Problem 1 Warmup.

a. Prove that if a language $L$ contains finitely many elements then it is regular.

b. If a language $L$ contains infinitely many elements, can it still be regular?

c. Give the state diagram for a DFA accepting the language consisting of all strings over $\Sigma = \{a, b, c, d\}$ that are sorted, i.e., where any $a$s occur before any $b$s before any $c$s before any $d$s. For example, $\text{abbd}$ is sorted, whereas $\text{da}$ is not. Note that we consider the empty string to be sorted.

Problem 2 Let $M = (Q, \Sigma, \delta, q_0, F)$ be a DFA. Let $w = w_1w_2\cdots w_n$ be a string accepted by $M$, for which let $r_0, r_1, \ldots, r_n$ be the corresponding accepting path. (See the section titled “Formal Definition of Computation,” page 40, in Sipser.) Suppose that $r_i = r_j$ for some $i$ and $j$ such that $i < j$. Prove that, in addition to accepting $w$, $M$ also accepts some string $w'$ that is shorter than $w$, i.e., such that $|w'| < |w|$.

Problem 3 We defined DFAs to have a set $F$ of accepting states. In this problem we consider a variant of DFAs that have exactly one accepting state, $q_{\text{accept}}$. We call this variant of DFAs one-DFAs.

a. Give a formal definition of one-DFAs, including the parts of a one-DFA, what it means for a one-DFA to accept a string $w$, and the language of a one-DFA.

b. Show that for any one-DFA $M_1$ there exists an equivalent DFA $M$ (i.e., such that $L(M_1) = L(M)$).

c. Show that the converse is not true: there exist regular languages not recognized by any one-DFA.

d. Consider an analogous definition of one-NFAs, NFAs with exactly one accepting state. How do one-NFAs compare to NFAs?