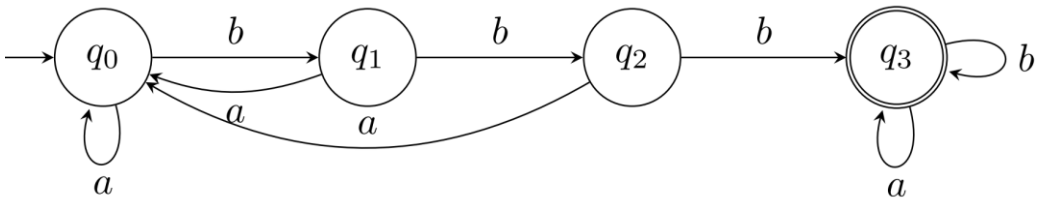
COSC 3106: Practice Problem Set#1 - Solutions

Problem 1: What language is recognized by the following DFA over the alphabet $\{a, b\}$? Provide justification for your answer.



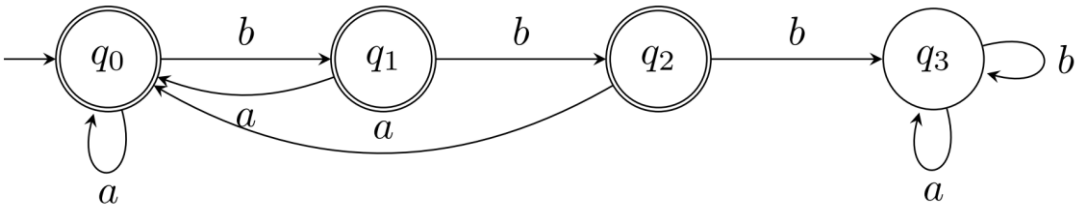
Solution

Language: All strings over $\{a, b\}$ that contain three consecutive b's (i.e., the substring bbb).

- Reading a from q_0, q_1, q_2 sends you to q_0: any a reset the count of consecutive b's.
- Reading b advances $q_0 \to q_1 \to q_2 \to q_3$.
- Once q_3 is reached, the machine stays in q_3 on any further input.
- Hence the recognized language is

$$L = \{\, w \in \{a, b\} \mid bbb \text{ occurs in } w \,\}.$$

Problem 2: Construct a new DFA that accepts its complement? As always, verify your answer.



Solution

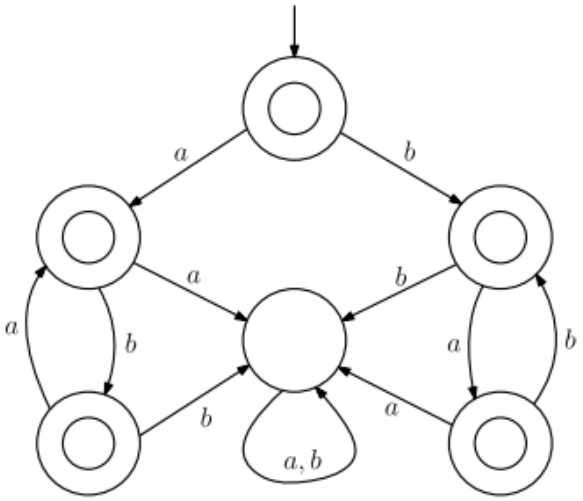
Language: All strings over $\{a, b\}$ that do not contain three consecutive bbb's.

$$L = \{w \in \{a, b\} *| \; bbb \text{ is not a substring of } w\}.$$

- The DFA's states q_0, q_1, q_2 (all accepting) track how many consecutive trailing b's have just been seen: 0, 1, or 2, respectively.
- Reading a from q_1 or q_2 sends you back to q_0 (resets the run of b's).
- Reading b advances $q_0 \to q_1 \to q_2 \to q_3$.

- q_3 is a non-accepting sink (self-loops on a, b), reached iff the input has some bbb. Thus the automaton accepts exactly those strings with no bbb.

Problem 3: Define the language of the following DFA?



Solution

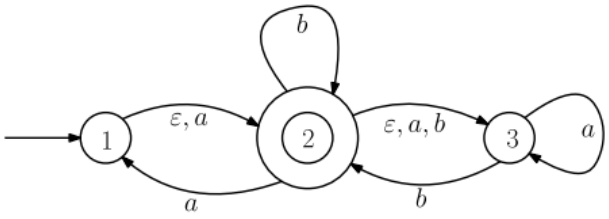
Let us define the non-accept state as q. The following three claims follow from the state diagram.

- If we reach the state q, then we will stay there forever.
- If we are in any accept state: If we read aa or bb, then we will reach state q.
- The only way to reach state q is by reading aa or bb.

This means that a string will be rejected if and only if it contains either the substring aa or bb. Therefore, the DFA accepts exactly the complement of those strings. In other words, it accepts all strings in which a's and b's strictly alternate. the DFA accepts:

- The empty string ε,
- a single letter (a or b),
- any alternating sequence of the form $(ab)^k$ for $k \geq 1$,
- alternating sequences ending in a, i.e. $(ab)^k a$ for $k \geq 1$,
- alternating sequences of the form $(ba)^k$ for $k \geq 1$.
- alternating sequences ending in b, i.e. $(ba)^k b$ for $k \geq 1$,

Problem 4: Apply the method discussed in class to transform the given NFA (over the alphabet $\{a, b\}$) into an equivalent DFA.



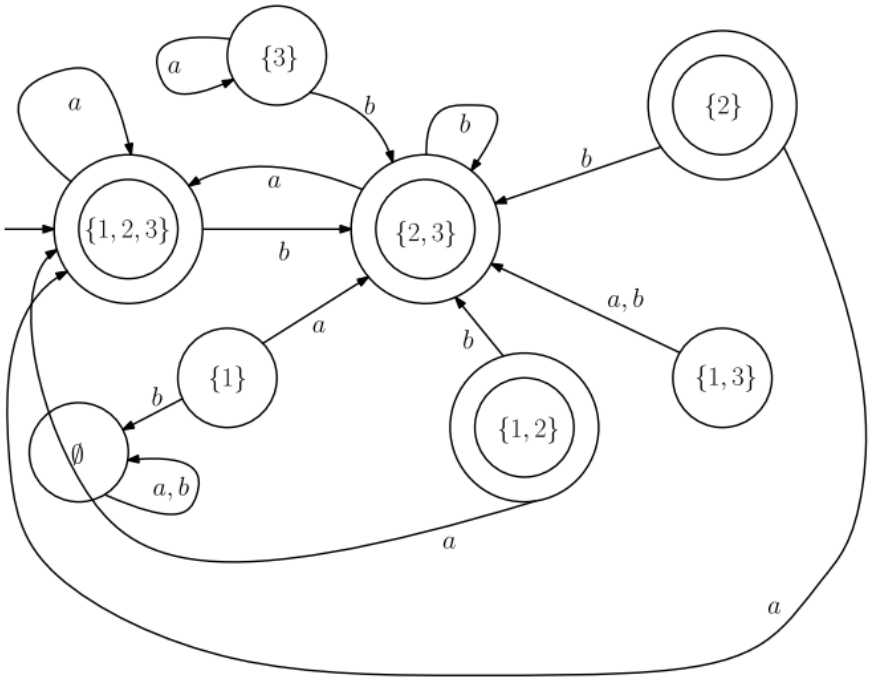
Draw the complete state diagram of the DFA. Since the NFA has 3 states, the equivalent DFA will have up to $2^3 = 8$ states. After drawing the full diagram, simplify it by eliminating any states that are not reachable from the start state.

Solution

Each state of the DFA corresponds to a subset of $\{1, 2, 3\}$. Thus, the DFA has $2^3 = 8$ states. The accept states of the DFA are all states that contain the accept state 2 of the NFA. Thus, the DFA has 4 accept states.

What is the start state of the DFA: We take the start state 1 of the NFA and add to it all states that can be reached by following zero or more ε-transitions.

Note that state 2 can be reached by following one ε-transition from state 1. Also, state 3 can be reached by following two ε-transitions from state 1. Thus, the start state of the DFA is $\{1, 2, 3\}$. Now we need the transitions of the DFA. Remember, to simulate one step of the NFA, the DFA reads one symbol (a or b) and then follows zero or more ε-transitions. The state diagram of this DFA:



In this diagram, the states $\emptyset, \{1\}, \{2\}, \{3\}, \{1, 2\}$, and $\{1, 3\}$ cannot be reached from the start state $\{1, 2, 3\}$. Thus, we can remove these states, which leads to the final DFA: