The Minimization of FSA

CHAPTER 4

Why?

- The fewer states an automaton has, the less time it will take to recognize a word.
- It will also take up less memory space if it needs to be saved.
- It is therefore logical to want to minimize this time by trying to reduce the number of states. While we can find many automatons to recognize the same language, we can only find one minimal automaton that recognizes the same language.
- How? Minimization is achieved by eliminating so-called unreachable states and by merging (or combining) states that recognize the same language

Definition

A deterministic finite state automaton (DFA)is minimal if and only if any other DFA recognizing the same language has at least the same number of states

Unreachable states

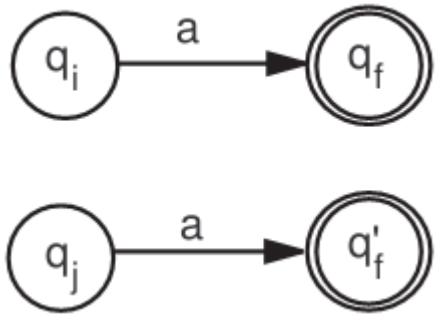
A state is said to be unreachable (non-accessible) if there is no path to reach it from the initial state.

Unreachable states are unproductive, meaning they will never participate in the recognition of a word. Thus, the first step in minimizing a DFA is to eliminate these states

β-equivalent states

Two states q_i and q_j are said to be β-equivalent if they allow reaching the final states using the same words. We write: qi β qj

Thus, the minimization algorithm simply consists of merging these states into one



Note

The β-equivalence relation is called a congruence relation.

The number of equivalence classes of the β-equivalence relation is equal to the number of states in the minimal automaton because the states of each equivalence class recognize the same language (they will be merged)

Minimize a DFA

The method for reducing a DFA is as follows:

1. Clean the automaton by eliminating the unreachable states;
2. Group congruent states (those belonging to the same equivalence class).

Algorithm: Grouping of congruent states

Note: This algorithm can only be applied if the automaton is deterministic

1. Create two classes: A containing the final states and B containing the non-final states;
2. If there exists a symbol aaa and two states qi and qj from the same class such that δ(qi,a) and δ(qj,a) do not belong to the same class, then create a new class and separate qi and qj. Leave in the same class all states that lead to the same class;
3. Repeat step 2 until there are no more classes to separate.

Algorithm

The parameters of the minimal automaton are as follows:

- Each congruence class is a state of the minimal automaton;
- The class that contains the old initial state becomes the initial state of the minimal automaton;
- Any class containing a final state becomes a final state;
- The transition function is defined as follows: let A be a congruence class obtained, a be a symbol from the alphabet, and q_i be a state $q_i \in A$ such that $\delta(q_i, a)$ is defined. The transition $\delta(A, a)$ is equal to the class B that contains the state q_j such that $\delta(q_i, a) = q_j$.

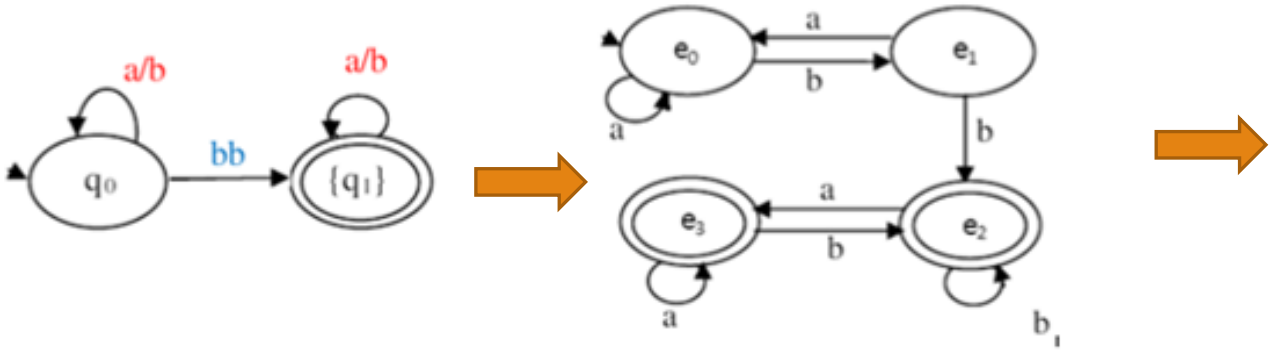
Example 1

$A = (\Sigma, Q, q_0, \delta, F)$ be such that $Q = \{q_0, q_1\}, F = \{q_1\}$, and:

$\delta : \delta(q_0, a) = q_0, \delta(q_0, b) = q_0, \delta(q_0, bb) = q_0, \delta(q_1, a) = q_1, \delta(q_1, b) = q_1$

1. Draw the DFA A
2. Find the minimal automaton A' equivalent to A

Example 1

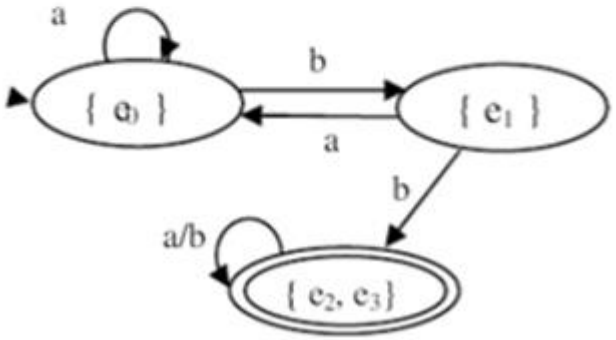


β_0 : { e_0, e_1 } { e_2, e_3 }

β_1 : { e_0 } { e_1 } { e_2, e_3 }

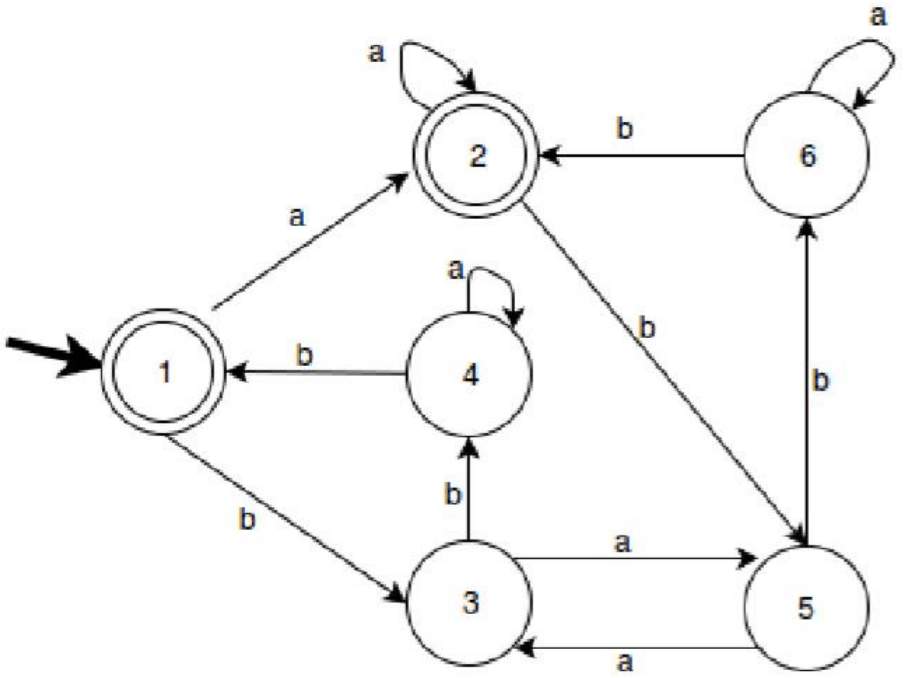
β_2 : { e_0 } { e_1 } { e_2, e_3 }

stop $\beta_1 = \beta_2$



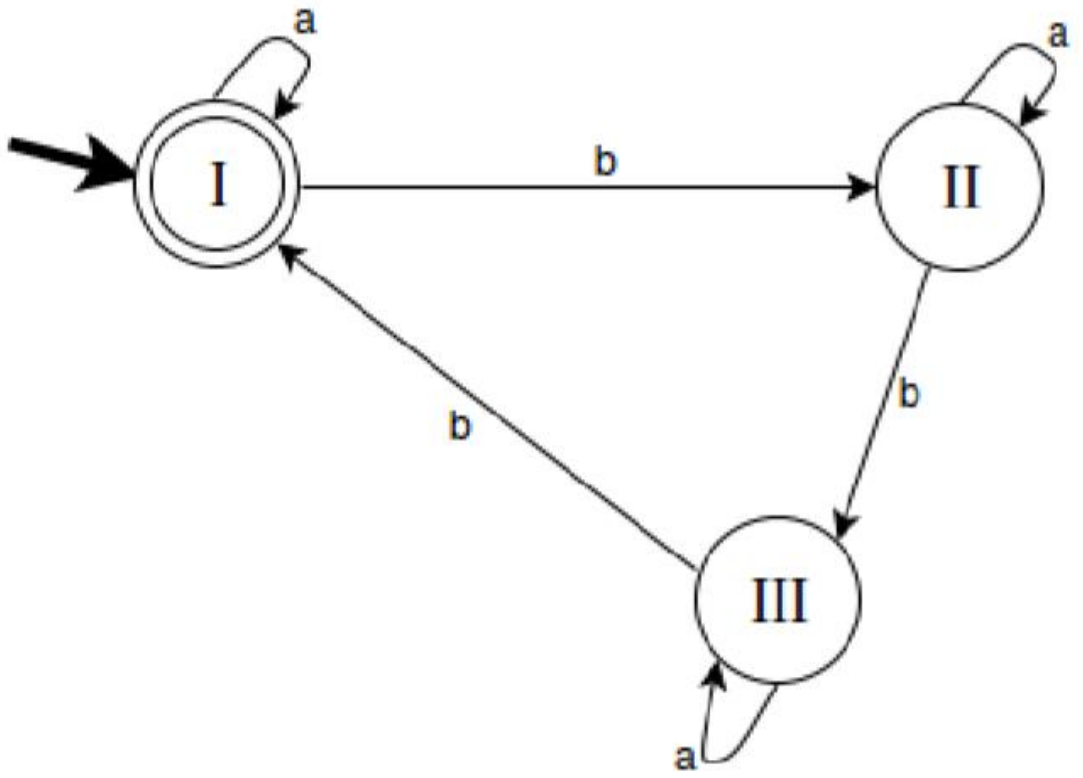
Example 2

Find the minimal automaton A' equivalent to A



Example 2

$$I = \{1,2\}, II = \{3,5\}, III = \{4,6\}$$



Example 3

Let the following automaton be minimized (the final states are states 1 and 2, while state 1 is the initial state):

	a	b
1	2	5
2	2	4
3	3	2
4	5	3
5	4	6
6	6	1
7	5	7

Example 3

Step 1:

A= {1,2}

B = {3,4,5,6}

Step 2:

A= {1,2}

B = {4,5,6}

C = {3}

Step 3:

A = {1,2}

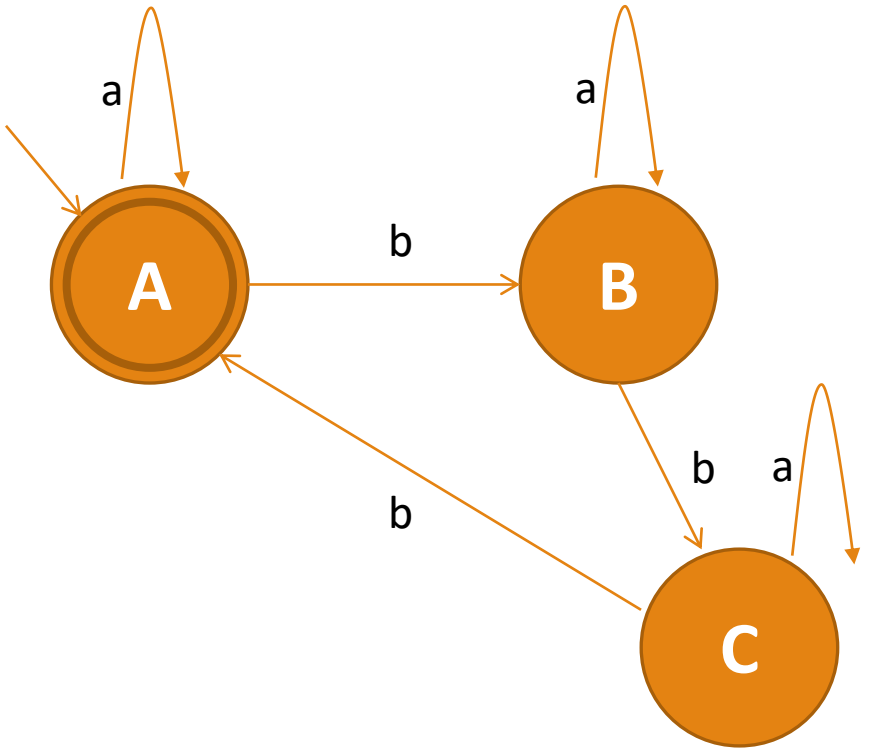
B = {4,5}

C = {3,6}

Stop

État	a	b
1	2	5
2	2	4
3	3	2
4	5	3
5	4	6
6	6	1
7	5	7

Example 3



Thank you!