COMS 3261, Computer Science Theory (Fall 2023): Assignment 1 Solutions

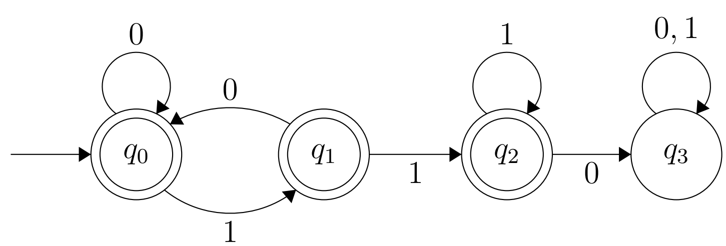
Instructions

- Problems 1-6 are each worth 10 points.
- Submit your solutions in pdf format. Late homeworks will **not** be accepted.
- You can discuss with TAs, the prof, and other students, but please acknowledge them at the beginning of each problem. All of your solutions must be written in your own words.

Problems

1. Give deterministic finite automata accepting the following language. You may describe your DFA with a diagram or a formal description. If you give a diagram be sure to include all transitions, specify the start state, as well as all final states. Try to use the smallest number of states.

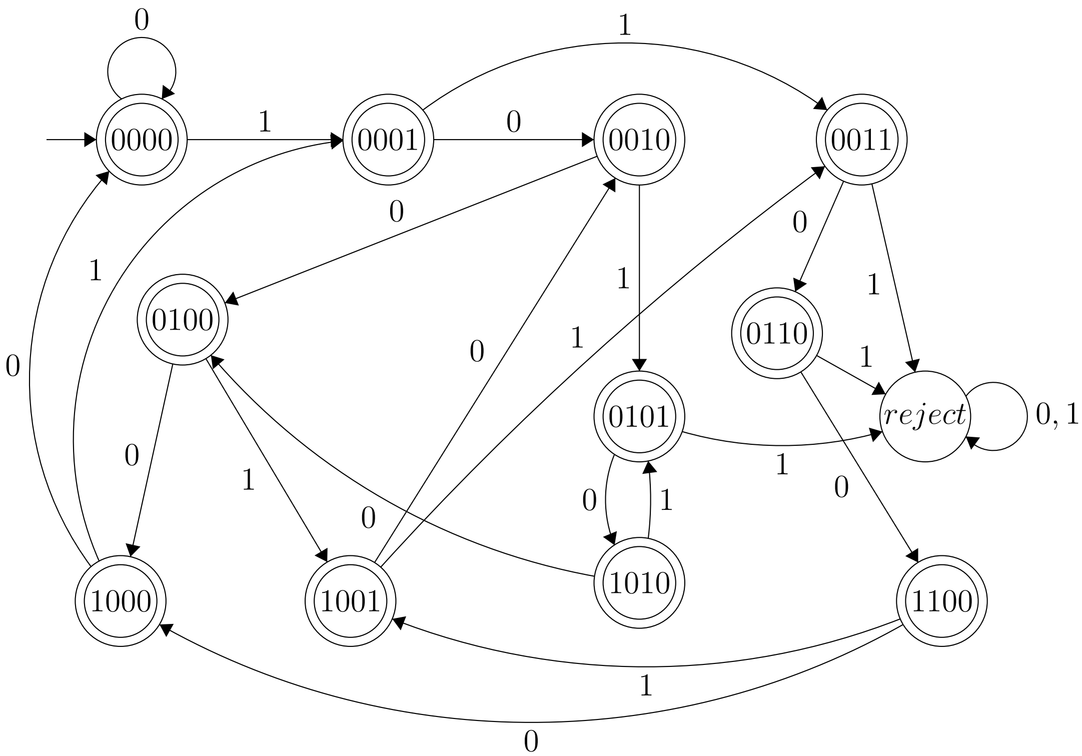
$$\mathcal{L} = \{w \in \{0,1\}^* \mid w \text{ does not contain the substring } 110\}$$

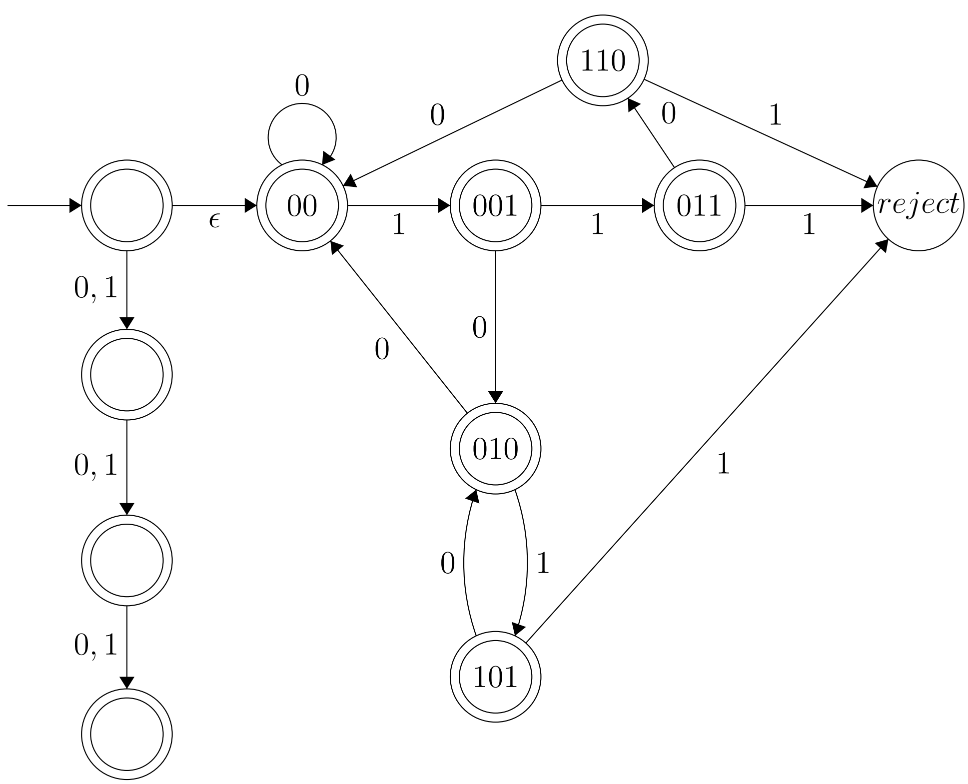


2. Give an NFA accepting the following language. As in the previous question, you can describe your NFA with a diagram or a formal description. Try to use the smallest number of states.

$$\mathcal{L} = \{w \in \{0,1\}^* \mid \text{every 4 consecutive symbols contains at least 2 } 0's \text{ and } |w| \geq 4\}$$

There are several ways to construct the NFA for the language. Below are two possible implementations, where the first is a DFA keeping track of the previous 4 symbols read. Note why the second NFA is correct: we only need to keep track of the previous 3 symbols in the string to determine if the next input symbol leads to the reject state.


0000
0001
0010
0011
0100
0101
0110
1000
1001
1010
1100
reject
0, 1


00
001
011
110
010
101
reject
ϵ
0, 1
0, 1
0, 1

3. If A is any language, let $Half(A)$ be the set of all first halves of strings in A. So x is in $Half(A)$ if there is some string y of the same length as x such that the string xy is in A. Prove that if A is regular, then $Half(A)$ is also regular.

   Let $M = (q_0, \Sigma, Q, \delta, F)$ be a DFA that accepts A. We will describe a nondeterministic automata, N, for accepting $Half(A)$. On input x, N operates as follows. First, N guesses the state q^* that M would be in after it finishes reading x. Then in parallel (to be described later), N simulates M on x starting from state q_0, and N simulates M starting in state q^* on a guessed string y. To carry out this parallel computation nondeterministically (reading only x), the states of N will be triples of states (q_*, q_i, q_j) from Q. Let x be a string, where $|x| = n$. Then after processing x (after n steps), the set of states S_x that N can be in are as follows. A particular state (q_*, q_i, q_j) is in S_x if and only if: (1) M, when run on x beginning in q_0 ends in state q_i; and (2) There exists a string y, such that $|y| = |x|$ and such that M when run on y beginning in state q_*, ends up in state q_j. The final states F' of N will consist of triples (q_1, q_2, q_3) such that q_3 is a final state of M, and $q_1 = q_2$.

   Assume that q_0 is the start state for M which is not a final state. The transition function for N is as follows. Let the start state be (q_0, q_0, q_0). First, nondeterministically transition to (q, q_0, q), for all $q \in Q$. (Here we are guessing that q will be the state that M would end up in after we process the string x.) Now let (q_*, q_i, q_j) be some state, and let the next symbol being read be b. Then (r_*, q_i', q_j') is in $\delta'(q_*, q_i, q_j)$ if and only if: $q_* = r_*$, and $\delta(q_i, b) = q_i'$, and finally, there exists a symbol q such that $\delta(q_j, a) = q_j'$.

4. Prove that the following language is not regular. (Wait for Lectures 5 and 6.)

$$\mathcal{L} = \{w \in \{0,1\}^* \mid |w| \text{ is odd and the middle bit of w is a } 1\}$$

   Suppose for sake of contradiction that $\mathcal{L}$ defined above is a regular language. Thus there must exist a DFA, M, such that the language accepted by M is $\mathcal{L}$. Suppose that M has k states.

   Consider the string $w = 0^k10^k \in \mathcal{L}$. Because M only has k states, on input w, it must get into a loop (repeat states) while reading the string, as w is of length greater than k.

   We partition w into $w = xyz$, where $x = \epsilon$, $y = 0^i$, and $z = 0^{k-i}10^k$, where $i \geq 1$. By the pumping lemma, if we pump up (repeat the substring y), $xy^2z = 0^{k+i}10^k$ should also be in the language $\mathcal{L}$ if $\mathcal{L}$ was regular. But since the middle bit of x^2y is not 1, the pumped string is not in L. We have reached a contradiction. Therefore, $\mathcal{L}$ is not a regular language.

5. For a string w, let w^R denote the string w in reverse order. Is the following language regular? Prove or disprove your answer.

$$\mathcal{L} = \{w \in \{0,1\}^* \mid w \neq w^R\}$$

No, the language is not regular.

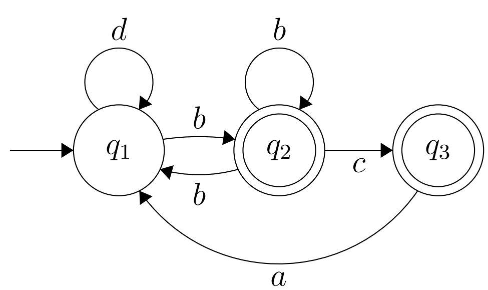
Since regular languages are closed under the complement operation, if L is regular, then it must be the case that $\overline{\mathcal{L}}$ is also regular, where

$$\overline{\mathcal{L}} = \{w \in \{0,1\}^* \mid w = w^R\}$$

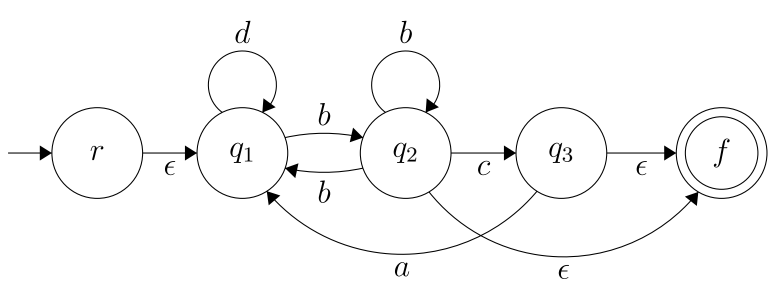
We will show that $\overline{\mathcal{L}}$ is not regular through proof by contradiction. Let M be a k-state DFA accepting $\overline{\mathcal{L}}$. Consider the string $w = 0^k10^k \in \overline{\mathcal{L}}$. Because M only has k states, it must loop (repeat states) on input w, as w is of length greater than k.

By the pumping lemma, there must be a way to partition w into $w = xyz$, where $|y| > 0$ and $|xy| \leq k$, such that for any $i \geq 0$, $xy^iz \in \mathcal{L}$. However, note that no matter how we partition the string, y will only contains 0's; and thus when we pump up, e.g. when $i = 2$, the string will no longer be in $\overline{\mathcal{L}}$ since there are now more 0's before the 1 than after the 1, and thus $w \neq w^R$. This raises a contradiction and the language does not satisfy the pumping lemma; thus, $\overline{\mathcal{L}}$ is not regular. Since regular languages are closed under the complement operation, it must be the case that $\mathcal{L}$ is also not regular.

6. Given the following NFA N, using the procedure described in class (and in the handout), construct the regular expression accepted by N. You may show some intermediate steps for partial credit but please keep the total number of steps to at most 4.

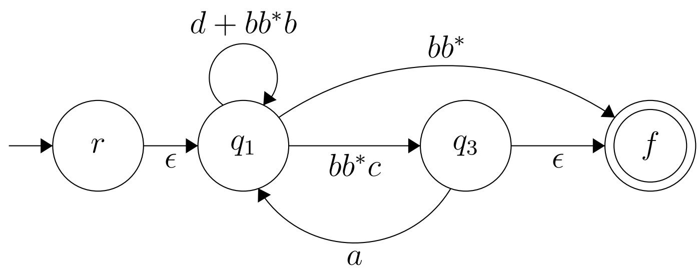


Step 1: Add a new start and accept state.



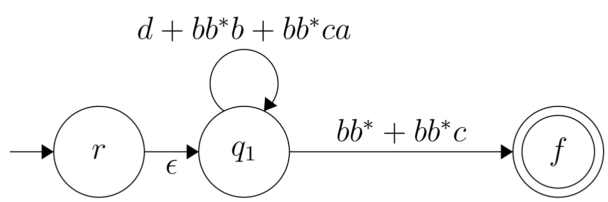
Step 2: Eliminate q_2.

- $q_1 \to q_2 \to q_1 = bb^*b$
- $q_1 \to q_2 \to q_3 = bb^*c$
- $q_1 \to q_2 \to f = bb^*$



Step 3: Eliminate q_3.

- $q_1 \to q_3 \to f = bb^*c$
- $q_1 \to q_3 \to q_1 = bb^*ca$



Step 4: Eliminate q_1.

- $r \to q_1 \to f$