CS 334 Fall 2021: Problem Set 2.

Problem 1. (10 points) For each of the following statements either argue that it is true, or else give a counterexample. Keep your explanations brief.

a) Every finite language is regular.
b) If every state of an NFA is an accept state then its language is Σ^*
c) If a k-state DFA accepts a string of length k then its language is infinite.
d) If every state of an NFA for language L is flipped then the new NFA accepts the complement of L.
e) If every state of an NFA is flipped then the language accepted is not necessarily regular.

Problem 2. (10 points) Similar to an NFA in structure, an OFA (obsessive finite state machine) is one that accepts an input string if and only if *every* possible final state after the entire input state has been processed is an accept state of the machine. In other words, every computation path that does not die must end in an accept state. Prove that OFAs recognize all and only regular languages.

Problem 3. (15 points) For any string $w = w_1 w_2 \cdots w_n$, the *reverse* of w, written w^R, is the string w in reverse order, $w_n \cdots w_2 w_1$. For any language A, let $A^R = \{w^R | w \in A\}$.
Show that if A is regular, so is A^R.

Problem 4. (20 points) In this problem you will design an FSA that checks if the sum of two numbers equals a third number. Each number is an arbitrarily long string of bits. At each step, the input to the FSA is a symbol that encodes 3 bits, one from each number. In other words, the alphabet of the FSA is:

$$\Sigma = \left\{ \begin{bmatrix}0\\0\\0\end{bmatrix}, \begin{bmatrix}0\\0\\1\end{bmatrix}, \begin{bmatrix}0\\1\\0\end{bmatrix}, \begin{bmatrix}0\\1\\1\end{bmatrix}, \begin{bmatrix}1\\0\\0\end{bmatrix}, \begin{bmatrix}1\\0\\1\end{bmatrix} \begin{bmatrix}1\\1\\0\end{bmatrix}, \begin{bmatrix}1\\1\\1\end{bmatrix} \right\}$$

In the input string $\begin{bmatrix}0\\0\\1\end{bmatrix}\begin{bmatrix}1\\0\\1\end{bmatrix}$ the bits in the top row represent the number 01, the bits in the middle row represent 00 and the third row represents 11. In this case, since $01+00 \neq 11$, the FSA must reject the input. On the other hand, the input string $\begin{bmatrix}0\\0\\1\end{bmatrix}\begin{bmatrix}0\\1\\0\end{bmatrix}\begin{bmatrix}1\\1\\0\end{bmatrix}$ will be accepted by the FSA since $001 + 011 = 100$.

Formally, let $B = \{w \in \Sigma^*: the\ bottom\ row\ of\ w\ equals\ the\ sum\ of\ the\ top\ two\ rows\}$, where Σ^* represents all finite strings over the alphabet Σ. Your goal is to design an FSA for the language B.

To get started, first design a 2-state FSA that recognizes B^R – this should be straightforward because the input arrives least significant bits first and most significant bits at the end. Next, use the technique of Problem 2 to design the final FSA for B.

Extra Credit Problem (20 points) For any string σ, over alphabet Σ, we define the string $SHIFT(\sigma)$ as follows: if $\sigma = aw, a \in \Sigma, w \in \Sigma^*$ then $SHIFT(\sigma) = wa$. For example, $SHIFT(0111) = 1110,\ and\ SHIFT(10110) = 01101.$
Prove that if L is regular, then so is $SHIFT(L) = \{SHIFT(\sigma): \sigma \in L\}$.