Sample solution to HW 1

(1) (a) The language \( L \) that consists of all the words in which \( a \) appears exactly twice.

\[
\begin{array}{cccc}
& b & a & b & b \\
q_0 & \rightarrow & q_1 & \rightarrow & q_2 \\
\end{array}
\]

(b) The language \( L \) that consists of all the words that starts with \( b \) and ends with \( a \).

\[
\begin{array}{cc}
& a, b \\
q_0 & \rightarrow & q_1 & \rightarrow & \rightarrow \\
\end{array}
\]

(c) The language \( L \) that consists of all the words that contains \( bba \).

\[
\begin{array}{cccc}
& b & b & a \\
q_0 & \rightarrow & q_1 & \rightarrow & q_2 & \rightarrow & q_3 \\
\end{array}
\]

(d) The language \( L \) that consists of all the words that do not contain \( ab \).

\[
\begin{array}{cc}
& a, b \\
q_0 & \rightarrow & q_1 & \rightarrow & q_2 & \rightarrow & \rightarrow \\
\end{array}
\]

Intuitively, as soon as the DFA “sees” \( a \), it cannot see any \( b \).

(e) The language \( L \) that consists of all the words \( w \) such that if \( w \) contains \( ab \), then \( w \) ends with \( ba \).

Define the following two languages.

\[
\begin{align*}
L_1 &= \{ w \mid w \text{ does not contain } ab \} \\
L_2 &= \{ w \mid w \text{ contains } ab \text{ and ends with } ba \text{ and their positions don’t intersect } \} \\
L_3 &= \{ w \mid w \text{ ends with } aba \} \\
\end{align*}
\]

Note that \( L = L_1 \cup L_2 \cup L_3 \). This observation already gives us a hint on how to construct an NFA for \( L \). An NFA for \( L_1 \cup L_2 \) is as follows.

\[
\begin{array}{cccc}
& b & a & a, b \\
q_0 & \rightarrow & q_1 & \rightarrow & q_2 & \rightarrow & \rightarrow \\
\end{array}
\]

To add \( L_3 \) into our NFA above, we simply “add” some states \( p_0, \ldots, p_4, r_1, \ldots, r_3 \) to capture those words that end with \( aba \), as shown in the following NFA.
(2) (a) $b^*ab^*ab^*$.
(b) $b\Sigma^*a$.
(c) $\Sigma^*bba\Sigma^*$.
(d) There are two ways to construct the regex for this language. One is by constructing the regex from the DFA in question (1.d) via the procedure described in the proof of Theorem 4.1. Another way is to observe the following: $w$ does not contain $ab$, if and only if one of the following holds.
- $w$ does not contain any $a$, i.e., it contains only $b$’s.
- Once an $a$ appears, no $b$ will appear.

With this observation, we can immediately construct the desired regex $e_1$ for $L$:

$$e_1 := b^* \cup b^*a^*.$$ 

(e) Recall that $L = L_1 \cup L_2 \cup L_3$, as mentioned in question (1). The regex for $L_1$ is $e_1$, as in question (2.d). The regex for $L_2$ and $L_3$ are:

$$e_2 := \Sigma^*ab\Sigma^*ba.$$ 
$$e_3 := \Sigma^*aba.$$ 

So the desired regex for $L$ is $e = e_1 \cup e_2 \cup e_3$, which is:

$$e = b^* \cup b^*a^* \cup \Sigma^*ab\Sigma^*ba \cup \Sigma^*aba.$$ 

(3) (a) $L$ consists of all the words in which $a$ appears exactly 3 times. $L$ is regular with regex:

$$b^*ab^*ab^*ab^*$$

(b) $L$ consists of all the words in which $a$ appears even number of times. $L$ is regular with regex:

$$b^* \cup (b^*ab^*ab^*)^*$$

(c) $L$ consists of all the words of even length. $L$ is regular with regex:

$$(\Sigma\Sigma)^*$$
(d) \(L = \{a^mba^n \mid 0 \leq m \leq n\}\).

\(L\) is not regular. The proof is via pumping lemma. Suppose to the contrary that \(L\) is regular. Let \(A = (\Sigma, Q, q_0, F, \delta)\) be its NFA.

Consider the word \(a^mba^n\), where \(m \geq |Q|\). By pumping lemma, we can partition \(a^m\) into three parts \(uvw\) such that for each \(i \geq 0\), \(uv^iwa^n\) is accepted by \(A\), which is not possible. We can choose an \(i \geq n\) which makes the number of a’s on the left hand side of \(b\) bigger than \(n\). Thus, there is no such NFA that accepts \(L\).

(e) \(L\) consists of all the words in which the number of occurrences of \(a\) is a prime number.

\(L\) is not regular. The proof is again via pumping lemma. Suppose to the contrary that \(L\) is regular. Let \(A = (\Sigma, Q, q_0, F, \delta)\) be its NFA.

Consider the word \(a^m\), where \(m\) is a prime number bigger than \(|Q|\). By pumping lemma, we can partition \(a^m\) into three parts \(uvw\) such that for each \(i \geq 0\), \(uv^iwa^n\) is accepted by \(A\).

Now, \(uvw\) consists of only \(a\)’s. So, the number of \(a\)’s in \(uv^iwa^n\) is precisely its length, which is:

\[|uv^iwa^n| = |u| + i|v| + |w|\]

If we consider \(i = |u| + |w|\), we have:

\[|uv^iwa^n| = |u| + (|u| + |w|)|v| + |w|\]

\[= (|u| + |w|)(|v| + 1)\]

Thus, \(|uv^iwa^n|\) cannot be a prime number. However, it is supposed to be accepted by \(A\). Therefore, there cannot be such NFA \(A\) that accepts \(L\).

(4) For a language \(L \subseteq \Sigma^*\) (not necessarily regular), we define the equivalence relation \(\sim_L\) on \(\Sigma^*\) as follows. \(u \sim_L v\), if the following holds: For every \(w \in \Sigma^*\), \(uw \in L\) if and only if \(vw \in L\). Prove that \(\sim_L\) is an equivalence relation and that \(L\) is a regular language if and only if \(\#(\sim_L)\) is finite.

That \(\sim_L\) is an equivalence relation is quite straightforward. So, we present the proof of the second part. We start with the “only if” part. Let \(L\) be a regular language and \(A\) be its DFA.

For a word \(w\), we denote by \(A(w)\) the state of \(A\) after reading \(w\). Or, more formally, if \(w = a_1 \cdots a_n\) and \(q_0a_1q_1 \cdots a_nq_n\) is the run of \(A\) on \(w\), then \(A(w) = q_n\).

We will first prove the following:

**Claim 1** For every words \(u, v\), if \(A(u) = A(v)\), then \(u \sim_L v\).

**Proof.** Let \(u\) and \(v\) be such that \(A(u) = A(v)\). Let \(u = a_1 \cdots a_n\) and \(v = b_1 \cdots b_m\).

We have to show that for every \(w \in \Sigma^*\), \(uw \in L\) if and only if \(vw \in L\). Let \(w = c_1 \cdots c_k\).

Consider the run of \(A\) on \(uw\):

\[p_0 a_1 p_1 \cdots a_n p_n c_1 t_1 \cdots c_k t_k\]

Likewise, consider the run of \(A\) on \(vw\):

\[s_0 b_1 s_1 \cdots b_m s_m c_1 t_1 \cdots c_k t_k\]

*This theorem is usually called Myhill-Nerode theorem.
Here both \( p_0, s_0 \) is the initial state of \( A \). Since \( A(u) = A(v) \), we have \( p_n = s_m \). Furthermore, \( A \) is deterministic. Thus, \( r_1 = t_1, \ldots, r_k = t_k \), and therefore,

\[
A(uw) = A(vw)
\]

This completes proof of Claim 1. \( \blacksquare \)

Claim 1 immediately implies that \( \#(\sim_L) \leq |Q| \), where \( Q \) is the set of states of \( A \). Thus, \( \#(\sim_L) \) is finite.

Now, we show the “if” direction. Let \( L \) be a language over \( \Sigma \), where \( \sim_L \) has finitely many equivalence classes \( C_1, \ldots, C_m \). Without loss of generality, we can assume that \( L \neq \emptyset \).

We first prove the following claim.

**Claim 2** There is \( i_1, \ldots, i_k \subseteq \{1, \ldots, m\} \) such that \( L = C_{i_1} \cup \cdots \cup C_{i_k} \). In other words, \( L \) is a union of some of the equivalence classes of \( \sim_L \).

**Proof.** Note that if \( w \sim_L v \), then either both of them are in \( L \), or both of them are not in \( L \). Thus, Claim 2 follows immediately. \( \blacksquare \)

Now, consider the following DFA \( A = (\Sigma, Q, q_0, F, \delta) \).

- \( Q = \{p_1, \ldots, p_m\} \), i.e., the number of states is precisely the number of equivalence classes in \( \sim_L \).
- \( q_0 \) is \( p_j \), where \( j \) is such that \( \epsilon \in C_j \).
- \( F = \{p_{i_1}, \ldots, p_{i_k}\} \), where \( i_1, \ldots, i_k \) are the indices in (5.a).
- \( \delta : Q \times \Sigma \to Q \) is defined as follows. For every \( p_i \in Q \), for every \( a \in \Sigma \), we pick an arbitrary \( w \in C_i \), and define \( \delta(p_i, a) = p_j \), where \( [wa] = C_j \).

Note that \( \delta \) is a well-defined function, i.e., for every \( w_1, w_2 \in C_i \), \( [w_1a] = [w_2a] \). In other words, the end result \( p_j \) remains the same for whichever \( w \) we pick, as long as \( w \) is from \( C_i \).

We will show that \( L(A) = L \). Recall that \( A(w) \) is the state of \( A \) after reading \( w \) starting from the initial state. From the construction of \( A \), for every word \( w \in \Sigma^* \), if \( [w] = C_j \), then \( A(w) = p_j \). Now,

\[
w \in L \text{ if and only if } w \in C_{i_1} \cup \cdots \cup C_{i_k}
\]

and hence,

\[
w \in C_{i_1} \cup \cdots \cup C_{i_k} \text{ if and only if } A(w) \text{ is one of } p_{i_1}, \ldots, p_{i_k}.
\]

Thus, \( w \in L \) if and only if \( w \in L(A) \), and hence, \( L = L(A) \).

---

1Here, for \( w \in \Sigma^* \), \([w]\) denotes the equivalence class of \( \sim_L \) that contains \( w \). See the notation in Lecture 1.