COMP 3803 — Fall 2025 — Solutions Problem Set 4

Question 1: 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.
(1.1) $\{a^{2n}b^n : n \geq 0\}$.
(1.2) $\{ww^R : w \in \{a,b\}^*\}$.
(If $w = w_1 \dots w_n$, then $w^R = w_n \dots w_1$. Note that $\varepsilon^R = \varepsilon$.)
(1.3) $\{a^n : n \geq 0\} \cup \{a^n b^n : n \geq 0\}$.
(1.4) $\{a^m b^n : m \geq 0, n \geq 0, m \neq n\}$.

Solution: The solutions for parts 3 and 4 are in a different file, because I cannot find the original LaTeXfile.

For the language

$$\{a^{2n}b^n : n \geq 0\}$$

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

- Walk along the input string from left to right.
- While reading a's: For each a, push a symbol S onto the stack.
- While reading b's: For each b, 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 $\Sigma = \{a, b\}$.
- Stack alphabet $\Gamma = \{\$, S\}$.

We use three states:

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

The instructions are as follows.

- $q_a a\$ \to q_a R\$S$ (push S onto the stack)
- $q_a aS \to q_a RSS$ (push S onto the stack)

- $q_a b\$ \to q_a N\$$
  - Explanation: The input string starts with 1; loop forever and, thus, do not accept.
- $q_a bS \to q_b NS$
  - Explanation: We have reached the first b. We switch to state q_b, do not move on the input tape, and do not modify the stack.
- $q_a \Box\$ \to q_a N\varepsilon$
  - Explanation: The input string is empty. We make the stack empty and, thus terminate and accept.
- $q_a \Box S \to q_a NS$
  - Explanation: The input string is non-empty and only contains a's; loop forever and, thus, do not accept.
- $q_b a\$ \to q_b N\$$
  - Explanation: There is a to the right of b; loop forever and, thus, do not accept.
- $q_b aS \to q_b NS$
  - Explanation: There is a to the right of b; loop forever and, thus, do not accept.
- $q_b b\$ \to q_b N\$$
  - Explanation: Too many b's; loop forever and, thus, do not accept.
- $q_b bS \to q_b' N\varepsilon$
  - Explanation: The first pop for the current b.
- $q_b \Box\$ \to q_b N\varepsilon$
  - Explanation: Make the stack empty, terminate, and accept.
- $q_b \Box S \to q_b NS$
  - Explanation: Too many a's; loop forever and, thus, do not accept.
- $q_b' a\$ \to$ cannot happen
- $q_b' aS \to$ cannot happen
- $q_b' b\$ \to q_b' N\$$
  - Explanation: Too many b's; loop forever and, thus, do not accept.

- $q'_b bS \rightarrow q_b R\varepsilon$
  - Explanation: The second pop for the current b.
- $q'_b\Box\$ \rightarrow$ cannot happen
- $q'_b\Box S \rightarrow$ cannot happen

For the language
$$\{ww^R : w \in \{a,b\}^*\}$$
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 (from top to bottom) is the same as the remaining part of the input string.
- Tape alphabet $\Sigma = \{a,b\}$.
- Stack alphabet $\Gamma = \{\$, a, b\}$.

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. The instructions are as follows.

- $qa\$ \rightarrow qR\$a$ (push; stay in start state)
- $qa\$ \rightarrow q'R\$a$ (push, switch to q')
- $qb\$ \rightarrow qR\$b$ (push; stay in start state)
- $qb\$ \rightarrow q'R\$b$ (push, switch to q')
- $q\Box\$ \rightarrow qN\varepsilon$ (input empty; accept)
- $qaa \rightarrow qRaa$ (push; stay in start state)
- $qaa \rightarrow q'Raa$ (push, switch to q')
- $qba \rightarrow qRab$ (push; stay in start state)

- $qba \to q'Rab$ (push, switch to q')
- $q\Box a \to qNa$ (loop forever)
- $qab \to qRba$ (push; stay in start state)
- $qab \to q'Rba$ (push, switch to q')
- $qbb \to qRbb$ (push; stay in start state)
- $qbb \to q'Rbb$ (push, switch to q')
- $q\Box b \to qNb$ (loop forever)
- $q'a\$ \to q'N\$$ (loop forever)
- $q'b\$ \to q'N\$$ (loop forever)
- $q'\Box\$ \to q'N\varepsilon$ (accept)
- $q'aa \to q'R\varepsilon$ (pop)
- $q'ba \to q'Na$ (loop forever)
- $q'\Box a \to q'Na$ (loop forever)
- $q'ab \to q'Nb$ (loop forever)
- $q'bb \to q'R\varepsilon$ (pop)
- $q'\Box b \to q'Nb$ (loop forever)

Question 2: Prove that the following languages are not context-free:
(2.1) $\{a^{n!} : n \geq 0\}$.
(2.2) $\{a^{n^2}b^n : n \geq 0\}$.
(2.3) $\{a^m b^n a^m b^n : m \geq 0, n \geq 0\}$.
(2.4) $\{a^m b^n c^k : 1 \leq m < n < k < 2m\}$.

Solution: The solutions for parts 3 and 4 are in a different file, because I cannot find the original LaTeXfile.

First, we prove that the language

$$A = \{a^{n!} : n \geq 0\}$$

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 $i \geq 0$.

Note: We may assume that $p \geq 2$: If, for this particular language, the pumping length is equal to one, then the statement of the Pumping Lemma is also true if we take $p = 2$.

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

Let k be the length of the string vy. It follows from 1. that $k \geq 1$. It follows from 2. that $k = |vy| \leq |vxy| \leq p$.

Consider the string $uvvxyyz$. This string is equal to $a^{p!+k}$. We have (using the fact that $p \geq 2$)

$$p! < p! + k \leq p! + p < p \cdot p! + p! = (p + 1)!.$$

Thus, the length of $uvvxyyz$ is strictly between two consecutive factorials. Therefore, this string is no in the language A. This is a contradiction, because by the Pumping Lemma, this string does belong to A. We conclude that A is not a context-free language.

Next we prove that the language

$$B = \{a^{n^2}b^n : n \geq 0\}$$

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

Case 1: Both v and y are in the block of a's.

Then the string $s' = uvvxyyz$ contains at least $p^2 + 1$ many a's and exactly p many b's. Therefore, this string is not in B. But, by the Pumping Lemma, s' is contained in B. This is a contradiction.

Case 2: Both v and y are in the block of b's.

Then the string $s' = uvvxyyz$ contains exactly p^2 many a's and at least $p + 1$ many b's. Therefore, this string is not in B. But, by the Pumping Lemma, s' is contained in B. This is a contradiction.

Case 3: The string vy contains at least one a and at least one b.

Let k and ℓ be such that $vy = a^k b^\ell$. Then $k \geq 1$, $\ell \geq 1$, and $k + \ell \leq p$.

The string $s' = uvvxyyz$ is equal to

$$s' = a^{p^2+k} b^{p+\ell}.$$

By the Pumping Lemma, s' is in B, implying that

$$(p+\ell)^2 = p^2 + k,$$

i.e.,

$$k = 2p\ell + \ell^2.$$

However,

$$2p\ell + \ell^2 \geq 2p + 1 > p \geq k + \ell > k.$$

This is a contradiction.

Question 3: We have seen that the regular languages are closed under the union, intersection, complement, concatenation, and star operations. In this question, we consider these operations for context-free languages.

(3.1) Let L and L' be context-free languages over the same alphabet Σ. Prove that the union $L \cup L'$ is also context-free.

(3.2) Let L and L' be context-free languages over the same alphabet Σ. Prove that the concatenation LL' is also context-free.

(3.3) Let L be a context-free language over the alphabet Σ. Prove that the star L^* of L is also context-free.

(3.4) Prove that both

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

and

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

are context-free languages.

(3.5) Prove that the intersection of two context-free languages is not necessarily context-free.

(3.6) Prove that the complement of a context-free language is not necessarily context-free.

Solution: For the first three parts, 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.

Next, 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,
- $R = R_1 \cup \{S \to \varepsilon | S_1 S\}$.

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

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

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

$$L = \{0^n 1 0^n : n \geq 0\}.$$

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^*.

For the fourth part, we consider

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

and

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

- We show that L is context-free: Any string in L starts with zero 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

$$\begin{array}{lcl} S & \rightarrow & AX \\ A & \rightarrow & \varepsilon \mid aA \\ X & \rightarrow & \varepsilon \mid bXc \end{array}$$

  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 (and nothing else).

- By a symmetric argument, 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 last 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 previous 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 the first part: $\overline{A} \cup \overline{B}$ is context-free.
- By our assumption, $\overline{\overline{A} \cup \overline{B}}$ is context-free.
- By De Morgan, the latter language is equal to $A \cap B$.