Theory of Automata
Finite Automata

Hafiz Tayyeb Javed

Section B Week-4 Lecture 01

Section A Week-03-Lecture-02

Contents

Finite Automata (FA) - Introduction

- Finite State Automata (FSA) or Finite State Machine (FSM)
  - An FA is a model of a system with discrete inputs and outputs
  - The system can be in any ONE of a finite number of the internal configurations or States.
  - The states of the system summarizes the information concerning the past inputs that is needed to determine the behavior of the system on subsequent inputs.
  - The system is changing the state as a result of new inputs.

Examples

- ON/OFF switch
- Control mechanism of an Elevator
  - It has finite set of states, the floors, to move to
  - It has finite set of inputs,
    - the call buttons on each floor and
    - the floor buttons in the carriage
  - The system only knows the current state/floor it is on
  - The after receiving an input the control determines the next state/floor to move to and the direction of the movement in relation to the current floor.

A News Paper wending Machine

  - Input to machine consists of Nickels(5), Dimes(10) and Quarters(25)
  - When 30 cents are inserted, the cover may be opened and a news paper removed
  - If total of coins exceed 30 cents, the machine accepts the over payment and does not give change.
  - Machine has no additional memory, however it knows that an additional 5 cents will unlatch the cover when 25 cents has previously been inserted.
- What is the language of the machine?
- All strings of *n, d, q* representing sum of 30 cents or more
- What are possible states?
- Need 30 cents, needs 25 cents, ----, needs 5 cents, needs 0 cent, to open the latch
- What is the final state?
- need 0 cents

News Paper Wending Machine FA Model

The Model

- $\Sigma = \{n, d , q\}$
- Q = { 0, 5,10, 15, 20, 25, 30)
- F = { 0 }
- S = { 30 }

δ	n	d	q
0	0	0	0
5	0	0	0
10	5	0	0
15	10	5	0
20	15	10	0
25	20	15	0
30	25	20	5

Definition

A **finite automaton** is a collection of three things:

1. A finite set of states, **one** of which is designated as the initial state, called the **start state**, and **some** (maybe **none**) of which are designated as **final states**.

2. An **alphabet Σ** of possible input letters.

3. A finite set of **transitions** that tell for each state and for each letter of the input alphabet which state to go next.

How Does a Finite Automaton work?

- Finite Automaton (FA) works by being presented with an input string of letters that it reads letter by letter starting from the leftmost letter of the input;
- Beginning at the start state, the letters read from input determine a sequence of states and guide the movement of the control along the path in the FA.
- This sequence of states ends when the last input letter has been read and the movement of the control stops.
- If the current state in which control happen to be one of the final states, the input is accepted as valid string from the language of the FA.

Example

Consider the following FA:

- The input alphabet has only the two letters a and b. (We usually use this alphabet throughout the chapter.)

- There are only three states, *x*, *y* and *z*, where *x* is the start state and *z* is the final state.

- The transition list for this FA is as follows:
  - Rule 1: From state *x* and input *a*, go to state y.
  - Rule 2: From state *x* and input *b*, go to state *z*.
  - Rule 3: From state *y* and input *a*, go to state *x*.
  - Rule 4: From state *y* and input *b*, go to state *z*.
  - Rule 5: From state *z* and any input, stay at state *z*.

Example Contd.

- Let us examine what happens when the input string ‘*aaa*’ is presented to this FA.

- First input a: state $x \rightarrow y$ by Rule 1.

- Second input *a*: state $y \rightarrow x$ by Rule 3.

- Third input *a*: state $x \rightarrow y$ by Rule 1.

- We did not finish up in the final state z, and therefore have an unsuccessful termination.

Example contd.

- The set of all strings that lead to a final state is called the language defined by the finite automaton.
- Thus, the string *aaa* is not in the language defined by this FA.
- We may also say that the string aaa is not accepted by this FA, or the string *aaa* is rejected by this FA.
- The set of all strings accepted is also called the language associated with the FA.
- We also say, “This FA accepts the language L”, or “L is the language accepted by this FA”, or “L is the language of the FA”, by which we mean that all the words in L are accepted, and all the inputs accepted are words in L

Example contd.

- It is not difficult to find the language accepted by this FA.
- If an input string is made up of only letter a's then the action of the FA will be to jump back and forth between state x and state y.
- To get to state z, it is necessary for the string to have the letter b in it. As soon as a b is encountered, the FA jumps to state z. Once in state z, it is impossible to leave. When the input string runs out, the FA will be in the final state z.
- This FA will accept all strings that have the letter b in them. Hence, the language accepted by this FA is defined by the regular expression

  $$(a + b)^*b(a + b)^*$$

Abstract/Formal definition of FA

1. A finite set of states Q = {q_0, q_1, q_2 q_3 …} of which q_0 is start state.

2. A subset of Q called final state (s).

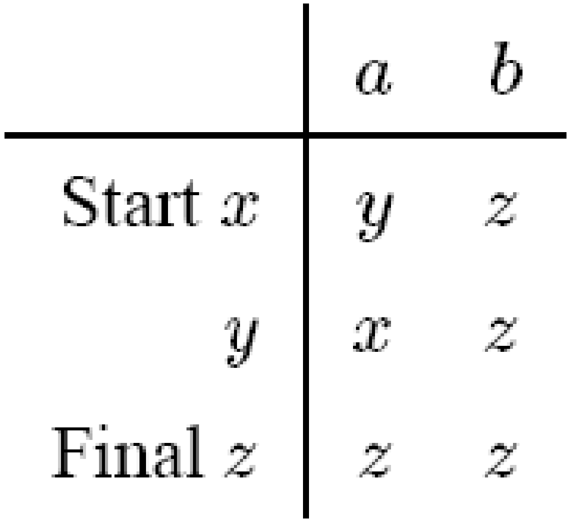
3. An alphabet ∑ = { x_1, x_2, x_3, …}.

4. A transition function δ associating each pair of state and letter with a state:

   $\delta(q,x_j) = x_k$

Transition Table

- The transition list can be summarized in a table format in which each row is the name of one of the states, and each column is a letter of the input alphabet.
- For example, the **transition table** for the FA above is



	a	b
Start x	y	z
y	x	z
Final z	z	z

Transition Diagrams

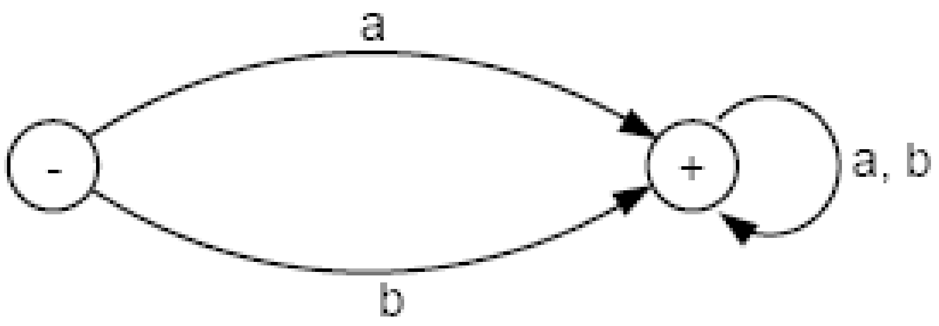
- Pictorial representation of an FA gives us more of a feeling for the motion.
- We represent each state by a small **circle**.
- We draw **arrows** showing to which other states the different **input letters** will lead us. We label these arrows with the corresponding input letters.
- If a certain letter makes a state go back to itself, we indicate this by a **loop**.
- We indicate the start state by a **minus sign**, or by labeling it with the word **start**.
- We indicate the final states by **plus signs**, or by labeling them with the word **final**.
- Sometimes, a start state is indicated by **an arrow**, and a final state is indicated by drawing **concentric circles.**

Transition Diagrams contd.

- When we depict an FA as circles and arrows, we say that we have drawn a **directed graph**.
- We borrow from Graph Theory the name **directed edge**, or simply **edge**, for the arrow between states.
- ***Every state has as many outgoing edges as there are letters in the alphabet.***
- **It is possible for a state to have no incoming edges or to have many.**

Example – Null String as an Input

- *By convention, we say that the null string starts in the start state and ends also in the start state for all FAs.*
- Consider this FA:



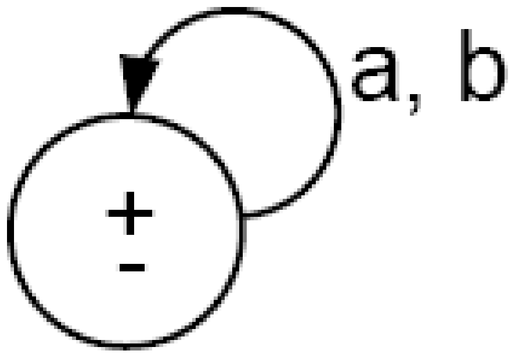
- The language accepted by this FA is the set of all strings except **Λ**.

The regular expression of this language is

$(a + b)(a + b)^* = (a + b)^+$

Example

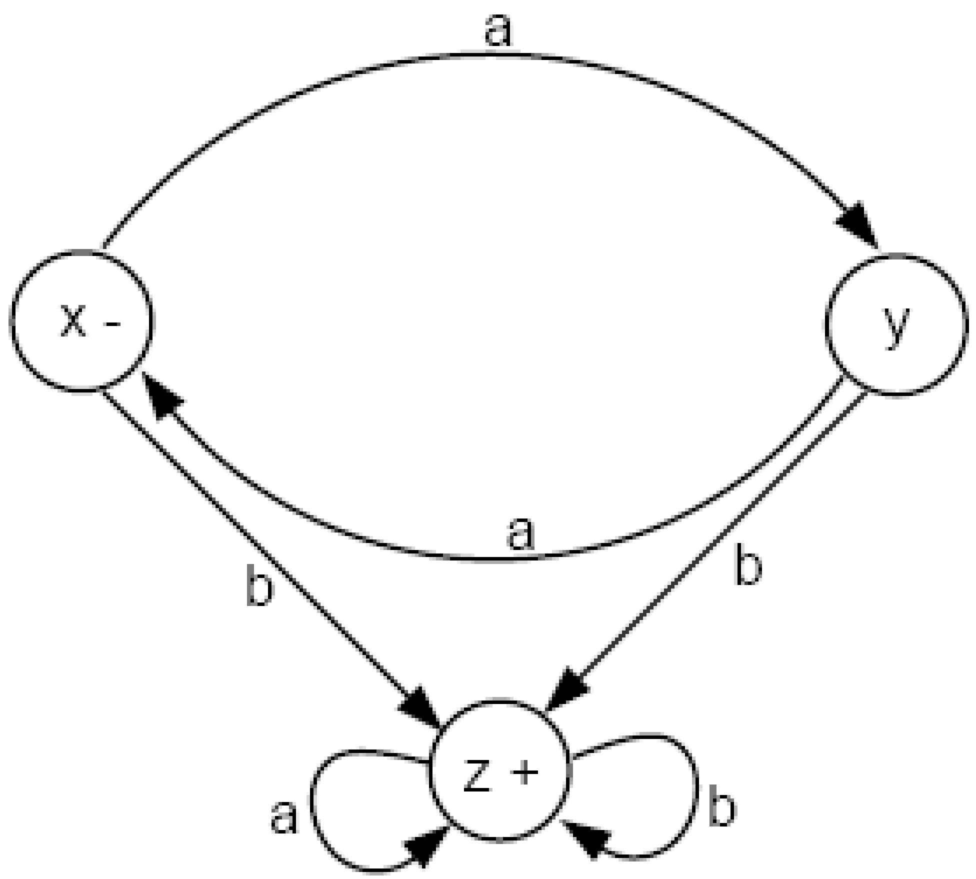
- One of many FAs that accept all words is



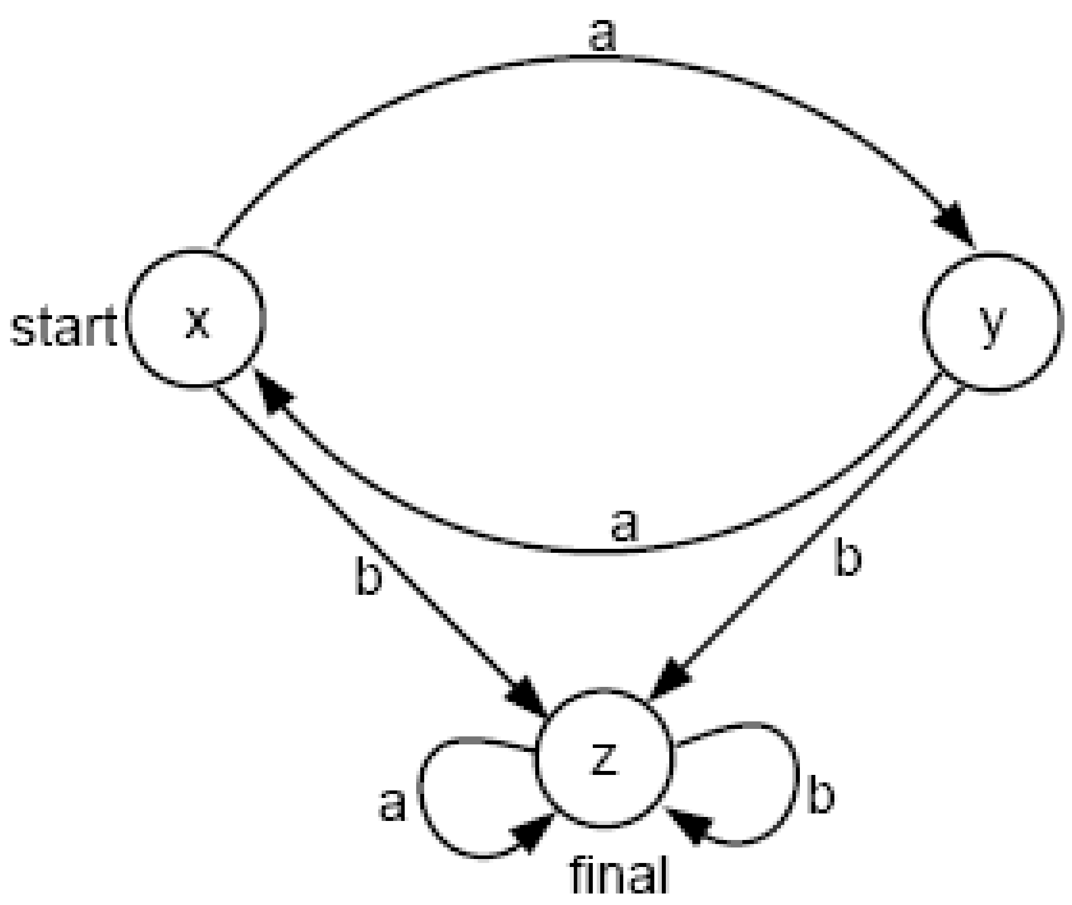
- Here, the ± means that the same state is both a start and a final state.
- The language for this machine is

  (a + b)*

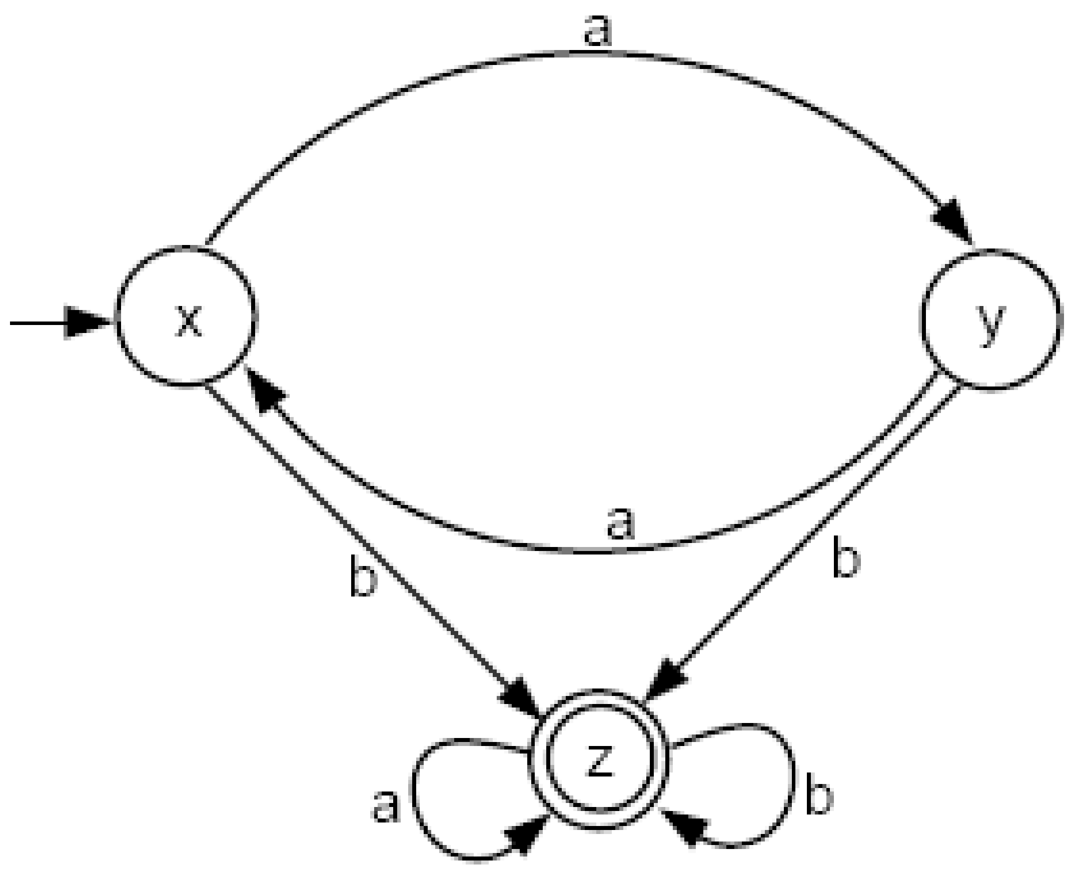
Transition Diagram (cont.)



Transition Diagram (cont.)



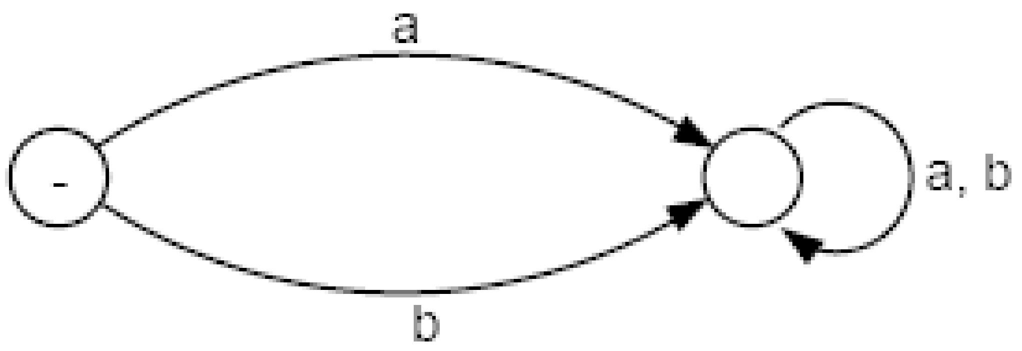
Transition Diagram (cont.)



Example

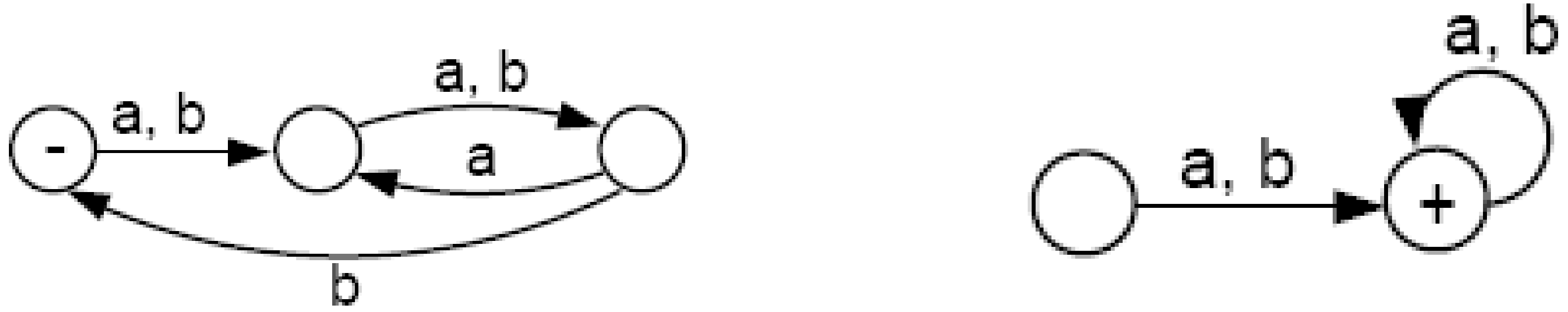
- There are FAs that accept no language. These are of two types:

1. The first type includes FAs that have no final states, such as



Example

2. The second type include FAs of which the final states can not be reached from the start state.
   1. This may be either because the diagram is in two separate components. In this case, we say that the graph is **disconnected**, as in the example below:



Example

2. Or it is because the final state has no incoming edges, as shown below: