1. The **symmetric difference** of two sets $S_1$ and $S_2$ is defined as

$$S_1 \ominus S_2 = \{ x : x \in S_1 \text{ or } x \in S_2, \text{ but } x \text{ is not in both } S_1 \text{ and } S_2 \}.$$ 

Show that the family of regular languages is closed under symmetric difference.

**Answer.**

From the definition of symmetric difference of two sets, we have that $S_1 \ominus S_2 = (S_1 \cap \overline{S}_2) \cup (S_2 \cap \overline{S}_1)$. Because of the closure of regular languages under intersection ($\cap$), complementation ($\overline{L}$), and union ($\cup$), the family of regular languages is closed under symmetric difference. \(\square\)

2. Let $L_1 = L(a^*baa^*)$ and $L_2 = L(aba^*)$. Find $L_1/L_2$.

**Answer.**

We first construct a DFA that accepts $L_1$ as follows. We check each state $q_0$, $q_1$, $q_2$, and $q_3$ to see whether there is a walk labeled $aba^*$ to the final state $q_2$. We see that only $q_0$ qualifies. Thus, the result $L_1/L_2 = L(a^*)$. \(\square\)

3. If $L$ is a regular language, prove that $L_1 = \{uv : u \in L, |v| = 2\}$ is also regular.

**Answer.**

From the definition of $L_1$, we have that $L_1 = LL'$, where $L' = \{v : |v| = 2\}$. $L'$ is regular since we can construct a DFA that accepts strings with two symbols. Thus, $L_1 = LL'$ is regular since the family of regular languages is closed under concatenation. \(\square\)

4. For a string $a_1a_2 \cdots a_n$ define the operation $\text{shift}$ as

$$\text{shift}(a_1a_2 \cdots a_n) = a_2 \cdots a_n a_1.$$ 

From this, we can define the operation on a language as

$$\text{shift}(L) = \{v : v = \text{shift}(w) \text{ for some } w \in L\}.$$
Show that the regularity is preserved under the \textit{shift} operation.

\textbf{Answer.}

Assume that the language $L$ is given in DFA $M = (Q, \Sigma, \delta, q_0, F)$. We construct an NFA $N = (Q', \Sigma, \delta', q_s, \{q_f\})$ satisfies that $\text{shift}(L) = L(N)$ with $Q' = Q \times \Sigma \cup \{q_s, q_f\}$ and $\delta'$ as follows. (Note that $Q \times \Sigma = \{[q, \sigma] : q \in Q, \sigma \in \Sigma\}$, where $[q, \sigma]$ is a state for the first symbol of the string in $L$ to be $\sigma$.)

- $\delta'(q_s, \lambda) = \{[\delta(q_0, \sigma), \sigma] : \sigma \in \Sigma\}$ (here we guess the first symbol to be $\sigma$, we will verify this guess "in the end" when the last symbol appears);
- $\delta'([q, \sigma], \sigma') = \{[\delta(q, \sigma'), \sigma]\}, \forall q \in Q, \forall \sigma, \sigma' \in \Sigma$;
- Add $q_f$ to $\delta'([r, \sigma], \sigma), \forall r \in F, \forall \sigma \in \Sigma$;

For example, the following is a DFA $M$ such that $L = L(M)$. The construction of the NFA $N$ satisfies that $\text{shift}(L) = L(N)$ is shown as follows.

Thus, the regularity is preserved under the \textit{shift} operation. \qed

5. Exhibit an algorithm that, given any three regular language, $L$, $L_1$, $L_2$, determines whether or not $L = L_1L_2$.

\textbf{Answer.}

Check if $L \text{xor } L_1L_2$ is empty or not. If yes, then $L = L_1L_2$. Otherwise, $L \neq L_1L_2$.

The algorithm for the check of $L \text{xor } L_1L_2$: Assume that the three languages are given in DFAs $M$, $M_1$, and $M_2$. 

2
• Construct an NFA $N$ for $L(M_1)L(M_2)$ by the algorithm in the textbook;
• Convert $N$ into DFA $M'$;
• Construct $M'' = M \times M'$ for $L(M)$ xor $L(M_1)L(M_2)$ by the algorithm given in the class;
• Minimize $M''$ as $M'''$ and determine whether $M'''$ is equivalent to the following DFA:

\[ \sigma \in \Sigma \]

\[ q \theta \]

6. Describe an algorithm which, when given a regular grammar $G$, can tell us whether or not $L(G) = \Sigma^*$.

Answer.

[Solution 1] Check if $L(G)$ xor $\Sigma^*$ is empty or not. If yes, then $L(G) = \Sigma^*$. Otherwise, $L(G) \neq \Sigma^*$. (This can be done by the method in Problem 5.)

[Solution 2] A desirable algorithm is described as follows.

• Convert the regular grammar $G$ to an NFA $M'$;
• Convert $M$ to a DFA $M$;
• Construct the complement DFA $\overline{M}$ of $M$ (change the non-final states in $M$ to final states and final states in $M$ to non-final states);
• Check if $\overline{M}$ accepts any string in $\Sigma^*$ or not. If no, that means $L(\overline{M}) = \emptyset$, i.e., $L(G) = \Sigma^*$. Otherwise, $L(G) \neq \Sigma^*$.

7. Prove that the following language is not regular.

\[ L = \{ a^n b^\ell a^k : n = \ell \text{ or } \ell \neq k \}. \]

Answer.

Let $m$ be the constant in the pumping lemma. We choose $w = a^m b^m a^m \in L$, $|w| = 3m \geq m$. For all possible $x, y, z$ with $w = xyz$, $|xy| \leq m$, $|y| \geq 1$, there are following cases:

• Case 1: $x = a^r$, $y = b^s$, $z = a^{m-r-s}b^m a^m$, $r + s \leq m$, $s \geq 1$. We let $i = 0$. $xy^0z = a^r(b^s)\overline{0}z = a^m b^m a^m \notin L$.
• Case 2: no other cases.
Thus, $L$ is not regular.

8. Prove that the following language is not regular.

$$L = \{ww : w \in \{a, b\}^*\}.$$ 

\textbf{Answer.}

Let $m$ be the constant in the pumping lemma. We choose $w = a^mbma^mb^m \in L$, $|w| = 4m \geq m$. For all possible $x, y, z$ with $w = xyz$, $|xy| \leq m$, $|y| \geq 1$, there are following cases:

- \textbf{Case 1:} $x = a^r$, $y = b^s$, $z = a^{m-r-s}b^ma^mb^m$, $r + s \leq m$, $s \geq 1$. We let $i = 0$.
  $$xy^0z = a^r(b^s)^0z = a^{m-s}b^ma^mb^m \notin L,$$
  \text{because}
  $$2m < 2^m + s \leq 2^m + m < 2^m + 2^m = 2^{m+1}, 2^m + s \neq 2^k \text{ for any } k$$

- \textbf{Case 2:} no other cases.

Thus, $L$ is not regular.

9. Determine whether or not the following language on $\Sigma = \{a\}$ is regular

$$L = \{a^n : n = 2^k \text{ for some } k \geq 0\}.$$ 

\textbf{Answer.}

Let $m$ be the constant in the pumping lemma. We choose $w = a^{2^m} \in L$, $|w| = 2^m \geq m$. For all possible $x, y, z$ with $w = xyz$, $|xy| \leq m$, $|y| \geq 1$, there are following cases:

- \textbf{Case 1:} $x = a^r$, $y = b^s$, $z = a^{2^m-r-s}b^ma^mb^m$, $r + s \leq m$, $s \geq 1$. We let $i = 2$.
  $$xy^2z = a^r(b^s)^2z = a^{2^m+s} \notin L,$$
  \text{because}
  $$2^m < 2^m + s \leq 2^m + m < 2^m + 2^m = 2^{m+1}, 2^m + s \neq 2^k \text{ for any } k$$

- \textbf{Case 2:} no other cases.

Thus, $L$ is not regular.

10. Make a conjecture whether or not the following language is regular. Then prove your conjecture.

$$L = \{a^n b^\ell : |n - \ell| = 2\}.$$ 

\textbf{Answer.}

Let $m$ be the constant in the pumping lemma. We choose $w = a^mb^{m+2} \in L$, $|w| = 2m + 2 \geq m$. For all possible $x, y, z$ with $w = xyz$, $|xy| \leq m$, $|y| \geq 1$, there are following cases:

- \textbf{Case 1:} $x = a^r$, $y = b^s$, $z = a^{m-r-s}b^{m+2}$, $r + s \leq m$, $s \geq 1$. We let $i = 0$.
  $$xy^0z = a^r(b^s)^0z = a^{m-s}b^{m+2} \notin L,$$
  \text{because}
  $$|(m - s) - (m + 2)| = |s - 2| \neq 2$$

- \textbf{Case 2:} no other cases.

Thus, $L$ is not regular.