Formal Languages & Automata Theory

Lecture 2 :

Deterministic finite Automata (DFA)

Prof. Dr. Ebtsam Abdelhakam
Faculty of Computers and Information
Minia University

Finite State Transition system

- Automaton is a machine which accepts the strings of a language L over an input alphabet Σ.

- *A finite automaton has a finite set of states with which it accepts or rejects strings.*

- The finite state represents a mathematical model of a system with certain input.
- The model finally gives certain output (Yes/No).

Applications of Finite Automata

- The applications of Finite Automata are as follows –

1. Design of the lexical analysis of a compiler.
2. Recognize the pattern by using regular expressions.
3. Helpful in text editors.
4. Used for spell checkers.
5. Protocol analysis text parsing.
6. Video game character behavior.
7. Security analysis.
8. CPU control units.
9. Natural language processing Speech recognition, etc.

Types of Automata

- There are four different types of Automata that are mostly used in the theory of computation (TOC).

1. *Finite-state machine (FSM).*
2. *Pushdown automata (PDA).*
3. *Linear-bounded automata (LBA).*
4. *Turing machine (TM).*

- When comparing these four types of automata, Finite-state machines are less powerful whereas Turing machines are more powerful.

Types of Finite Automata

Finite Automata

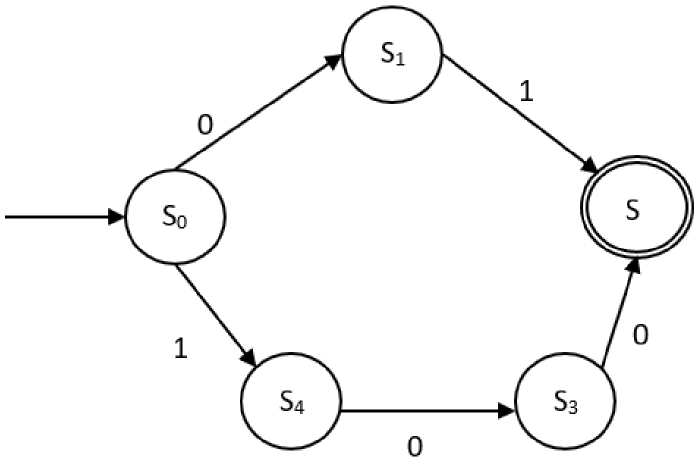
- **Deterministic Finite Automata (DFA)**
- **No deterministic finite Automata (NFA)**

Acceptance of Strings and Languages

The strings and languages can be accepted by finite automata, when it reaches to a final state.

There are two notations for representing FA.

For example:



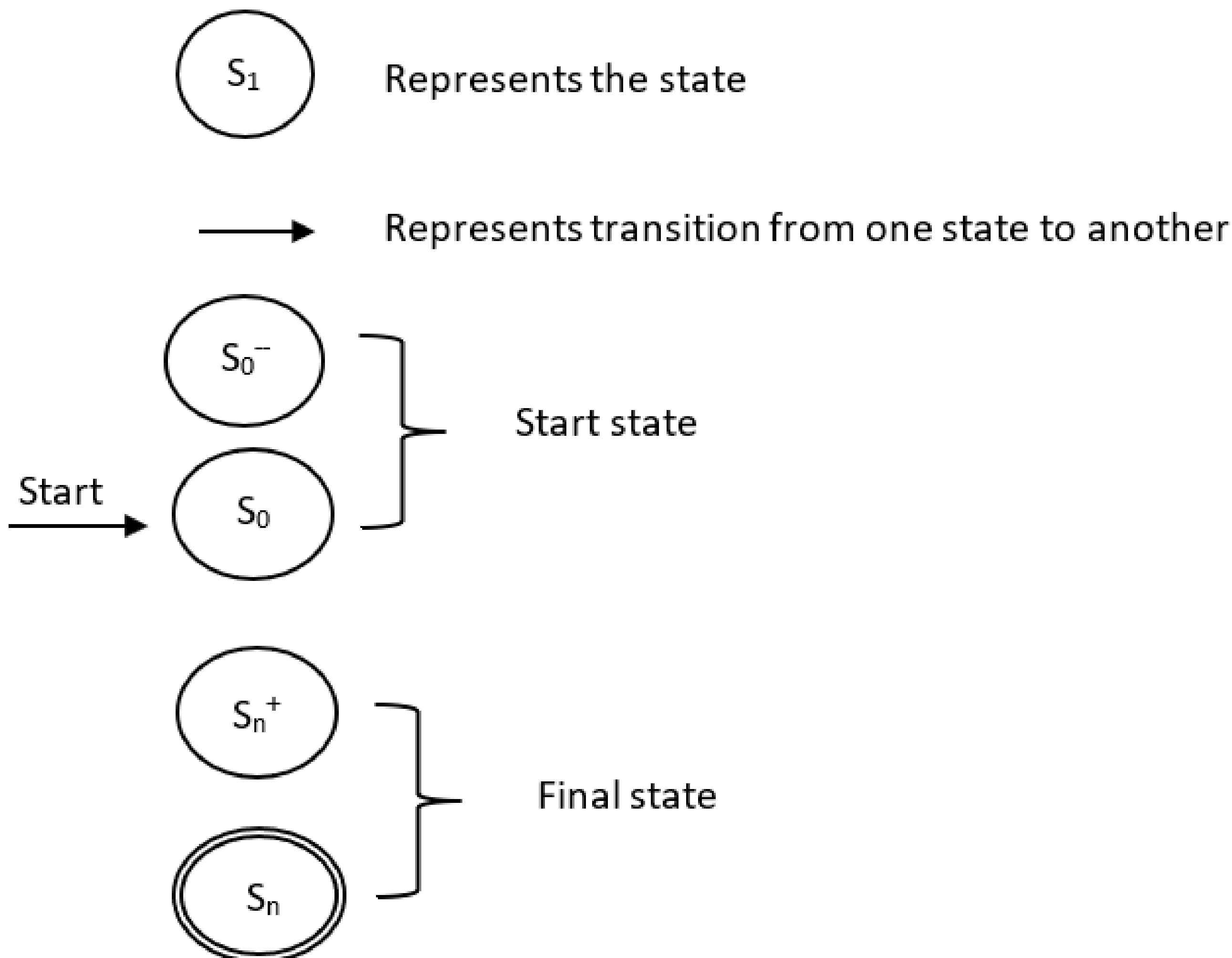
1. **Transition diagram.**
2. **Transition Table**

For example:

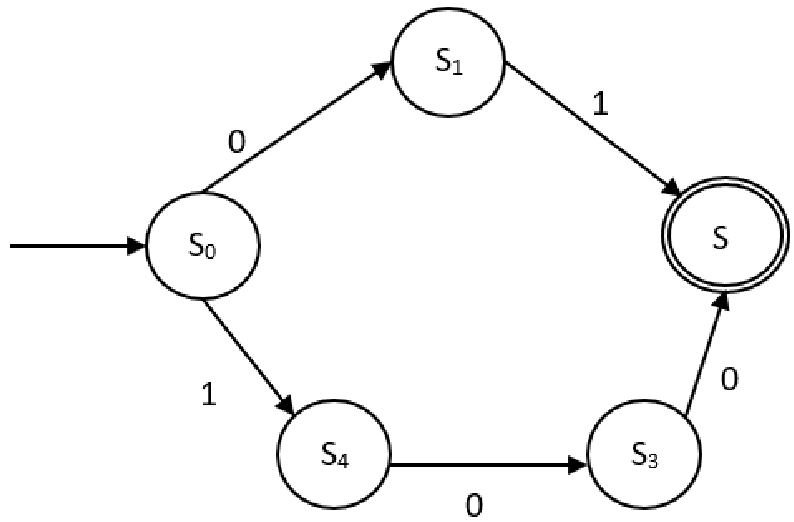
States \ Input	a	b
q_0	q_1	-
q_1	-	q_2
q_2	q_2	-

Transition Diagram

The notations used in transition diagram are:



For example:



The FA can be represented using translation. The machine initially is in start **state S_0** then on receiving input 0 it changes to **state S_1**. From S_0 on receiving input 1 the machine changes its state to S_4. **The state S_2** is a final state or accept state. When we trace the input for transition diagram and reach to a final state at end on input string then it is said that the input is accepted by transition diagram.

2- Transition table:

- This is a tabular representation of finite automata. For transition table the transition function is used.

For example:

Input / States	a	b
q_0	q_1	-
q_1	-	q_2
q_2	q_2	-

The rows of the table correspond to states and

columns of the table correspond to inputs.

Formal Definition of FA

A Finite automata is a collection of 5 – tuple (Q Σ, δ, q_0, F) where,

Q	is a finite sec of states, which is non-empty.
Σ	is input alphabet, indicates input set.
q_0	is an initial state and q0 is in Q i.e. $q_0 \in Q$.
F	is a set of final states.
δ	is a transition function or a mapping function. Using this function the next state can be determined.

Write 5-tuples ?

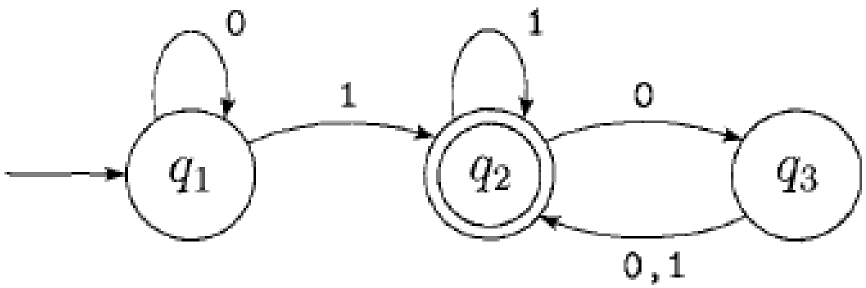


FIGURE 1.6
The finite automaton M_1

We can describe M_1 formally by writing $M_1 = (Q, \Sigma, \delta, q_1, F)$, where

1. $Q = \{q_1, q_2, q_3\}$,
2. $\Sigma = \{0,1\}$,
3. δ is described as

	0	1
q_1	q_1	q_2
q_2	q_3	q_2
q_3	q_2	q_2

4. q_1 is the start state, and
5. $F = \{q_2\}$.

Example 1.5: Design a FA which accepts the only input 101 over the input set Z = {0, 1}

Solution:

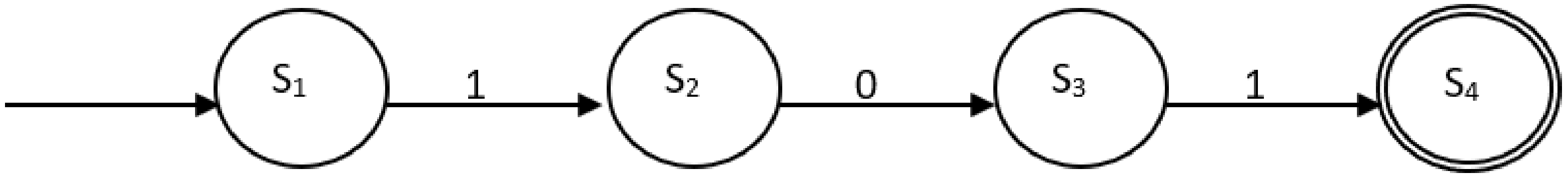


Fig. 1.10

Note that in the problem statement it is mentioned as only input 101 will be accepted. Hence in the solution we have simply shown the transitions for input 101. There is no other path shown for other input.

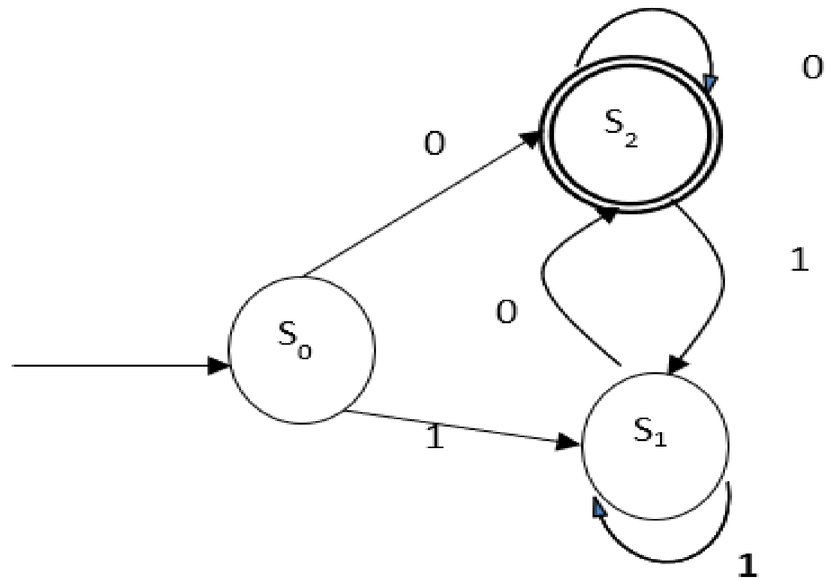
Example 1.6: Design a FA which check whether the given binary number is even.

Solution: The binary number is made up 0's and 1's when any binary number ends with 0 it is always even and when a binary number ends with 1 it is always odd. For example,

0100 is a even number, it is equal to 4.

0011 is a odd number, it is equal to 3.

And so, while designing FA we will assume one start, one state ending in 0 and other state for ending with 1. Since we want to check, whether give binary number is even or not, we will make the state for 0, the final state.



The FA indicated clearly S1 is a state which handles all the 1's and S2 is a state which handles all the 0's Let us take some input.

$$
\begin{aligned}
01000 \quad &\Rightarrow 0\underline{S_2}\,1000 \\
&01\underline{S_1}\;\;000 \\
&010\,\underline{S_2}\,00 \\
&0100\,\underline{S_2}\,0 \\
&01000\,\underline{S_2}
\end{aligned}
$$

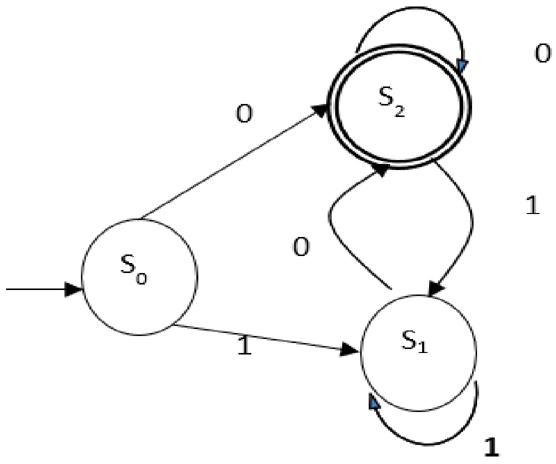
Now at the end of input we are in final or in accept state so it is a number similarly let us take another input.

$$
\begin{aligned}
1011 \quad &\Rightarrow 1\underline{S_1}011 \\
&10\underline{S_2}\,11 \\
&101\underline{S_1}\,1 \\
&1011\,\underline{S_1}
\end{aligned}
$$

Now we are at the state S_1 which is a start state.

Another idea to represent FA with the help of transition table.

States \ Input	0	1
→ S_0	S_2	S_1
S_1	S_2	S_1
S_2	S_2	S_1

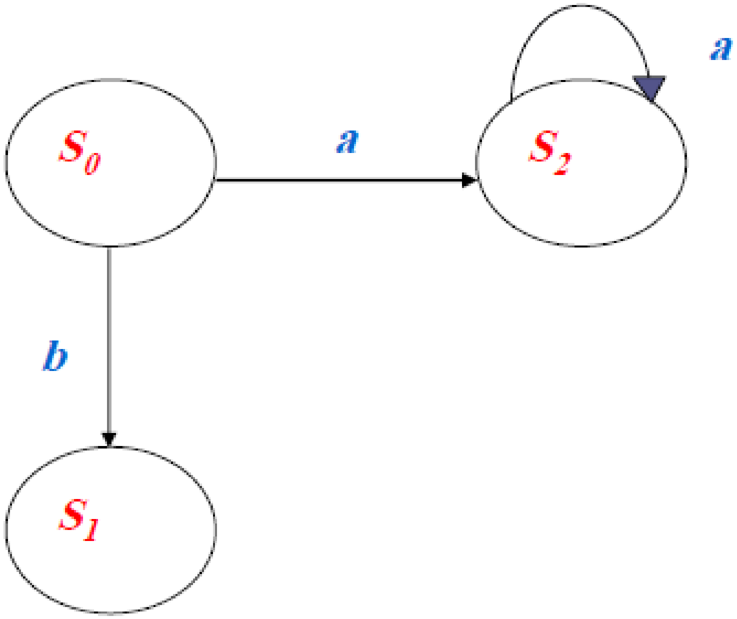


Thus, the table indicates in the first column all the current states. And under the column 0 and 1 the next states are shown.

The first row of transition table can be read as: when current **state is S_0**, on **input 0** the next state will be **S_2** and on **input 1** the next state will be **S_1**. The **arrow** marked to **S_0** indicates that it is a **Start state**.

Deterministic finite Automata (DFA)

The finite Automata is called Deterministic. Finite Automata if there is only one path for a specific input from current state to next state. For example, the DFA can be shown as below.



From state S_0 for input ' a ' there is only one path, going to S_2. Similarly, from S_0 there is only one path for input b going to S_1.

Example 1.7: Design FA which accepts only those strings which start 1 and ends with 0.

Solution: The FA will have a start state A from which only edge with input 1 will go to next state.

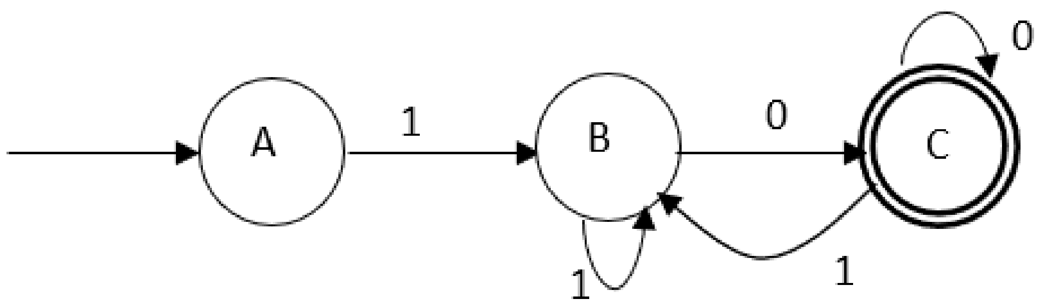


Fig. 1.12

In state B if we read 1 we will be in B state but if we read 0 at state B we will reach to state C which is a final state. In state C if we read either 0 or 1 we will go to state C or B respectively. Note that the special care is taken for 0, if the input ends with 0 it will be in final state.

Example 1.8: Design FA which accepts odd number of 1's and any number of 0's.

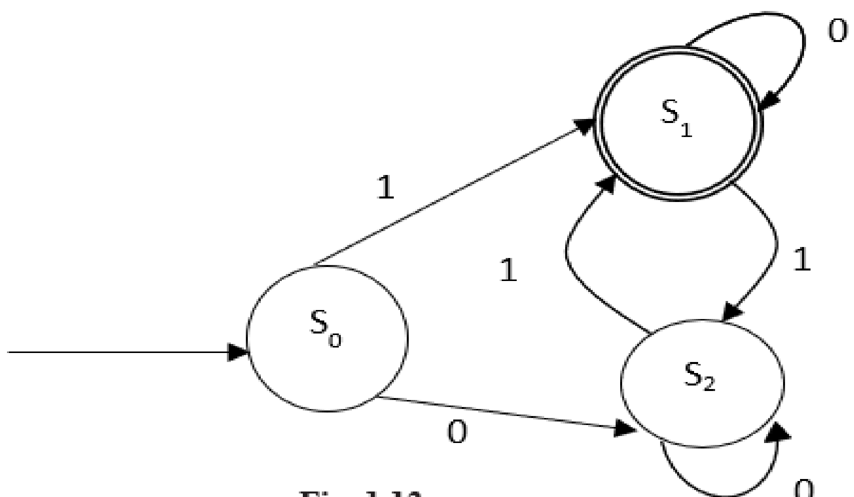


Fig. 1.13

Solution: In the problem statement, it is indicated that there will be a state which is meant for odd number of 1's and that will be the state. There is no condition on number of 0's.

At the start if we read input 1 them we will go to state S_1 which is a final state as we have read odd number if 1's. There can be any number of zeros at any state and therefore the self-loop is applied to state S_2 as well as to state S_1, for example, if the input is 10101101, in this string there are any number of zeros but odd number of ones.

Lab tutorial (1)

1: Design FA to check whether given decimal number is divisible by three.

2: Design FA which checks whether a given binary number is divisible by three.

3: Design FA which accepts even number of 0's and even number of 1’s.

4: Design FA to accept the string that always ends with 00.

5: Construct the transition graph for a FA which accepts a language 1 over ∑{0,1} in which every string start with 0 and ends with 1.

Lab tutorial (1)

6: Design FA to accept L, where L = (Strings in which a always appears tripled) over the set $\sum = (a, b)$.

7: Design FA to accept L where all the strings in L are such that total number of a 's in them are divisible by 3.

8: Design a DFA to accept string of a's and b's ending with 'abb' over $\sum = \{a, b\}$.

9: Design DFA which accepts all the strings not having more than two a's over $\Sigma = (a, b)$.

10: Design DFA over $\sum = \{a, b\}$ for.

i) $(ab)^n$ with $n \geq 0$.

ii) $(ab)^n$ with $n \geq 1$.

Any Questions?