Finite Automata

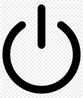
Prof. Shrisha Rao
IIIT Bangalore

srao@iiitb.ac.in

2025-08-05

Finite Automata

- An *automaton* is an abstraction of a device, without regard to its technology or internal workings. The plural of automaton is *automata*.
- The simplest non-trivial automaton is an ON-OFF switch:



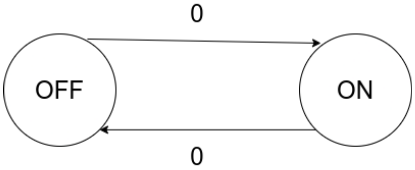
- The ON-OFF switch has exactly two *states*, namely ON and OFF.
- In general, a finite automaton (FA) has a finite number of states. The set of states of a FA is denoted by uppercase Q.

Input Symbols

- A FA also receives an input of some type. For instance, an ON-OFF switch is typically activated by a finger push.
- The input to a FA is denoted by a symbol such as $0, 1, a, b$. The numerical values or other meanings of such symbols are not pertinent; they are to be interpreted just as symbols that denote some inputs to a FA.
- For instance, the "finger push" input to an ON-OFF switch may be denoted by the symbol 0 (or equivalently, 1, a, etc.).
- The set of symbols accepted by a FA is called the *alphabet*, denoted by Σ.

Denoting a FA

- A FA is denoted with a "bubble diagram" with states denoted by circles, and transitions between states denoted by arrows. The labels on the arrows indicate the input symbols.
- For instance, the FA for an ON-OFF switch may be denoted as follows:



- In general, the states of a FA are denoted by uppercase letters, as A, B, C, etc. They can also be denoted by q_i, q_2, etc.

Transition Function

- So, a FA has a set of states Q and a set of input symbols Σ.
- It also has a *transition function* denoted by δ, and given by

$$\delta : Q \times \Sigma \longrightarrow Q.$$

- In other words, δ is given by $\delta(q_i, a) = q_j$, where $q_i, q_j \in Q$, and $a \in \Sigma$.
- For instance, with the ON-OFF switch, we have $\delta(\mathrm{ON}, 0) = \mathrm{OFF}$, and $\delta(\mathrm{OFF}, 0) = \mathrm{ON}$.

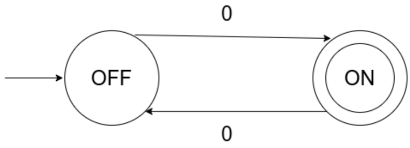
Denoting a FA as a 5-tuple

A FA can thus also be denoted as a 5-tuple $\langle Q, \Sigma, \delta, q_0, F \rangle$, where:

- Q is the set of states of the FA;
- Σ is the alphabet of the FA;
- $\delta : Q \times \Sigma \longrightarrow Q$ is the transition function of the FA;
- $q_0 \in Q$ is a distinguished state of the FA, called its *start state*; and
- $F \subseteq Q$ is the set of *accept states* of the FA.

Denoting Start State and Accept States

- In a bubble diagram, a start state (which is often shown left-most) is denoted by an inward arrow.
- Accept states are denoted by concentric circles (instead of a single circle).
- For instance, an ON-OFF switch where the switch is supposed to start in the OFF position, and where ON is the accept state, may be denoted as follows:



Strings and Accepted Strings

- A sequence of input symbols given to an automaton is called a *string*. Strings may be denoted as $01100, 01^2 0^2$, *abc*, etc., depending on the alphabet.
- A string is said to be *accepted* by a FA if it ends in an accept state when started from its start state and given that string of symbols in sequence.
- For instance, the ON-OFF switch started from the OFF state, with the ON state being the accept state, accepts all strings with an odd number of 0s.
- The set of strings accepted by a FA is called the *language* of the FA. The language of a FA is typically infinite.

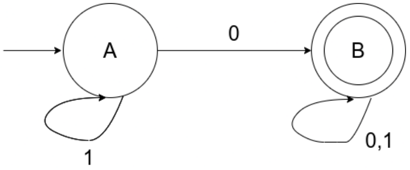
Constructing a FA

A typical exercise in this regard is to construct a FA that accepts a specified language.

(1) Construct a FA that accepts all binary strings that contain at least one 0.

A FA that accepts strings with at least one 0

Solution:



Left for you to do: give the 5-tuple specification of this FA.

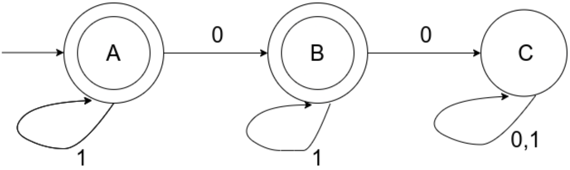
Construct other FAs

(2) Construct a FA that accepts all binary strings that contain at most one 0.

(3) Construct an FA that accepts all non-zero binary numbers that are multiples of 4.

FA that accepts strings with at most one 0

Solution:



Again, give the 5-tuple specification of this FA.

Further Exercises

For all of these, give a bubble diagram representation as well as writing down the appropriate 5-tuple specification.

(4) Construct a FA that accepts any string with an odd number of 1s.

(5) Construct a FA that accepts all strings over $\{a, b, c\}$ that contain an odd number of a's.

(6) Construct a FA that accepts only strings over $\{a, b, c\}$ where the progression of the symbols is in reverse alphabetical order. In other words, *cba*, *ccbbaaa*, *ba*, etc., are all to be accepted, but *aac*, *aabccab*, *aabbaccc*, etc., are not.

(7) Construct a FA that accepts all non-empty binary strings where the number of 0s is odd, and the number of 1s is a multiple of 3.