Some Homework 3 Solutions

Note: These solutions are not necessarily model answers. Rather, they are designed to be tutorial in nature, and sometimes contain a little more explanation than an ideal solution. Also, bear in mind that there may be more than one correct solution.

1. (a) Not regular. Proof by contradiction: assume that \( L = \{ w : w \text{ has balanced parentheses} \} \) is regular. Let \( n \) be the constant guaranteed to exist by the pumping lemma. Consider the string \( w = (n)^n \) — i.e., \( n \) (‘s followed by \( n \) )’s. Clearly \( w \) has balanced parentheses, so \( w \in L \). Thus, since \( |w| \geq n \), by the pumping lemma we must be able to write \( w = xyz \) with \( |xy| \leq n \), \( |y| \geq 1 \), and such that \( xy^iz \in L \) for all \( i \geq 0 \). However, since \( w \) starts with \( n \) (‘s, \( y \) must consist entirely of one or more (‘s. Therefore, for any \( i > 1 \), \( xy^iz \notin L \) since it has more (‘s than )’s. This is a contradiction, so \( L \) is not regular.

(b) Regular. The key observation here is that successive occurrences of \( abb \) and of \( bba \) in any string over \( \{a, b\} \) must alternate along the string. To see this, one can show that in any string \( w \), between any two occurrences of \( abb \) there is an occurrence of \( bba \) and vice versa. Consider an arbitrary substring of \( w \) delimited by two occurrences of \( abb \). This string has the form \( abbuab \), where \( u \) is a possibly empty string. If \( u \) contains no \( a \) symbols, then the string \( bbua \) ends in \( bba \). Otherwise, suppose that the first \( a \) in \( u \) occurs at position \( i \); then the string \( bbu_1 \ldots u_i \) ends in \( bba \). For the other direction, again consider an arbitrary substring of \( w \) delimited by two occurrences of \( bba \). Then the reversal \( w^R \) of \( w \) has the form \( abbuab \) for some string \( u \). By the above argument, \( w^R \) must contain \( bba \) as a substring, so \( w \) itself contains an occurrence of \( abb \).

For a string \( w \), let \( D(w) \) denote the difference between the number of occurrences of \( abb \) and of \( bba \) in \( w \). By the above argument, for any \( w \), \( |D(w)| \leq 1 \). At this point it is not difficult to see what a DFA for our language should look like. The states should keep track of the last two symbols seen, as well as the sign of the quantity \( D(w) \). (See diagram; the start state is \( q_{st} \), and the accept states are marked in thicker lines.)
[The key point in this question is to observe that occurrences of abb and bba must alternate along the string.]

(e) Not regular. Assume that this language is regular, and let \( n \) be any number bigger than both 2 and the pumping length. Consider the string \( w = a^n b^n \); this contains \( n - 2 \) copies of \( aaa \) and \( n - 2 \) copies of \( bbb \), so it is in the language. By the pumping lemma there exists a split \( w = xyz \) such that \(|xy| \leq n\), \( y \neq \epsilon \) and \( xy^2z \) is also in the language. However, whenever \( x \) and \( y \) satisfy the first two conditions, the string \( xy^2z \) will be of the form \( a^m b^n \), for some \( m > n \). This string has more copies of \( aaa \) than \( bbb \), so it cannot be in the language. Contradiction.

(f) Not regular. We do a proof by contradiction using closure properties. (Note that it apparently isn’t possible to use the pumping lemma directly here, because we’d have to show that any possible pumping of a substring \( y \) leads to a string that’s not in \( L \), which is hard as \( L \) is not very tightly constrained.)

So assume \( L = \{0^i 1^j : i, j \geq 0 \text{ and } i \neq j \} \) is regular. Since regular languages are closed under complementation, the complement \( \overline{L} \) is also regular. Now consider the language \( L' = \{0^i 1^j : i, j \geq 0 \} \), which is certainly regular (it is denoted by the regular expression \( 0^*1^* \)). Since regular languages are also closed under intersection, \( \overline{L} \cap L' \) must also be regular. However, \( \overline{L} \cap L' = \{0^i 1^j : i \geq 0 \} \), which we know is not regular (as we saw in class, by the same argument we used to show that the set of 0-1 strings with equal numbers of 0’s and 1’s is not regular). Therefore we have a contradiction, so we deduce that \( L \) itself must not be regular.

An alternative argument for this part, using the Myhill-Nerode Theorem, goes as follows. We show that the relation \( \sim_L \) splits \( \{0, 1\}^* \) into infinitely many equivalence classes, which implies that \( L \) is not regular. Indeed, consider the collection of strings \( C = \{0^n : n \geq 0\} \). We claim that all strings in \( C \) are in distinct equivalence classes. For suppose that there exists \( m \neq n \) such that \( 0^m \sim_L 0^n \). Then, by the definition of \( \sim_L \), \( 0^m 1^n \sim_L 0^n 1^n \). But this is impossible since \( 0^m 1^n \in L \) while \( 0^n 1^n \notin L \).

2. (a) Since \( L \) is regular, there is a DFA \( M \) that accepts it. Modify the DFA in the following way: take all outgoing edges from all the accepting states (including self-loops) and reroute them to point to a dead state. We claim that the resulting DFA \( M' \) decides \( \text{min}(L) \). To see this, note that \( M' \) certainly cannot accept any string that is not accepted by \( M \). And a string \( w \) is accepted by \( M' \) iff the accepting computation of \( M \) on \( w \) does not pass through any intermediate accepting states. But this latter condition corresponds precisely to saying that no proper prefix of \( w \) is accepted by \( M \), as required.

(b) (not *) This language is finite and therefore regular, since every finite language is regular. (To see this, just write down a regular expression that takes the union of the singleton strings.) However, note that given a FA for \( L \) we do not in general know how to construct a FA for \( L_{10} \); we know only that such an FA exists. Thus, unlike parts (a) and (c) of this problem, this proof is not constructive.

(d) (coming)

4 (b) False. E.g., let \( \Sigma = \{0, 1\} \), \( L = \{0^n 1^n : n \geq 0\} \), and \( L' = \{0^n 1^n : m \neq n\} \). Then \( L \) and \( L' \) are both non-regular. However \( L_1 \cap L_2 = \emptyset \), which is regular.

(c) False. For a counterexample, let \( L \) be any non-regular language (e.g., \( L = \{0^i 1^i : i \geq 0\} \)). Then we can write \( L = \bigcup_{i=1}^{\infty} L_i \), where each \( L_i \) consists just of the \( i \)th string in \( L \) in lexicographic order. Clearly each \( L_i \) is finite and hence regular. However, the union of all of the \( L_i \) is \( L \), which is not regular.