Regular Languages

Regular language:

a Language recognized (Accepted) by some Finite Automata (FA)

Finite Automata (FA) = Finite State Machine (Models of Computer)

FA=∑States +∑ Rules for going from $State_1 \rightarrow State_2$,

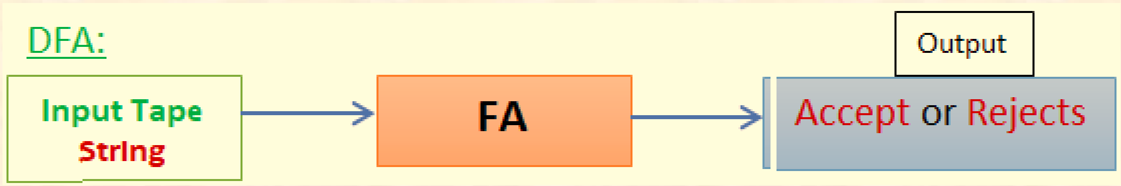
depending on the input Symbol.

Finite Automata (FA)

Deterministic Finite Automata (DFA)

DFA: FA in which, Every State has exactly one transition for each input Symbol.

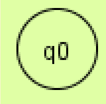
- good models for computers with limited memory
- No Temporary Memory

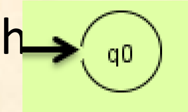


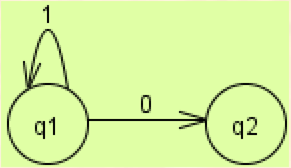
We can define the DFA informally (using the state Diagram) or formally (by defining each item or group).

Deterministic Finite Automata DFA

Informal Definition of DFA (State Diagram):

- a **state diagram** is used to model the dynamic behavior of a machine in response to Input Symbols.
- **Elements of state diagram**
  - **State**: A **state** represents the **state** of a machine at a particular given point of time.it is represented by a circle (containing the *name of the state*). Exp. 

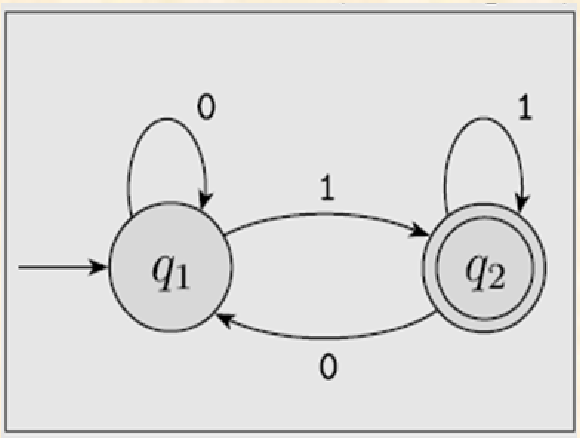
  - Initial (Start) **State**: This **state** shows the first state of the machine. It's marked by an arrow that goes into th 

Exp.
  - **Transition**: The transition from one **state** to another **state** of a machine is represented by an **arrow**. The transition is labeled with the Input Symbol that triggered it and the action that results from it. A state can have a transition that points back to itself. Exp. 

Deterministic Finite Automata DFA

Informal Definition of DFA (State Diagram):

Example:



- This machine contains 2 states q1 and q2, q1 is the start state, q2 is the final state.
- Input Alphabet (Symbols) is the set ∑={0,1}.
- At beginning, the machine is in state q1 and if symbol 0 enters, it moves to state q1, but if symbol 1 enters, it moves to state q2, etc.

Deterministic Finite Automata DFA

Formal Definition of DFA:

DFA is a 5-tuple, $M=(Q, \Sigma, \delta, q_0, F)$:

- Q: Finite Set of States , example $Q=\{q0,q1,q2\}$
- ∑: Input Alphabet (Symbols), example: $\Sigma=\{0,1\}$
- $\delta: Q\times\Sigma \rightarrow Q$: Rules for moving (Transition Function)

δ(current state, input symbol)=next state

Exp. 

- $q_0 \in Q$: Start (initial) State
- $F \subseteq Q$: Accept (Final) States Set

Deterministic Finite Automata DFA

Example:

The definition of the machine M accepts any word from the alphabet {0, 1} ending with symbol 1 in both formal definition and informal definition equivalently.

Formal Definition

$M(Q, \Sigma, \delta, q_1, F)$

$Q = \{q_1, q_2\}$

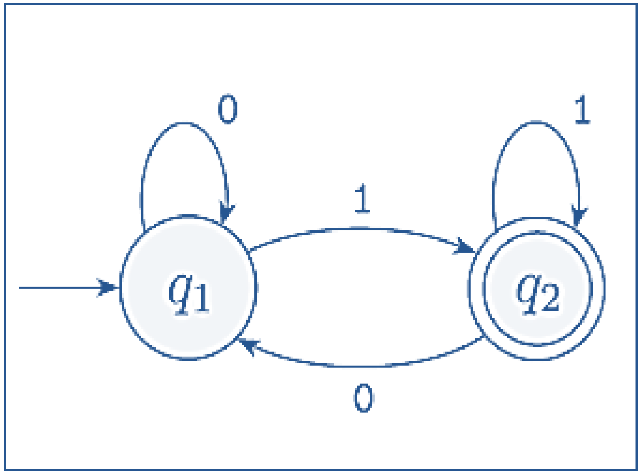
$\Sigma = \{0,1\}$

q_1 Start State

$F = \{q_2\}$

δ	0	1
q_1	q_1	q_2
q_2	q_1	q_2.

Informal Definition (State Diagram)



We say that the M machine accepts the Language L and we denote it with L(M):

L(M) = {w: w ends in a 1}

Deterministic Finite Automata DFA

If A= {All Strings that M accepts} , then

Language of machine M→L(M)=A,

or

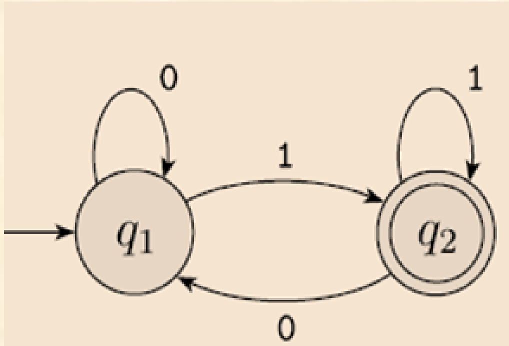
M recognizes (accepts) A →L(M)=A

M may accept several strings, but

it always recognizes only one Language.

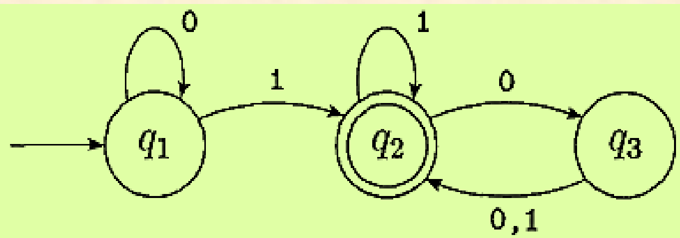
In the example,

L(M) = {w: w ends in a 1}



Deterministic Finite Automata DFA

Example: DFA M=(Q, ∑, δ, q_1, F)



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

δ	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
5. $F = \{q_2\}$.

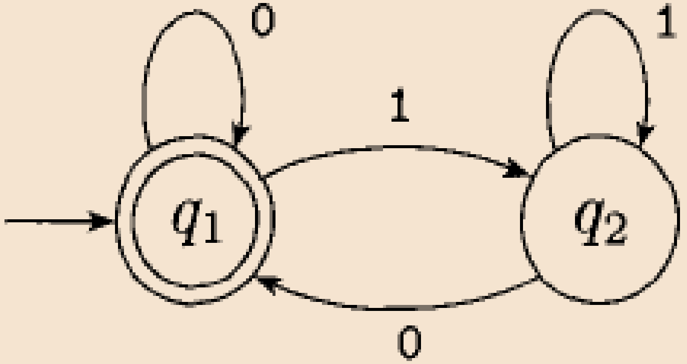
Or →

δ	0	1
$\rightarrow q_1$	q_1	q_2
q_2	q_3	q_2
q_3	q_2	q_2

L(M) = {w: w contains at least one 1 *followed by* an even number of 0's or 01.}

Deterministic Finite Automata DFA

Example: DFA $M=(Q, \Sigma, \delta, q_1, F)$



$L(M) = \{w \mid w \text{ is the empty string } \varepsilon \text{ or ends in a } 0\}.$

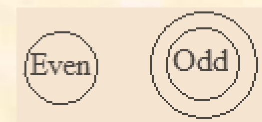
Homework: write the formal Definition of the DFA .

Designing DFA

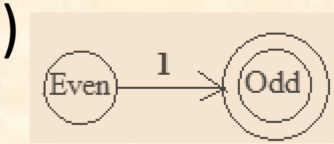
Let: $\sum=\{0,1\}$ L(M)={All strings with an Odd numbers of 1's}

Design M :

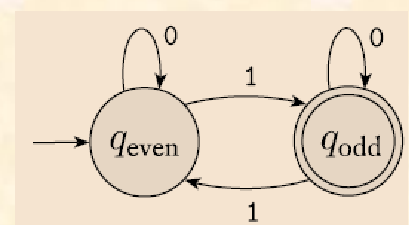
1. Determine the necessary information (States)
2. Represent this information as finite list of States (even/odd).
3. Set the accept states



4. Set the start state (for 0 symbols/ε empty string)



5. Assign the Base transitions:
6. Set transition for each symbol



Designing DFA

Let: $\sum$={0,1}

L(M)={All strings that contain the string 001 as a substring}

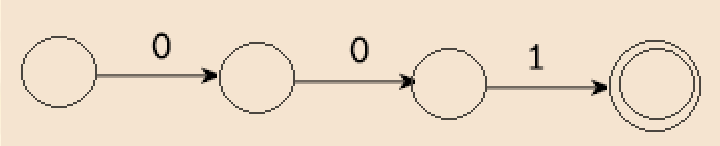
examples: 0010, 1001, 001, 1111001111 (in the Language)

11, 0000 (not in the language)

Design M :

1. States & Accept states & start state (as Base Transition):

if 0 (first) → remember- 0 (second) → remember- 1(third) :Success



2. Set transition for each symbol: