CS150 Homework 2

Ann-Marina Miyaguchi − amiya017
Ivan Neto − ineto001
Due Date: Apr 11, 2022

Problem 1: (1 point) Give an NFA recognizing $L = \{w \mid w \text{ begins with 1 and ends with 0 over } \{0,1\}^*\}$.

Solution 1:

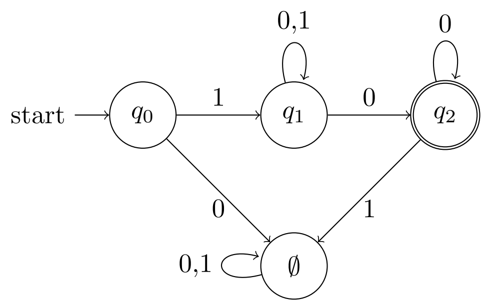


Figure 1: We start at q_0. From there, if 0, then we go to the hang state and the machine breaks. If 1, then q_1 is reached. From q_1, any input can reach back into q_1. Finally, if 0 from q_1 then the accept state is reached. From the accept state q_2, if 1 then the hang state is reached because the input does not end with 0. If 0 from q_2, we remain in the accept state.

Problem 2: (1 point) Give a DFA recognizing $L = \{w \mid \text{every odd position of } w \text{ is a 1 over } \{0,1\}^*\}$. For example, the "0" is in the 3rd (an odd position) for "a101" and "a" is in the 1st position (also odd), but the "1's" are in even positions.

Solution 2:

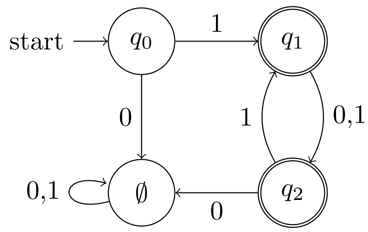


Figure 2: We begin at q_0. On input 0, we immediately go to the hang state because that would mean there is a 0 at position 1. Instead, on input 1, we reach q_1. From here, we do not care about the input at this state because it is an even position. So at input $x = (1 \cup 0)$, we move to q_2. At this point, we want a 1 because the incoming character will be an odd position. If 1, then we accept the string and go back to q_1, with the process starting over. However, if 0, then we go to the hang state because the 0 is in an odd position and thus not part of L.

Problem 3: (2 points) Draw an NFA accepting the set of strings over $\{0, 1\}$ whose 5th symbol from the right is 0. How many states would you say the equivalent DFA would take?

Solution 3:

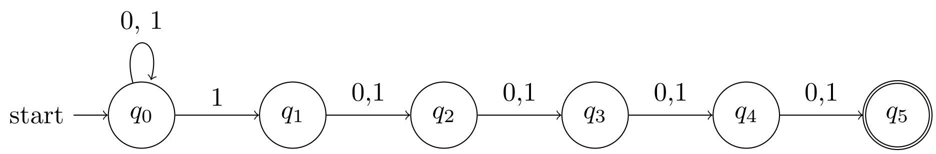


Figure 3: Thank you to the following source for a tutorial on tikz automata (Overleaf): `https://www3.nd.edu/~kogge/courses/cse30151-fa17/Public/other/tikz_tutorial.pdf`

The equivalent DFA would take $2^n = 2^6 = 64$ states. Because an NFA can be described with the states given by the power set of the states in the NFA $(P(Q))$, and because $|P(Q)| = 2^{|Q|}$, the maximum number of states that the DFA could have is $2^6 = 64$. However, this is not the actual number that we will use for the DFA because some of the states in the power set do not have transitions coming out of them. Thus, those can be discarded. But at maximum, the number of states that the equivalent DFA to this NFA would have is 64.

Problem 4: (2 points total) Give *either* a DFA or an NFA accepting the following languages over $\Sigma = \{0, 1\}$. Please state which kind you provided (DFA or NFA).

a. the set of strings with 011 as a substring

b. $\{w \mid w$ does NOT contain the substring $110\}$

Solution 4:

Part a): NFA

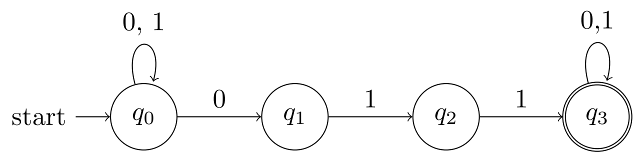


Figure 4: We begin at q_0 and remain at q_0 on any input until we reach a possible 011 substring. Then on 0, we move to q_1. We do this with the input 011 until we reach q_3. Then, any input is accepted and thus the regular expression (0|1)*011(0|1)* is accepted, which is the language described by a.

Part b): DFA

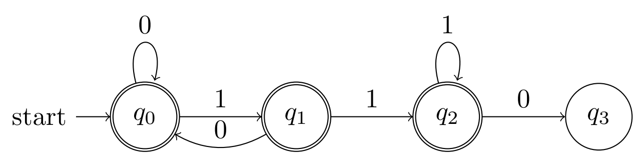


Figure 5: We create a DFA for this task. We begin at q_0 and wait until a 1 is reached by remaining at q_1 on any 0 input. All such strings must be accepted because they do not contain 110. Then, once the machine begins detecting 110, on input 1, we move to q_1. This string should be accepted because at this point 110 is not present. Then, if an input 0 is detected, we go back to q_0 because the processing "resets". Instead, on input 1, we move to q_2. At this point, the expression 0^*11 has been reached. Finally, on input 1, we can accept because then the string 0^*111 has been reached, which does not contain 110. Alternatively, on input 0, we must reject the string and any subsequent strings because then at this point the string 0^*110 has been reached, which is not in th

Problem 5: (4 points) Give proof sketches that the regular languages are closed under:

a. union

b. intersection

c. concatenation

d. reversals

e. complements

f. Kleene star

Solution 5:

Part a.

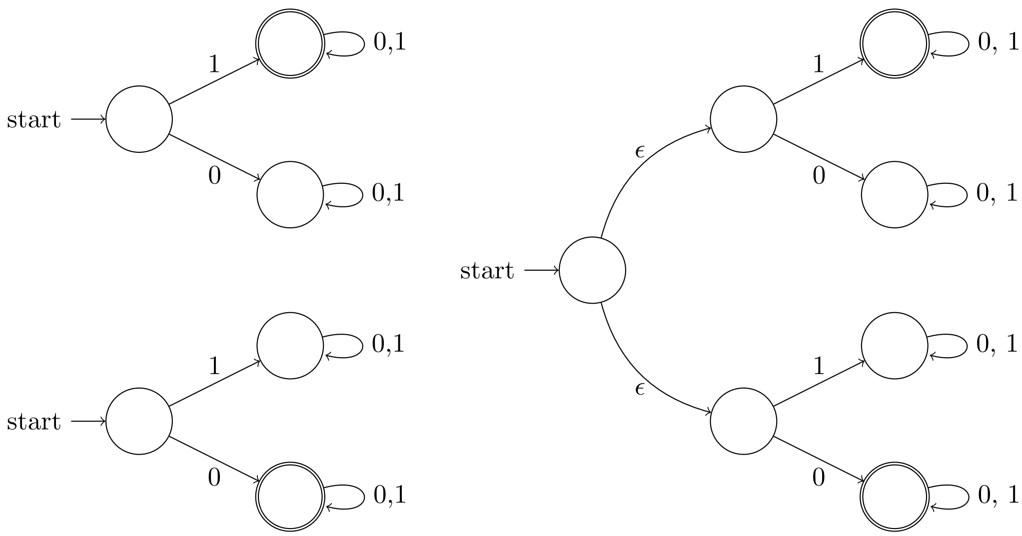


Figure 6: On the left, M_1 and M_2. On the right, the union between M_1 and M_2, $M = M_1 \cup M_2$.

Part b.

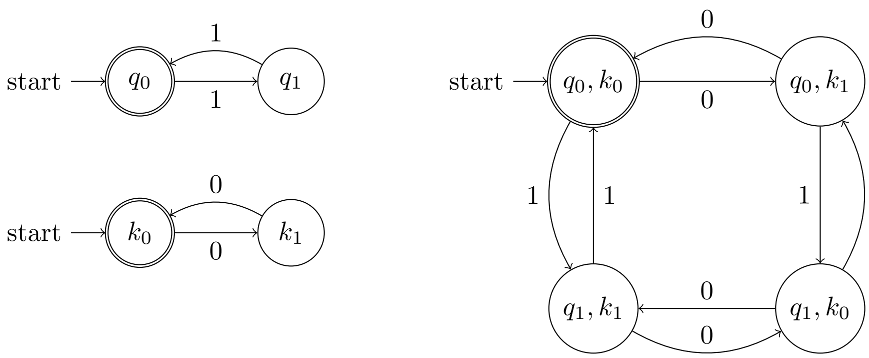


Figure 7: On the left, M_1 and M_2. On the right, the automata which accepts the intersection of $L(M_1)$ and $L(M_2)$. $Q = Q_1 \times Q_2$, $F = F_1 \times F_2$, $\delta = Q \times \Sigma \to Q$. The states for our automata ends up being the Cartesian product of Q_1 and Q_2. As we such, we end up with 4 different states

Part c.

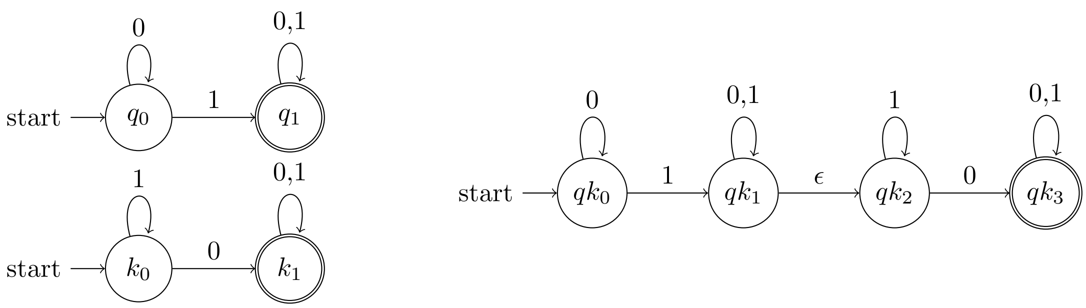


Figure 8: On the left, M_1 and M_2. On the right, the automata which accepts the concatenation of $L(M_1)$ and $L(M_2)$, $M = M_1 \cdot M_2$.

Part d.

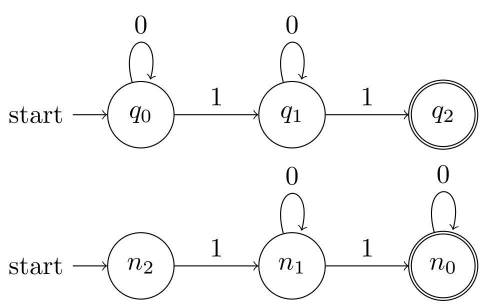
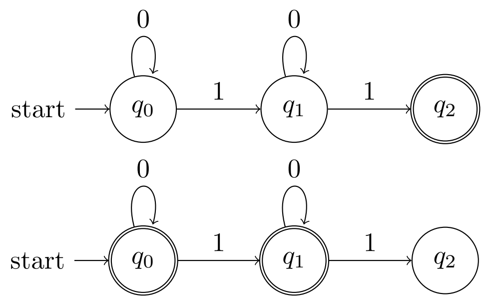


Figure 9: On the top, M_1 represents the language $L(M_1) = 0^*10^*1$. On the bottom, M_2 represents the language $L(M_2) = 10^*10^*$. $R(A)$ denotes the reverse operation of a language A. Thus $R(L(M_1)) = L(M_2)$. In order to do this, we flipped all transition directions, and switched the start state with the end state.

Part e.



Part f.

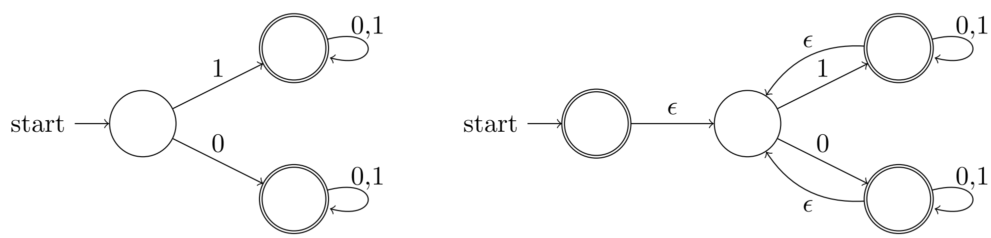


Figure 10: On the left, M. On the right, the automata which recognizes $L(M)^*$.

Problem 6: (1 **bonus** point) Draw an NFA accepting the set of strings over $\{0, 1\}$ such that the number of 0's is divisible by 3 and the number of 1's is divisible by 2.

Solution 6:

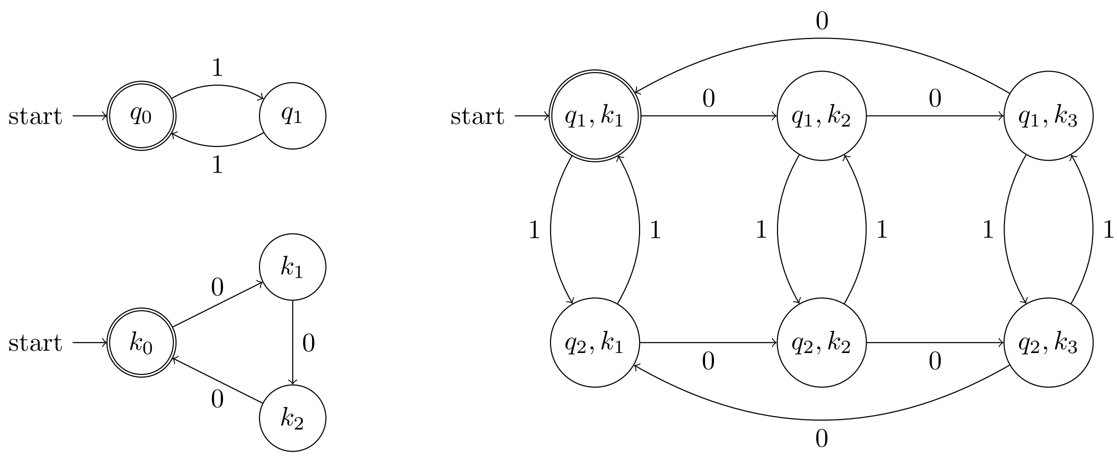


Figure 11: On the left, we have M_1 and M_2, where $L(M_1) = \{w : \text{count of 1s is even}\}$, and $L(M_2) = \{w : \text{count of 0s is divisible by 3}\}$. We build a DFA, shown on the right, that can process the intersection between M_1 and M_2 and name it N. Note that DFAs are a subset of all NFAs, and thus this DFA is an NFA. N processes inputs sequentially, accounting for all possibilities for the transitions over $M_1 \times M_2$.
