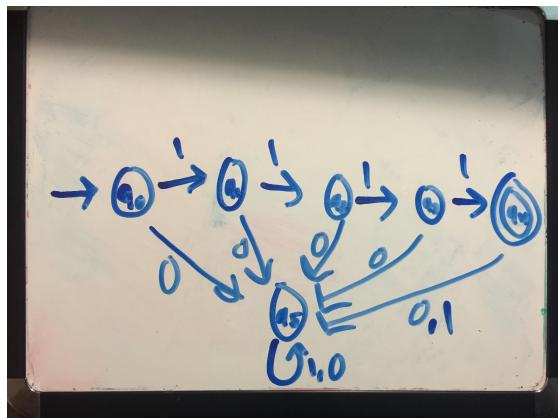


Simrun Heir and Michael Sanchez

I pledge my honor that I have abided by the Stevens Honor System.

Problem 1:

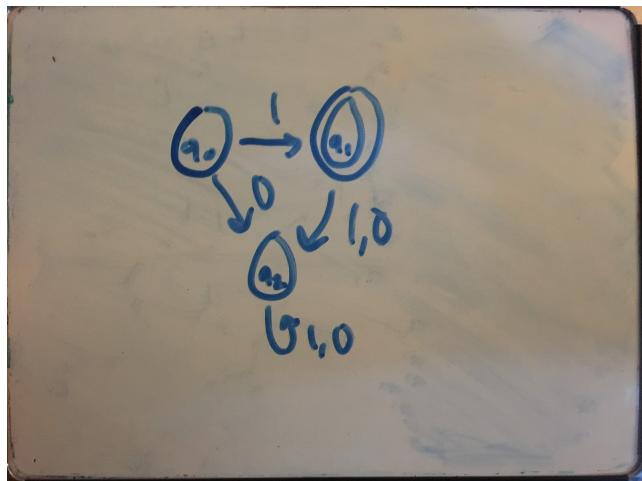


a.

Explanation: In order to ensure that the only input accepted is 1111, any input of 0 results in the machine getting caught in the state  $q_5$ , which loops back into itself for any input, 1 or 0, and if too many 1's are input, the machine will still get caught in  $q_5$ .

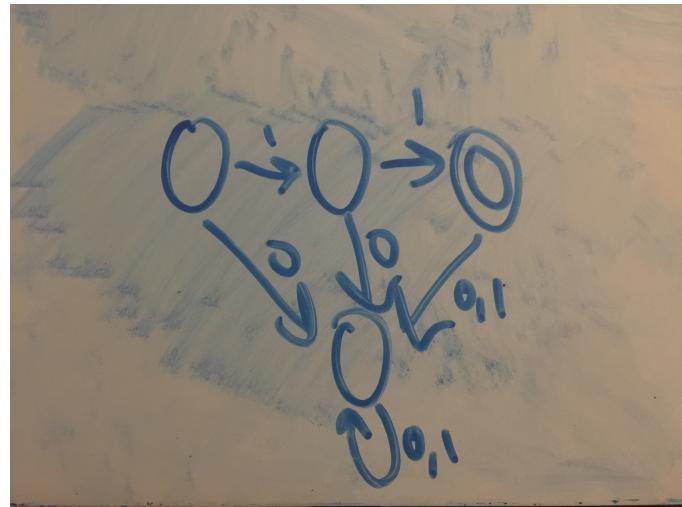
b. For each number  $k$  greater than or equal to 3, there exists a language that can be accepted by an FA with  $k$  states that cannot be recognized by an FSA with fewer than  $k$  states.

i. Let  $M_1$  be a  $k$  state FSA  $M_1 = (Q_1, \Sigma_1, \delta_1, q_0, F_1)$ , where  $k = 3$ , and  $\Sigma_1 = \{0, 1\}$ , constructed as follows



ii. Let  $L_1$  be a regular language, where  $L_1 \in \Sigma$  and  $L_1 = \{1\}$

iii. Let  $M_2$  be a  $k+1$  state FSA  $M_2 = (Q_2, \Sigma_2, \delta_2, p_0, F_2)$ , where  $\Sigma_2 = \{0, 1\}$ , constructed as follows



- iv. Let  $L_2$  be a regular language, where  $L_2 \in \Sigma_2$  and  $L_2 = \{11\}$
- v.  $M_1$  does not accept  $L_2$
- vi. By induction, for all  $k \geq 3$  there is a language that is accepted by a  $k$ -state FSA but not an FSA with less than  $k$  states

### Problem 2:

Let A be regular. This means there exists a DFA S that recognizes this language. w represents a string that is part of the language A. w<sub>1</sub> is the first input of the string, it causes the transition out of the start state q<sub>0</sub>. w<sub>n</sub> is the last input of the string, which brings the S into its accept state. Since w<sup>R</sup> is the reverse of w, w<sub>n</sub> is now the first input and w<sub>1</sub> is now the last input. Using this, you are able to create a DFA R by going through S backwards according to the reverse order given by w<sup>R</sup>. This means that the accept state of S becomes the start state of R, q'<sub>0</sub>, and the start state of S is the accept state of R. Since A<sup>R</sup> = {w<sup>R</sup> | w is an element of A} and w<sup>R</sup> is accepted by R, A<sup>R</sup> is regular.

### Problem 3:

9:33 PM Fri Sep 18

$\Sigma = \{0, 1\}$

Problem 3 make 3 state DFA that recognizes  $B^R$  then use method from problem 2 to make final FSA for B

DFA  $B^R$ :

\*state represent current value of carry bit  
 $q_0$ : carry 0  
 $q_1$ : carry 1  
 $q_2$ : failed addition

Transition function:

- $q_0 \rightarrow q_1$ : when you have a carry of 0, and the addition is correct, resulting in a carry of 0
- $q_0 \rightarrow q_2$ : when you have a carry of 0, and the addition is correct, resulting in a carry of 1
- $q_0 \rightarrow q_2$ : when you have a carry of 0, and the addition is incorrect
- $q_1 \rightarrow q_1$ : when you have a carry of 1, and the addition is correct, resulting in a carry of 1
- $q_1 \rightarrow q_2$ : when you have a carry of 1, and the addition is correct, resulting in a carry of 0
- $q_1 \rightarrow q_2$ : when you have a carry of 1, and the addition is incorrect
- $q_2 \rightarrow q_2$ : when the addition at some point before the current input was incorrect, thusly causing the string to be invalid

Similar to Problem 2, The DFA for B would follow a reverse order of  $B^R$  as the reverse of  $B^R$  is B.

DFA B:

$q_0$ : carry 0  
 $q_1$ : carry 1  
 $q_2$ : failed addition

each state represents the carry that must be produced by the addition of the bit in the current input; if the addition is possible, then the transition function takes the DFA to the carry that is needed to make the addition for the input work; if the addition is not possible, you are not able to produce the carry with the bits in the input and a carry of either 0 or 1, then the addition fails and the machine transitions to  $q_2$  and rejects the string.