COMP 2805 — Solutions Assignment 1

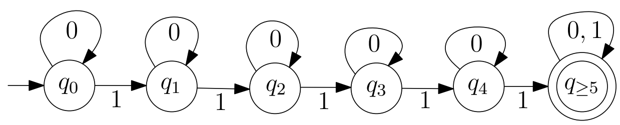
Question 1: For each of the following languages, construct a DFA that accepts the language. In all cases, the alphabet is $\{0, 1\}$.

(1.1) $\{w : w$ contains at least five 1s$\}$.

Solution: We will use six states:

- q_0: no 1 has been read
- q_1: exactly one 1 has been read
- q_2: exactly two 1s have been read
- q_3: exactly three 1s have been read
- q_4: exactly four 1s have been read
- $q_{\geq 5}$: five or more 1s have been read

The start state is q_0; there is one accept state: $q_{\geq 5}$. Here is the state diagram:

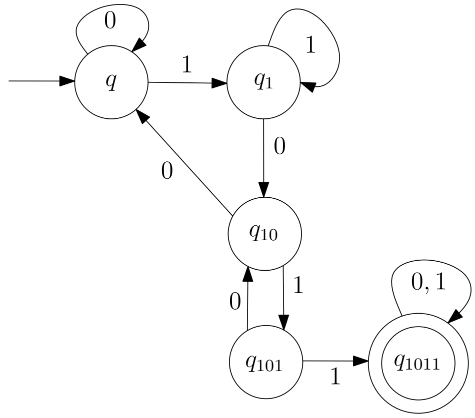


(1.2) $\{w : w$ contains the substring 1011, i.e., $w = x1011y$ for some strings x and $y\}$.

Solution: We will use five states:

- q: start state
- q_1: we have just seen 1 and hope that the next three symbols are 011.
- q_{10}: we have just seen 10 and hope that the next two symbols are 11.
- q_{101}: we have just seen 101 and hope that the next symbol is 1.
- q_{1011}: the input string contains 1011.

The start state is q; there is one accept state: q_{1011}. Here is the state diagram:

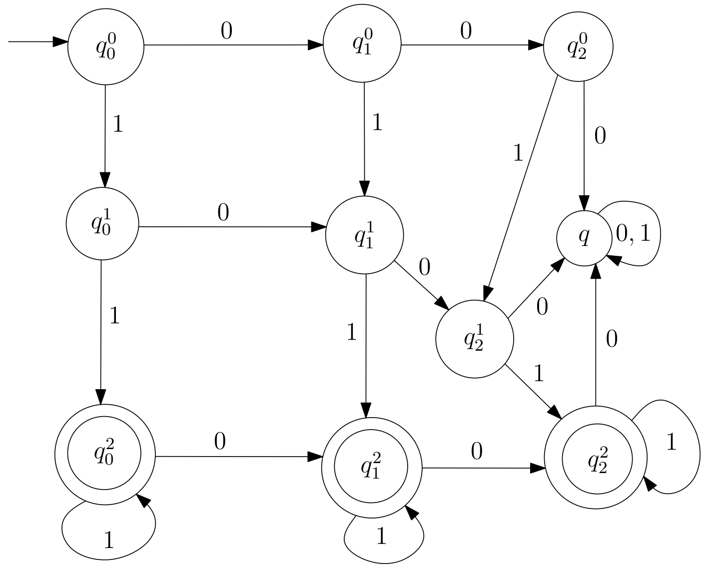


(1.3) $\{w : w$ contains at least two 1s and at most two 0s$\}$.

Solution: We will use ten states:

- q_0^0: we have seen zero 0s and zero 1s
- q_0^1: we have seen zero 0s and one 1
- q_0^2: we have seen zero 0s and at least two 1s
- q_1^0: we have seen one 0 and zero 1s
- q_1^1: we have seen one 0 and one 1
- q_1^2: we have seen one 0 and at least two 1s
- q_2^0: we have seen two 0s and zero 1s
- q_2^1: we have seen two 0s and one 1
- q_2^2: we have seen two 0s and at least two 1s
- q: we have seen at least three 0s

The start state is q_0^0; there are three accept states: q_0^2, q_1^2, and q_2^2. Here is the state diagram:

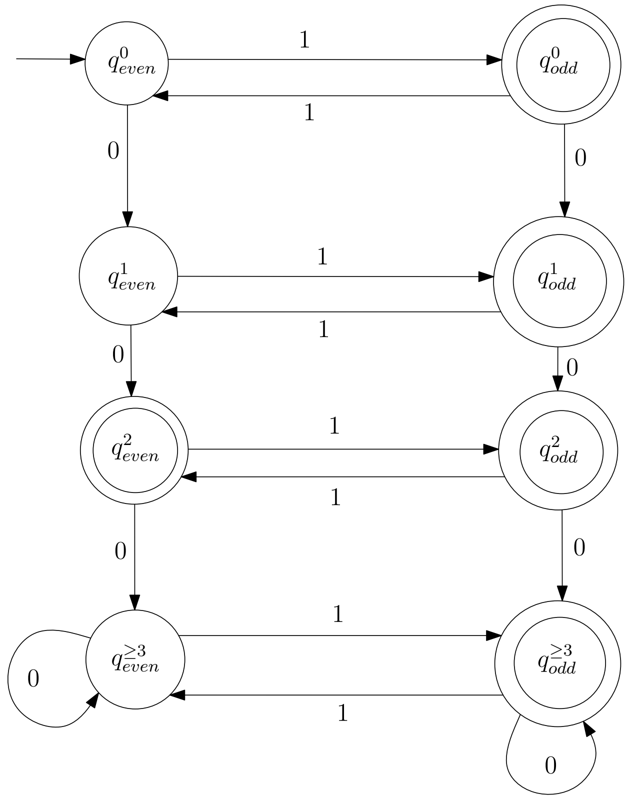


(1.4) $\{w : w \text{ contains an odd number of 1s or exactly two 0s}\}$.

Solution: We will use eight states; each state records

- whether we have seen an even or odd number of 1s (given by the subscript) **and**
- whether we have seen zero, one, two, or at least three 0s (given by the superscript).

Here is the state diagram:



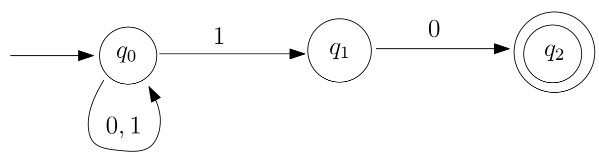
Question 2: For each of the following languages, construct an NFA, with the specified number of states, that accepts the language. In all cases, the alphabet is $\{0, 1\}$.

(2.1) The language $\{w : w \text{ ends with } 10\}$ with three states.

Solution: We will use the following states:

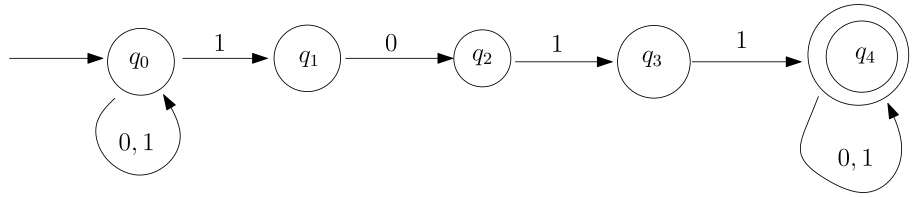
- q_0: we have not reached the rightmost two symbols in the input string.
- q_1: we have read the entire string except for the rightmost symbol (which we hope is 0); the last symbol read was 1.
- q_2: we have read the entire string and the last two symbols were 10.

The start state is q_0; there is one accept state: q_2. Here is the state diagram:



(2.2) The language $\{w : w \text{ contains the substring } 1011\}$ with five states.

Solution: Here is the state diagram:



(2.3) The language $\{w : w$ contains an odd number of 1s or exactly two 0s$\}$ with six states.

Solution: First, we give a DFA (which is also an NFA) with two states that accepts all strings having an odd number of 1s. Then, we give an NFA with three states that accepts all strings having exactly two 0s. Finally, we apply the union construction to these NFA's.

Here is the DFA with two states that accepts all strings having an odd number of 1s. It has states

- q_0: we have seen an even number of 1s.
- q_1: we have seen an odd number of 1s.

q_0 is the start state, whereas q_1 is the accept state. Here are the transitions:

- When in state q_0 and reading 1: go to state q_1
- When in state q_0 and reading 0: go to state q_0
- When in state q_1 and reading 1: go to state q_0
- When in state q_1 and reading 0: go to state q_1

Here is the NFA with three states that accepts all strings having exactly two 0s. It has states

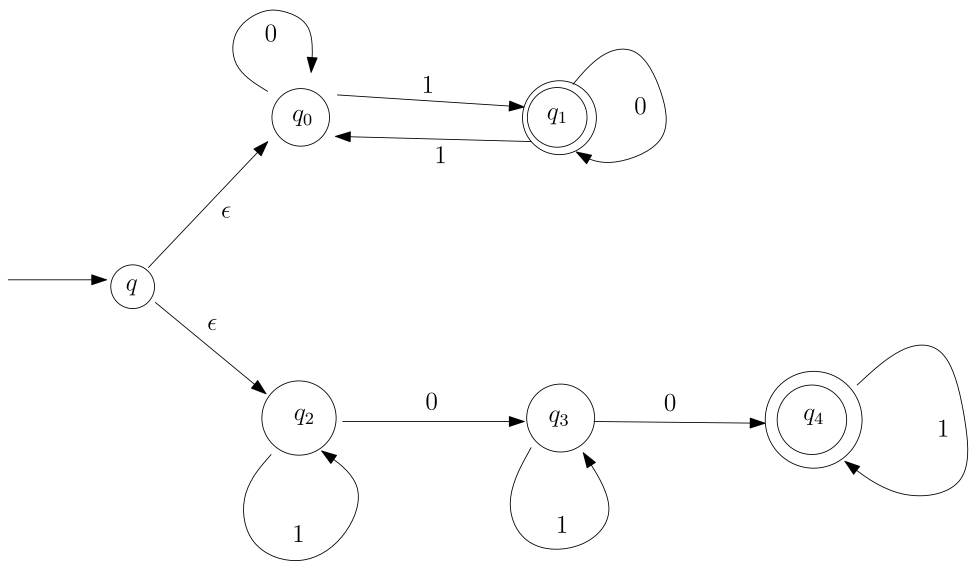
- q_2: we have not seen a 0 yet.
- q_3: we have seen exactly one 0.
- q_4: we have seen exactly two 0s.

q_2 is the start state, q_4 is the accept state. Here are the transitions:

- When in state q_2 and reading 1: go to state q_2
- When in state q_2 and reading 0: go to state q_3
- When in state q_3 and reading 1: go to state q_3
- When in state q_3 and reading 0: go to state q_4
- When in state q_4 and reading 1: go to state q_4

This automaton is nondeterministic, so there are no other arrows.

We now draw the state diagrams of these two automata; then we apply the union construction to them: Add a new start state q and give it ϵ-arrows to the start states of the two automata. Here is the result:



Question 3: Let A be a regular language. Prove that there exists an NFA that accepts A and that has exactly one accept state. (*Hint:* There exists a DFA/NFA that accepts A. If this automaton has more than one accept state, modify it.)

Solution: Let $M = (Q, \Sigma, \delta, q, F)$ be a DFA that accepts A. If $|F| = 1$, then we are done, so we assume that F contains at least two states. We construct an NFA N that accepts A and that has exactly one accept state:

1. We make a copy of M.
2. We make each accept state of M a non-accept state of N.
3. We create a new state q', which will be the only accept state of N.
4. We add an ϵ-transition from every state of F to q'.
5. There are no transitions from q' to any state of N.

Formally, we define the NFA $N = (Q', \Sigma, \delta', q, F')$, where

- $Q' = Q \cup \{q'\}$,
- the start state q of N is the start state of M,
- $F' = \{q'\}$,

- for every $r \in Q'$ and every $a \in \Sigma_\epsilon$,

$$\delta'(r,a) = \begin{cases} \{\delta(r,a)\} & \text{if } r \in Q \text{ and } a \neq \epsilon, \\ \emptyset & \text{if } r \in Q \setminus F \text{ and } a = \epsilon, \\ \{q'\} & \text{if } r \in F \text{ and } a = \epsilon, \\ \emptyset & \text{if } r = q'. \end{cases}$$

The careful reader will ask whether this proof also works if the language A is empty. If $A = \emptyset$, then the set F of accept states of M could be empty as well. In this case, however, the above proof is correct.

Question 4: For any string $w = w_1 w_2 \dots w_n$, we denote by w^R the string obtained by reading w backwards, i.e., $w^R = w_n w_{n-1} \dots w_2 w_1$. For any language A, we define A^R to be the language obtained by reading all strings in A backwards, i.e.,

$$A^R = \{w^R : w \in A\}.$$

Let A be a regular language. Prove that the language A^R is also regular. (*Hint:* Use the previous question.)

Solution: Let $M = (Q, \Sigma, \delta, q, F)$ be an NFA that accepts A. By the previous question, we may assume that F consists of exactly one state, say $F = \{q'\}$.

If $w \in A$, then in the state diagram of M, there exists a path from q to q', such that by following this path, we read the string w. Observe that by traversing this path *backwards*, we read the string w^R.

We obtain an NFA for the language A^R, by

1. making a copy of M,
2. making q' the start state,
3. making q the accept state,
4. reversing all arrows in the state diagram of M.

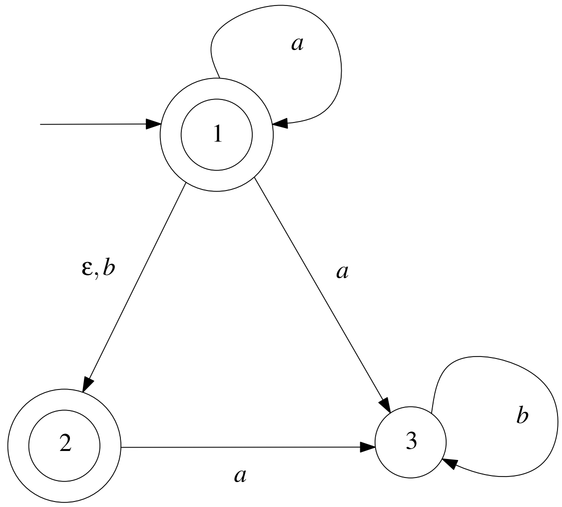
Formally, we define $N = (Q, \Sigma, \delta', q', F')$, where

- Q is the set of states of M,
- the start state q' of N is the accept state of M,
- $F' = \{q\}$,
- for every $r \in Q$ and every $a \in \Sigma_\epsilon$,

$$\delta'(r,a) = \{r' \in Q : r \in \delta(r', a)\}.$$

You should convince yourself that this construction also works if the language A is empty.

Question 5: Use the construction given in class (and described in the notes) to convert the following NFA to an equivalent DFA.



Solution: Following the construction given in class, the DFA has the following eight states:

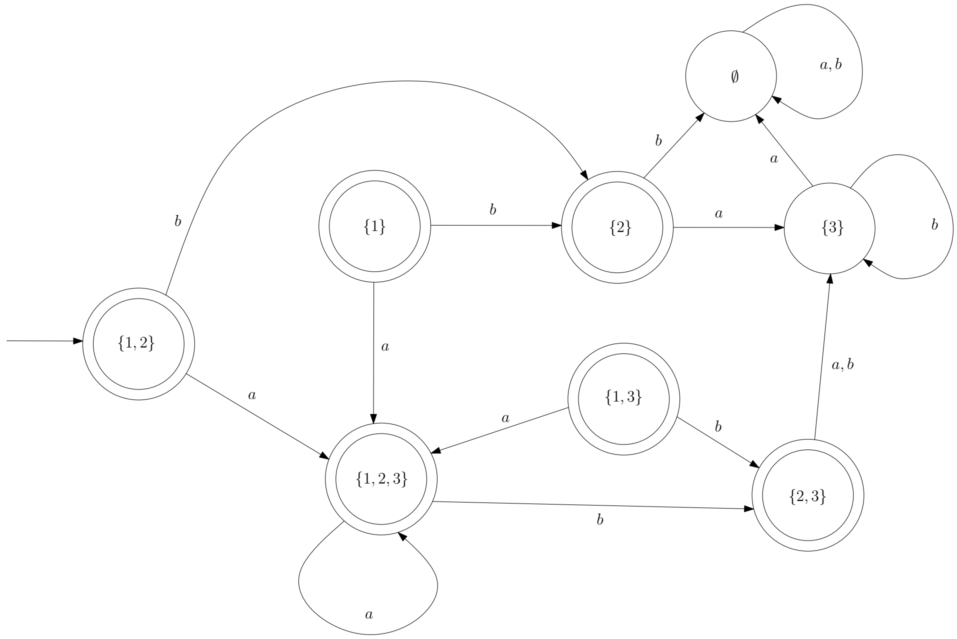
$$\emptyset, \{1\}, \{2\}, \{3\}, \{1,2\}, \{1,3\}, \{2,3\}, \{1,2,3\}.$$

The start state of the DFA is the set of all states of the NFA that can be reached from the NFA's start state 1 by making zero or more ϵ-transitions. Hence, the start state of the DFA is $\{1,2\}$.

The set of accept states of the DFA consists of all states of the DFA that contain at least one accept state of the NFA. That is, the set of accept states of the DFA consists of all states of the DFA that contain 1 or 2. This gives the following six accept states:

$$\{1\}, \{2\}, \{1,2\}, \{1,3\}, \{2,3\}, \{1,2,3\}.$$

The transition function of the DFA is specified in the following state diagram:



The NFA above is correct, but, since the state $\{1, 3\}$ cannot be reached from the start state $\{1, 2\}$, it can be removed. Also, since the state $\{1\}$ cannot be reached from the start state $\{1, 2\}$, it can be removed. This gives the final DFA: