HW1

Due: n/a

Reminder: Submit your assignment on Gradescope by the due date. Submissions must be typeset. Each page should include work for only one problem (i.e., make a new page/new pages for each problem). See the course syllabus for the late policy.

While collaboration is encouraged, please remember not to take away notes from any labs or collaboration sessions. Your work should be your own. Use of other third-party resources is strictly forbidden.

Please monitor ED discussion, as we will post clarifications of questions there.

Problem 1

Describe the language $L$ recognized by the DFA $D$ shown below, and prove that $L = L(D)$.

\[ L = \{ w \mid w \text{ has no consecutive 0's} \} \]

Solution:

Sample Proof. We will prove that the described language is recognizes by the DFA $D$ by proving the following claims:

1. All words not in the language are rejected by the DFA 
   \[ (w \notin L \implies w \notin L(D)) \]
2. All words rejected by the DFA are not in the language

\[(w \notin L(D) \implies w \notin L)\]

1. To prove that all words not in the language are rejected by the DFA, consider the \(L\)'s description. By definition of \(L\), any word not in \(L\) must have at least one occurrence of consecutive 0's. When the DFA eventually begins processing these 0's, it can be in one of three states: \(q_0\), \(q_1\), and \(q_2\). Notice that a sequence of consecutive 0's contains at least two 0's. Regardless of the state we are in, realize that 2 consecutive 0's will get us to \(q_2\) – either we've started in \(q_2\) and stayed there, or by the transitions, we've ended up in \(q_2\). Realize that because words cannot leave \(q_2\) and \(q_2\) is not an accepting state, we will not accept these words. So, words with consecutive 0's are rejected.

2. To show that all words rejected by the DFA are not in the language realize the following. The only non-accepting state we have is \(q_2\). The only way to get to \(q_2\) is to get a 0 when we're in \(q_1\). The only way to get to \(q_1\) is to get a 0 when we're in \(q_0\). Since \(q_1\) exits immediately back to \(q_0\) if \(q_1\) encounters a 1, we know that the only way to get to \(q_2\) is to strictly get 2 0's in a row (since every time we get a 1 outside of these states, we end back at \(q_0\)).

Thus, all words rejected by the DFA must have consecutive 0's.

This is what we needed to show.

**Problem 2**

Design a DFA for each of the following languages over the alphabet \(\{0, 1\}\), and provide a simple proof of the correctness of each DFA:

1. \(\{w_B | \text{the length of } w_B \text{ is at least 2}\}\)
2. \(\{w_B | \text{each time a 0 appears in } w_B, \text{ it is never followed by a 1}\}\)

**Solution:**

1. [Diagram of DFA]
Sample Proof. We will prove that this finite state machine is correct by construction. In particular, note that any input that terminates consumption in $q_0$ is of length 0, any input that terminates consumption in $q_1$ is of length 1, and any input that terminates consumption in $q_2$ is of length 2. As such, we accept any string that terminates in these states. Any other string is rejected.

2.

![Diagram](image.png)

Sample Proof. We will prove that the described language is recognized by the DFA $D$ by proving the following claims:

(a) All words in the language are accepted by the DFA
\[ (w \in L \implies w \in L(D)) \]

(b) All words not in the language are rejected by the DFA
\[ (w \notin L \implies w \notin L(D)) \]

a) Notice that the first time a 0 appears in a word, the rest of the word cannot contain any 1’s if we want the word fulfill the criterion of the language. Thus, each word in $L$ consists of a sequence of ones (possibly of length zero) followed by a sequence of zeroes (also possibly of length zero). By the DFA’s construction, processing the sequence of 1’s leaves us at $q_0$, as $q_0$ is the start state and transitions to itself on 1. Processing the sequence of 0’s either leaves us at $q_0$ if the sequence is empty or brings us to and stops at $q_1$. Both of these states are accepting, so all word in $L$ are accepted by the DFA.

b) For a word to not be in $L$, it must contain a sequence of at least one zero followed by a 1. Regardless of what state we are in, processing a sequence of 0’s followed by a 1 leaves us in state $q_2$, which is a rejecting state that loops on itself for all symbols in our alphabet. Thus, any word not in $L$ would be rejected.

Since all words in $L$ are accepted and all words not in $L$ are rejected, the DFA recognizes $L$. 
Problem 3

The following language (over the alphabet \{a, b\}) is the intersection of two simpler languages:

\[ L = \{ w | w \text{ contains at least 2 } a\text{'s and exactly 1 } b \} \]

1. Identify the two simpler languages and prove that they are regular.

2. Using what you just proved, explain why \( L \) must also be regular.

Solution:

1. The simpler languages:

(a) \( \{ w_A | w_A \text{ contains at least 2 } a\text{'s} \} \)

\[ q_0 \xrightarrow{a} q_1 \xrightarrow{a} q_2 \]
\[ b \xrightarrow{b} \]

Sample Proof. We will prove that the finite state diagram above is correct by construction. In particular, we can notice that any element that terminates consumption in \( q_0 \) must have no \( a\)’s, any element that terminates consumption in \( q_1 \) must have 1 \( a \), and any element that terminates consumption in \( q_2 \) must have 2 or more \( a\)’s, since once \( q_2 \) is reached, a string will stay in \( q_2 \) no matter what the rest of the string is. Thus, since \( q_2 \) is the only accept state, this DFA accepts the set of strings with at least 2 \( a\)’s, and all other strings are rejected.

(b) \( \{ w_B | w_B \text{ contains exactly 1 } b \} \)

\[ q_0 \xrightarrow{b} q_1 \]
\[ a \xrightarrow{a} \]
Sample Proof. We will prove that the finite state diagram above is correct by construction. In particular, we can notice that any element that terminates consumption in \( q_0 \) must have no \( b \)'s, any element that terminates consumption in \( q_1 \) must have 1 \( b \), and any other element will move from \( q_1 \) to the reject state. Thus, since \( q_1 \) is the only accept state, this DFA accepts the set of strings with exactly 1 \( b \), and all other strings are rejected.

2. We have proven above that the languages \( \{ w_A \mid w_A \text{ contains at least 2 } a \text{'s} \} \) and \( \{ w_B \mid w_B \text{ contains exactly 1 } b \} \) are regular. Notice that by construction, \( L \) is simply the intersection of these two languages. Since regular languages are closed under intersection, this means that \( L \) itself must be regular.

Sample Proof of closure under intersection: We will first prove that regular languages are closed under complementation then apply closure under complementation and union to

\[
L_1 \cap L_2 = \overline{L_1} \cup \overline{L_2}
\]

Sample Proof of closure under complement: Let \( L \) be a regular language, and let \( D \) be a DFA such that \( L(D) = L \). By definition,

\[
\overline{L} = \{ w \in \Sigma^* \mid w \notin L \}.
\]

Because \( L \) is a regular language, for all \( w \in L \), \( w \) will terminate on an accepting state. Since every string in \( \Sigma^* \) must terminate on some state by definition of a DFA, for all \( w \in \overline{L} \), \( w \) must terminate on a non-accepting state. Now consider \( \overline{D} \), a DFA whose transitions are identical to \( D \) but whose non-accepting states are now accepting states and vice-versa. By our construction, all \( w \in \overline{L} \) terminate on an accepting state in \( \overline{D} \) (and a non-accepting one in \( D \)), so \( \overline{L} \subseteq L(\overline{D}) \).

Now, we must prove \( L(\overline{D}) \subseteq \overline{L} \). For each \( w \in L(\overline{D}) \), \( w \) must terminate on an accepting state in \( \overline{D} \) and a non-accepting state in \( D \) by our construction of \( \overline{D} \). By definition, \( w \notin L(D) \), and since \( L = L(D) \), \( w \notin L \). This is equivalent to \( w \in \overline{L} \). Thus, if \( w \in L(\overline{D}) \), then \( w \in \overline{L} \), so \( L(\overline{D}) \subseteq \overline{L} \). Because \( L(\overline{D}) \subseteq \overline{L} \) and \( \overline{L}(\overline{D}) \), we know \( \overline{L} = L(\overline{D}) \), so \( \overline{L} \) is a regular language. We have proven that the complement of a regular language is a regular language.

All that's left is to apply the complement to each of \( L_1 \) and \( L_2 \) to get the regular languages \( \overline{L_1} \) and \( \overline{L_2} \), apply the union to get \( \overline{L_1} \cup \overline{L_2} \), and finally apply the complement once more to arrive at the regular
language $L_1 \cap L_2$. (Note: accept solutions which create a DFA for $L$
and prove its correctness as well.)