Reminder

• In class we defined deterministic and non-deterministic finite automata (DFA/NFA) as 5-tuple of \((Q, \Sigma, \delta, q_0, F)\), and said that a language is regular if there is some finite automata that recognizes it.

• Closure Properties of regular languages: If \(A, B\) are regular, then so is their union \((A \cup B)\), concatenation \((A \circ B = AB)\), star \((A^*)\), intersection \((A \cap B)\), complement \((\overline{A})\) and reversal \((A^R)\). (the last two were in the HW, and the last one is also in Example 3 in these notes.)

• The pumping lemma as a tool to show non-regularity of languages.

**Lemma 1 (The Pumping Lemma)** If \(A\) is a regular language, then there is a number \(p\) (pumping length) such that if \(s \in A\) and \(|s| \geq p\), then \(s\) can be written as \(s = xyz\) with

1. \(xy^iz \in A\), for \(i \geq 0\),
2. \(|y| > 0\), and
3. \(|xy| \leq p\).

• Non-Regular languages:
  - \(\{1^n0^n \mid n \geq 0\}\), and
  - \(\{w \mid w\ \text{has= numbers of 0s and 1s}\}\).

**Example 1 — Binary Addition is Regular**

Let \(\Sigma_3\) be the set containing all columns of 0s and 1s of height three. Namely,

\[
\Sigma_3 = \left\{ \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \\ 1 \end{bmatrix}, \begin{bmatrix} 0 \\ 0 \\ 1 \\ 1 \\ 0 \\ 1 \end{bmatrix}, \cdots, \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} \right\}.
\]

A string of symbols in \(\Sigma_3\) gives three rows of 0s and 1s. Consider each row to be a binary number and let

\[
L_1 = \{w \in \Sigma_3^* \mid \text{the bottom row of } w \text{ is the binary addition of the top two rows}\}.
\]

For example,

\[
\begin{bmatrix} 1 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 1 \end{bmatrix}, \begin{bmatrix} 1 \end{bmatrix} \in L_1, \quad \begin{bmatrix} 1 \\ 1 \end{bmatrix}, \begin{bmatrix} 0 \\ 0 \end{bmatrix} \notin L_1,
\]

Prove that \(L_1\) is regular.
Solution

The idea is to use that regular languages are closed under reversal operation, and go over the addition operation bit by bit, starting from the least significant bit. When going over the addition bit by bit, we need to remember if the last addition resulted in carry, or not. This will be done by having two states, one that symbol we had carry, and one not.

The next automaton recognizes $L_1^R$.

What about the empty string? The automaton above assumes that the empty string is not an encoding of any number, and thus $\varepsilon \notin L_1$. However, it is conceivable that one decides to interprets the empty string as an encoding of the number 0 (in that case the language of valid encodings of the number 0 is $0^*$). Since $0 + 0 = 0$, we have $\varepsilon \in L_1$. The automaton that recognizes $L_1^R$ under this interpretation is even simpler than the above one. Its starting state will simply be the non-carry state (the original starting state will be removed), and the rest will remain the same.
Example 2 — Some Non-Regular Languages

Prove the following languages are non-regular.

1. \( L_2 = \{ x = y + z \mid x, y, z \text{ are binary integers, and } x \text{ is the sum of } y \text{ and } z \} \) over \( \Sigma = \{0, 1, +, =\} \).

2. \( L_3 = \{ 0^i 1^j \mid i \geq j \geq 0 \} \).

3. \( L_4 = \{ w \mid w \text{ has } \neq \text{ numbers of 0s and 1s} \} \).

Solution

1. We use the pumping lemma to “pump up”. Assume \( L_2 \) is regular. Use the pumping lemma to get a pumping length \( p \) satisfying the conditions of the pumping lemma. Set \( s = 1^p 1^p \) (note that the first ‘=’ means ‘equals to’, and the second ‘=’ is a letter in \( \Sigma \)). Obviously, \( s \in L_2 \) and \( |s| \geq p \). Thus, the pumping lemma implies that the string \( s \) can be written as \( xyz \) with \( x = 1^a, y = 1^b \) and \( z = 1^c = 0^p + 1^p \), where \( b \geq 1 \) and \( a + b + c = p \). However, the string \( s' = xy^2 z = 1^a + 2b + c = 0^p + 1^p \not\in L_2 \), since \( a + 2b + c > p \), and thus the binary integer represented by \( 1^a + 2b + c \) is greater than the binary integer represented by \( 0^p \). That contradicts the pumping lemma. Thus \( L_2 \) is not regular.

Note that this example shows that representation of the language matters. In the previous example, where the representation of integers and the alphabet were different, the language of addition was regular.

2. We use the pumping lemma to “pump down”. Assume \( L_3 \) is regular. Use the pumping lemma to get a pumping length \( p \) satisfying the conditions of the pumping lemma. Set \( s = 0^p 1^p \). Obviously, \( s \in L_2 \) and \( |s| \geq p \). Thus, the pumping lemma implies that the string \( s \) can be written as \( xyz \) with \( x = 0^a, y = 0^b \) and \( z = 0^c = 0^p + 1^p \), where \( b \geq 1 \) and \( a + b + c = p \). However, the string \( s' = xy^0 z = 0^a + 1^p \not\in L_2 \), since \( a + c < p \). That contradicts the pumping lemma. Thus \( L_3 \) is not regular.

3. We use the closure properties of regular languages. Assume \( L_4 \) is regular. Since the regular languages are closed under complement, \( L_4 = \{ w \mid w \text{ has } = \text{ numbers of 0s and 1s} \} \) is regular. But, we saw in class that the latter language is not regular, a contradiction. Thus \( L_4 \) is not regular.

Example 3 — The Regular Languages are closed Under Reversal

If \( A \) is any language, let \( A^R \) be the reversal language of \( A \). Formally, \( A^R = \{ w \mid w^R \in A \} \), where for \( w = w_1 w_2 \cdots w_n \), \( w^R = w_n w_{n-1} \cdots w_1 \).

Prove that the regular languages are closed under reversal. Namely, that if \( A \) is regular, then \( A^R \) is regular.

Solution

Given a DFA \( M \) that recognizes \( A \), we describe a NFA \( N \) that recognizes \( A^R \) using the following idea. The transition function will be the “inverse” of the original transition function. We turn the
start state into the accepting state and the accepting states into the start state. But what if $M$
had more than one accepting state? We add additional state to be the new single starting state,
and connect it via $\varepsilon$-transitions to the accepting states of $M$. The formal definition follows.

Let $M = (Q, \Sigma, \delta, q_0, F)$. Construct NFA $N = (Q \cup \{t\}, \Sigma, \delta', t, F')$, for
$t \notin Q$, that recognizing the reversal of $A$ as follows:

1. $F' = \{q_0\}$.

2. $\delta'(q, a) = \{r \in Q \mid \delta(r, a) = q\}$ if $q \neq t$, and $\delta'(t, \varepsilon) = F$. 