\}$, $\Sigma = \{0, 1\}$, and R consists of the two rules

$$S \to \epsilon | 00S1.$$

Next, we do

$$L_2 = \{w | w \text{ starts and ends with different symbols}\}.$$

I should have made it clear whether or not the empty string ϵ is contained in L_2.

First we consider the case when ϵ is not included in L_2. Any string in L_2 either

- starts with 0, is followed by an arbitrary binary string, and ends with 1, or
- starts with 1, is followed by an arbitrary binary string, and ends with 0.

This leads to the context-free grammar $G = (V, \Sigma, R, S)$, where $V = \{S, X\}$, $\Sigma = \{0, 1\}$, and R consists of the rules

$$S \to 0X1 | 1X0$$

$$X \to \epsilon | 0X | 1X$$

Observe that from X, we can derive all binary strings (including the empty string). Thus, from S, we can derive all strings in L_2 (and nothing else).

If we assume that ϵ is contained in L_2, we get the rules

$$S \to \epsilon|0X1|1X0$$

$$X \to \epsilon|0X|1X$$

Next, we do

$$L_3 = \{w|\ w \text{ is a palindrome}\}.$$

Any string in L_3 either

- is empty,
- is equal to 0,
- is equal to 1,
- starts with 0, followed by a palindrome, and ends with 0, or
- starts with 1, followed by a palindrome, and ends with 1.

This leads to the context-free grammar $G = (V, \Sigma, R, S)$, where $V = \{S\}$, $\Sigma = \{0, 1\}$, and R consists of the rules

$$S \to \epsilon|0|1|0S0|1S1$$

Finally, we do

$$L_4 = \{a^m b^n c^n | m \geq 0, n \geq 0\}.$$

Any string in L_4

- starts with 0 or more a's, followed by a string of the form $b^n c^n$, for some $n \geq 0$.

This leads to the context-free grammar $G = (V, \Sigma, R, S)$, where $V = \{S, X\}$, $\Sigma = \{a, b, c\}$, and R consists of the rules

$$S \to AX$$

$$A \to \epsilon|aA$$

$$X \to \epsilon|bXc$$

Observe that from A, we can derive all strings of the form a^m for some $m \geq 0$. From X, we can derive all strings of the form $b^n c^n$, for some $n \geq 0$. Therefore, from S, we can derive all strings in L_4 (and nothing else).

Question 2: Let L and L' be context-free languages over the same alphabet Σ.
(2.1) Prove that the union $L \cup L'$ of L and L' is also context-free.
(2.2) Prove that the concatenation LL' of L and L' is also context-free.
(2.3) Prove that the star L^* of L is also context-free.

Solution: Since L is context-free, there is a context-free grammar $G_1 = (V_1, \Sigma, R_1, S_1)$ that generates L. Similarly, since L' is context-free, there is a context-free grammar $G_2 = (V_2, \Sigma, R_2, S_2)$ that generates L'. We assume that $V_1 \cap V_2 = \emptyset$. (If this is not the case, then we rename the variables of G_2.)

First, we show that $L \cup L'$ is context-free. Let $G = (V, \Sigma, R, S)$ be the context-free grammar, where

- $V = V_1 \cup V_2 \cup \{S\}$, where S is a new variable, which is the start variable of G,
- $R = R_1 \cup R_2 \cup \{S \to S_1 | S_2\}$.

From the start variable S, we can derive the strings S_1 and S_2. From S_1, we can derive all strings of L, whereas from S_2, we can derive all strings of L'. Hence, from S, we can derive all strings of $L \cup L'$. In other words, the grammar G generates the union of L and L'. Therefore, this union is context-free.

Next, we show that LL' is context-free. Let $G = (V, \Sigma, R, S)$ be the context-free grammar, where

- $V = V_1 \cup V_2 \cup \{S\}$, where S is a new variable, which is the start variable of G,
- $R = R_1 \cup R_2 \cup \{S \to S_1 S_2\}$.

From the start variable S, we can derive the string $S_1 S_2$. From S_1, we can derive all strings of L, whereas from S_2, we can derive all strings of L'. Hence, from S, we can derive all strings of the form uv, where $u \in L$ and $v \in L'$. In other words, the grammar G generates the concatenation of L and L'. Therefore, this concatenation is context-free.

Finally, we show that L^* is context-free. Any string in L^* is either

- empty or
- a string in L, followed by a string in L^*.

Let $G = (V, \Sigma, R, S)$ be the context-free grammar, where

- $V = V_1 \cup \{S\}$, where S is a new variable, which is the start variable of G,
- $R = R_1 \cup \{S \to \epsilon | S_1 S\}$.

From the start variable S, we can derive all strings S_1^n, where $n \geq 0$. From S_1, we can derive all strings of L. Hence, from S, we can derive all strings of the form $u_1 u_2 \ldots u_n$, where $n \geq 0$, and each string u_i $(1 \leq i \leq n)$ is in L. In other words, the grammar G generates the star of L. Therefore, L^* is context-free.

By the way, the context-free grammar $G = (V_1, \Sigma, R, S_1)$, where

$$R = R_1 \cup \{S_1 \to \epsilon | S_1 S_1\}$$

may *not* generate L^*. Here is an example:

The context-free grammar $G_1 = (V_1, \Sigma, R_1, S_1)$, where $V_1 = \{S_1\}$, $\Sigma = \{0, 1\}$, and $R_1 = \{S_1 \to 0S_1 0|1\}$ generates the language

$$L = \{0^n 10^n : n \geq 0\}.$$

From the grammar G above, we can obtain the string 00:

$$S_1 \Rightarrow 0S_1 0 \Rightarrow 00.$$

However, this string 00 is not in L^*.

Question 3: Give (deterministic or nondeterministic) pushdown automata that accept the following languages. For each pushdown automaton, start by explaining the algorithm in plain English, then mention the states that you are going to use, then explain the meaning of these states, and finally give the list of instructions.
(3.1) $\{0^{2n}1^n | n \geq 0\}$.
(3.2) $\{ww^R | w \in \{0, 1\}^*\}$.
(If $w = w_1 \ldots w_n$, then $w^R = w_n \ldots w_1$.)

Solution: For the language

$$\{0^{2n}1^n | n \geq 0\}$$

we can use a deterministic pushdown automaton. The approach is as follows:

- Walk along the input string from left to right.
- While reading 0's: For each 0, push a symbol S onto the stack.
- While reading 1's: For each 1, pop the top two symbols from the stack. Popping two symbols is done in two steps: Pop the top symbol and do not move on the input tape; then again pop the (new) top symbol and make one step to the right on the input tape.
- Tape alphabet $\Sigma = \{0, 1\}$.
- Stack alphabet $\Gamma = \{\$, S\}$.

We use three states:

- q_0: This is the start state. If we are in this state, then we are reading the block of 0's. The stack contains \$ at the bottom; the number of S-symbols on the stack is equal to the number of 0's read so far.
- q_1: We have already seen a 1. If the current symbol on the input tape is a 1, then we are going to do the first pop.
- q_1': The current symbol on the input tape is a 1; we are going to do the second pop.

The instructions are as follows.

- $q_0 0\$ \to q_0 R\$S$ (push S onto the stack)
- $q_0 0S \to q_0 RSS$ (push S onto the stack)
- $q_0 1\$ \to q_0 N\$$
  - Explanation: The input string starts with 1; loop forever and, thus, do not accept.
- $q_0 1S \to q_1 NS$
  - Explanation: We have reached the first 1. We switch to state q_1, do not move on the input tape, and do not modify the stack.
- $q_0 \square\$ \to q_0 N\epsilon$
  - Explanation: The input string is empty. We make the stack empty and, thus terminate and accept.
- $q_0 \square S \to q_0 RSS$
  - Explanation: The input string is non-empty and only contains 0's; loop forever and, thus, do not accept.
- $q_1 0\$ \to q_1 N\$$
  - Explanation: There is a 0 to the right of a 1; loop forever and, thus, do not accept.
- $q_1 0S \to q_1 NS$
  - Explanation: There is a 0 to the right of a 1; loop forever and, thus, do not accept.
- $q_1 1\$ \to q_1 N\$$
  - Explanation: Too many 1's; loop forever and, thus, do not accept.
- $q_1 1S \to q_1' N\epsilon$
  - Explanation: The first pop for the current 1.

- $q_1 \square \$ \rightarrow q_1 N \epsilon$
  - Explanation: Make the stack empty, terminate, and accept.
- $q_1 \square S \rightarrow q_1 N S$
  - Explanation: Too many 0's; loop forever and, thus, do not accept.
- $q_1' 0 \$ \rightarrow$ cannot happen
- $q_1' 0 S \rightarrow$ cannot happen
- $q_1' 1 \$ \rightarrow q_1' N \$$
  - Explanation: Too many 1's; loop forever and, thus, do not accept.
- $q_1' 1 S \rightarrow q_1 R \epsilon$
  - Explanation: The second pop for the current 1.
- $q_1' \square \$ \rightarrow$ cannot happen
- $q_1' \square S \rightarrow$ cannot happen

For the language

$$\{ww^R | w \in \{0,1\}^*\}$$

we use a nondeterministic pushdown automaton. The approach is as follows:

- Walk along the input string from left to right.
- Guess when we enter the second half of the input string.
- All symbols in the first half of the input string are pushed onto the stack.
- After we have entered the second half, we check if the contents of the stack is the same as the remaining part of the input string.
- Tape alphabet $\Sigma = \{0, 1\}$.
- Stack alphabet $\Gamma = \{\$, 0, 1\}$.

We will use two states:

- q: This is the start state. If we are in this state, then we have not guessed yet that we have entered the second half of the input string.
- q': We have guessed already that we have entered the second half of the input string.

The instructions are as follows.

- $q0\$ \to qR\$0$ (push; stay in start state)
- $q0\$ \to q'R\$0$ (push, switch to q')
- $q1\$ \to qR\$1$ (push; stay in start state)
- $q1\$ \to q'R\$1$ (push, switch to q')
- $q\Box\$ \to qN\epsilon$ (input empty; accept)
- $q00 \to qR00$ (push; stay in start state)
- $q00 \to q'R00$ (push, switch to q')
- $q10 \to qR01$ (push; stay in start state)
- $q10 \to q'R01$ (push, switch to q')
- $q\Box 0 \to qN0$ (loop forever)
- $q01 \to qR10$ (push; stay in start state)
- $q01 \to q'R10$ (push, switch to q')
- $q11 \to qR11$ (push; stay in start state)
- $q11 \to q'R11$ (push, switch to q')
- $q\Box 1 \to qN1$ (loop forever)
- $q'0\$ \to q'N\$$ (loop forever)
- $q'1\$ \to q'N\$$ (loop forever)
- $q'\Box\$ \to q'N\epsilon$ (accept)
- $q'00 \to q'R\epsilon$ (pop)
- $q'10 \to q'N0$ (loop forever)
- $q'\Box 0 \to q'N0$ (loop forever)
- $q'01 \to q'N1$ (loop forever)
- $q'11 \to q'R\epsilon$ (pop)
- $q'\Box 1 \to q'N1$ (loop forever)

Question 4: Prove that the following languages are not context-free:
(4.1) $\{0^n\,1\,0^{2n}\,1\,0^{3n} | n \geq 0\}$.
(4.2)

$$\{\ w \in \{a,b,c\}^* \quad | \quad \begin{array}{l} w \text{ contains more } b\text{'s than } a\text{'s } \textbf{and} \\ w \text{ contains more } c\text{'s than } a\text{'s } \}. \end{array}$$

Solution: First, we prove that the language

$$A = \{0^n\,1\,0^{2n}\,1\,0^{3n} | n \geq 0\}$$

is not context-free.

Assume that A is context-free. By the pumping lemma, there is an integer $p \geq 1$, such that for all strings $s \in A$ with $|s| \geq p$, the following holds: We can write $s = uvxyz$, where

1. vy is non-empty,
2. vxy has length at most p,
3. the string uv^ixy^iz is in A, for all $i \geq 0$.

Consider the pumping length p. We choose $s = 0^p\,1\,0^{2p}\,1\,0^{3p}$. Then s is a string in A, and the length of s is $6p + 2$, which is at least p. Thus, we can write $s = uvxyz$ such that 1., 2., and 3. above hold.

Case 1: v or y contains a 1.

The string $s' = uvvxyyz$ contains at least three 1's; thus, this string is not in A. But, by the pumping lemma, s' is contained in A. This is a contradiction.

Case 2: v or y overlaps the leftmost block of 0's.

Since $|vxy| \leq p$, the string y does not overlap the rightmost block of 0's. Thus, since $|vy| \geq 1$, the string $s' = uvvxyyz$ starts with more than p many 0's, and ends with exactly $3p$ many 0's. Therefore, this string is not in A. But, by the pumping lemma, s' is contained in A. This is a contradiction.

Case 3: Neither v nor y contains a 1, and neither v nor y overlaps the leftmost block of 0's.

The string $s' = uvvxyyz$ starts with exactly p many 0's. The middle block of 0's has length more than $2p$ or the rightmost block of 0's has length more than $3p$. Therefore, this string is not in A. But, by the pumping lemma, s' is contained in A. This is a contradiction.

Next we prove that the language

$$B = \{\ w \in \{a,b,c\}^* \quad | \quad \begin{array}{l} w \text{ contains more } b\text{'s than } a\text{'s } \textbf{and} \\ w \text{ contains more } c\text{'s than } a\text{'s } \} \end{array}$$

is not context-free.

Assume that B is context-free. By the pumping lemma, there is an integer $p \geq 1$, such that for all strings $s \in B$ with $|s| \geq p$, the following holds: We can write $s = uvxyz$, where

1. vy is non-empty,

2. vxy has length at most p,

3. the string uv^ixy^iz is in B, for all $i \geq 0$.

Consider the pumping length p. We choose $s = a^p b^{p+1} c^{p+1}$. Then s is a string in B, and the length of s is $3p + 2$, which is at least p. Thus, we can write $s = uvxyz$ such that 1., 2., and 3. above hold.

Case 1: v or y overlaps the block of a's.

Since $|vxy| \leq p$, the string y does not overlap the block of c's. Thus, since $|vy| \geq 1$, the string $s' = uvvxyyz$ contains at least $p + 1$ many a's, and exactly $p + 1$ many c's. Therefore, this string is not in B. But, by the pumping lemma, s' is contained in B. This is a contradiction.

Case 2: Neither v nor y overlaps the block of a's.

Consider the string $s' = uxz$. Since $|vy| \geq 1$, this string contains at most p many b's or at most p many c's. On the other hand, it contains exactly p many a's. Therefore, this string is not in B. But, by the pumping lemma, s' is contained in B. This is a contradiction.

Question 5: In Question 1, you have shown that

$$L = \{a^m b^n c^n | m \geq 0, n \geq 0\}$$

is a context-free language. By a symmetric argument, the language

$$L' = \{a^m b^m c^n | m \geq 0, n \geq 0\}$$

is context-free.
(5.1) Argue that the intersection of two context-free languages is not necessarily context-free. (You may use any result that was proven in class.)
(5.2) Argue that the complement of a context-free language is not necessarily context-free.

Solution: For the first part:

- L is context-free.
- L' is context-free.
- Let $L'' = L \cap L'$.
- $L'' = \{a^n b^n c^n | n \geq 0\}$.
- We have seen in class that L'' is not context-free.

The second part is proved by contradiction. Thus, we assume that for any context-free language A, the complement $\overline{A}$ is also context-free. Under this assumption, we will show that the intersection of any two context-free languages is also context-free. This will contradict the first part of the question.

Let A and B be two arbitrary context-free languages.

- By our assumption, both $\overline{A}$ and $\overline{B}$ are context-free.
- From Question 2: $\overline{A} \cup \overline{B}$ is context-free.
- By our assumption, $\overline{\overline{A} \cup \overline{B}}$ is context-free.
- By De Morgan, the latter language is equal to $A \cap B$.