Listen Learn Lead

Engineering Excellence for Empowerment

Nadar Saraswathi College of Engineering & Technology

NSCET
E-LEARNING
PRESENTATION

LISTEN … LEARN… LEAD…

COMPUTER SCIENCE AND ENGINEERING

III YEAR / V SEMESTER

CS8501 – Theory of Computation

P.MAHALAKSHMI,M.E,MISTE

ASSISTANT PROFESSOR

Nadar Saraswathi College of Engineering & Technology,

Vadapudupatti, Annanji (po), Theni – 625531.

UNIT I

Automata Fundamentals

Introduction

Theory of Computation

- ✓ The **theory of computation** is the branch that deals with whether and how efficiently problems can be solved on a model of computation, using an algorithm.
- ✓ The field is divided into three major branches: automata theory, computability theory and computational complexity theory.

Automata Theory

- ✓ Automata comes from the greek word which means self-acting.
- ✓ **Automata theory** is the study of abstract machines and the computational problems that can be solved using these machines. These abstract machines are called automata.
- ✓ This automaton consists of

states (represented in the figure by circles),

transitions (represented by arrows).

UNIT I

Introduction to Formal Proof

Formal Proof

- ✓ In logic and mathematics, a formal proof or derivation is a finite sequence of sentences , each of which is an axiom, an assumption, or follows from the preceding sentences in the sequence by a rule of inference.
- ✓ Formal proof is a proof in which step by step procedure is used to solve the problems.

i) Deductive Proof

- ✓ Sequence of statements given with logical reasoning in order to prove initial or first statement.
- ✓ Initial statement is called as hypothesis – may be true or false typically depending on values of its parameters.

"if H then C"

- ✓ The theorem that is proved when go from a hypothesis H to a conclusion C is the statement. C is deduced from H.

Example: Theorem of Deductive Proof

If $x \geq 4$, then $2^x \geq x^2$

Proof

if H then C

Here H is $x \geq 4$ -> Hypothesis

C is $2^x \geq x^2$ -> Conclusion

The above statement $2^x \geq x^2$ will be true. It depends on the value of x.

$2^x \geq x^2$ is true for certain values of x and false for some other values.

Ex:

C is false for x=3

$2^3 \geq 3^2$ → $8 \geq 9$ (false)

So $2^x \geq x^2$ is true only when $x \geq 4$

x=4 → $2^4 \geq 4^2$ → $16 \geq 16$

x=5 → $2^5 \geq 5^2$ → $32 \geq 25$

So as much as increases the value of x, the conclusion will be true.

Topic

Additional Forms of Proof

Additional forms of proof are

i) Proof about set

ii) Proof about contradiction

iii) Proof about counter example

i) Proof about set

Collection of elements or items are called as set.

By giving proof about sets , need to prove certain properties of the set.

Example

- ✓ Consider two expressions A & B then prove A&B are equivalent
- ✓ Show that the set represented by expression A is same as the set represented by expression B.

  Let $P \cup Q = Q \cup P$

Expression A   Expression B   → Prove   A=B

- ✓ This proof is the kind of "if and only if" – Meaning is element x is in A if and only if it is in B.

Proof : P ∪ Q = Q ∪ P → Commutative law of union

LHS

Sl. No.	Statement	Justification
1.	x is in P ∪ Q 1	Given
2.	x is in P or x is in Q	1) and by definition of union
3.	x is in Q or x is in P	2) and by definition of union
4.	x is in Q ∪ P	3) Rule 2) and by definition of union

RHS

Sl. No.	Statement	Justification
1.	x is in Q ∪ P	Given
2.	x is in Q or x is in P	1) and by definition of union
3.	x is in P or x is in Q	2) and by definition of union
4.	x is in P ∪ Q	3) Rule 2) and by definition of union

Hence P ∪ Q = Q ∪ P. Thus A=B is true

ii) Proof by Contradiction

Form of Indirect Proof

Establishes the truth or validity of a preposition

"H and not C implies contradiction"

Starts by assuming that the opposite preposition is true and then shows that such an assumption leads to contradiction.

Example:

Prove that $\sqrt{2}$ is irrational or not rational

Proof

Assume that $\sqrt{2}$ is a rational number

That means $(\sqrt{2})2=(a\backslash b)2$ ---------→ (1)

Where a & b are integers and a\b is irreducible and can be simplified by lowest term, both of them can't be even.

Squaring on both sides of equation (1)

$$2=a^2\backslash b^2$$

That is $2b^2 = a^2$ b=1 a2=2 b=2 a2= 8

This shows that LHS is even. Hence RHS is also even.

Because a^2 is 2 times the integer b^2

Now if we write

$a=2k$ then

$$2b^2 = (2k)^2$$

$$= 4k^2$$

$$b^2 = 2k^2$$

So b is an even number. Thus we have established both a and b are even, which is contradiction to our assumption.

Hence √2 is irrational.

iii) Proof by Counter Example

- ✓ In order to prove certain statements, we need to see all possible conditions in which that statement remains true.
- ✓ There are some situation in which the statement can't be true.

Example 1

All primes are odd. 5/2=2 2%2=1 2%2=1

Disproof :

The integer 2 is prime but 2 is even

Example 2

There is no such pair of integers a & b such that a mod b = b mod a

Disproof: Consider a=2 and b=3 then clearly

$$2 \bmod 3 \neq 3 \bmod 2$$

Correct version: To change the statement slightly

$$a \bmod b = b \bmod a \text{ when } a=b$$

- ✓ This type of proof is called counter example. Such proof is true only at some specific condition

Topic

Inductive Proof

Introduction

- ✓ Special proofs based on some observations
- ✓ Used to prove recursively defined objects
- ✓ Recursive kind of proof which consists of sequence of parameterized statements that use the statement itself with lower values of its parameters.
- ✓ Its called as proof by mathematical induction

Induction on Integers

Given a statement S(n), n is integer.

i) Basis

Assume the lowest possible value.

Initial step like n=0 or n=1

Prove the result is true for n=0 or n=1.

ii) Induction Hypothesis

- ✓ Assign value of n to some other value K
- ✓ It means to check whether the result is true for n=K or not.

iii) Inductive Step

If n=K is true then check whether the result is true for n=K+1 or not

If we get same result at n=K+1 then that given proof is true by principle of mathematical induction.

Example

Prove by induction on n that $\sum_{i=0}^{n} i = n(n+1)/2$

(or) Prove by induction on n that P(n)=1+2+3+………+n=n(n+1)/2

Solution

i) Basis

Initially, Assume n=1 then

LHS n=1

RHS n(n+1)/2 = 1(1+1)/2 = 2/2 =1

LHS=RHS. Hence Proved

ii) Induction Hypothesis

Assume n=K and obtain the result for it

$1+2+3+\text{------}+K=K(K+1)/2$

iii) Inductive Step

Assume that equation is true for n=K, then check it is also true for n=K+1 or not.

$n=K+1 \quad 1+2+3+\text{------}+(K+1)=(K+1)((K+1)+1)/2$

LHS

1+2+3+……….+K+(K+1) = K(K+1)/2 + (K+1)

$= (K(K+1) + 2(K+1))/2 = ((K+1)(K+2))/2$

$= (K+1)(K+2)/2$

$= (K+1)(K+1+1)/2$

= RHS

Thus by induction, it is true for all n.

Basic Mathematical Notations and Techniques

Alphabets

Finite collection of symbols, which includes set of lower case & upper letters, digits, operators etc.

Ex: ∑ = { a-z, A-Z, 0,1,---}

S={a,b,c,-----,z}

W= {0,1}

String

Finite collection of symbols from alphabet.

Ex: abc,101

Empty string denoted by ε or null string. – string with zero symbols

Prefix - Leading symbols of that string

Suffix – Any number of trailing

Ex: "0011" Prefix → 0,00,001 Suffix → 1,11,011

Mango Prefix → Man Suffix → go

Operations on string

i) Concatenation – Two string are combined together to form a single string

Ex: X ={White} Y={house} XY={White house}

ii) Transpose – Reverse of a string

Ex: X={ababbb} X^R = {bbbaba}

iii) Palindrome – Strings can be read same from left to right as well as right to left.

Ex: "MADAM" "121"

Language

- ✓ A language over an alphabet is a collection of appropriate strings over that alphabet.
- ✓ Therefore, a language L is defined using an input set.
- ✓ Input set is denoted by ∑.
- ✓ Language formed by appropriate strings and strings are formed by alphabets.

Ex: Binary strings 0,1,01,10,1010,010, …. is a language over ∑={0,1}

Operations on Language

L1 & L2 – Two language then

i) Union of two language → $L1 \cup L2$

ii) Concatenation of two language → L1.L2

iii) Intersection of two language → $L1 \cap L2$

iv)Difference of two language → L1 – L2

Ex: Consider the string X=110 & Y=0110 then

XY=1100110 YX=0110110

Set

- ✓ Collection of well defined distinct elements without repetition.
- ✓ Elements are enclosed within curly brackets '{' & '}'
- ✓ Every element is separated by comma

Ex: A={a,b,c,d,e}

A is a set with five elements.

Subset

Set A is called subset of set B, if every element of set A is present in set B but not reverse.

Denoted by A⊆B

Ex:

A={1,2,3} B={1,2,3,4,5}

Therefore A⊆B

Empty Set

Set having no element. Denoted by A={ }. Written as ϕ

Null String – No value character. Denoted by ε.

Length of the set – Number of elements in a set. Denoted by |A|.

Ex: A={1,2,3,4} |A|=4

Operations on set: Consider A={1,2,3} B={2,3,4}

i) Union -> A ∪ B = {1,2,3,4} ii) Intersection –> A ∩ B = {2,3}

iii) Difference -> A-B = {1}

Kleen Closure

Star closure

Denoted by L^* - Set of all strings obtained by concatenating zero or more strings from L.

Ex: Let $\sum = \{a,b\}$

$\sum^* = \{\varepsilon,a,b,aa,bb,ab,ba,aa........\}$

Positive Closure

Defined as one or more strings from L.

Denoted by L^+ or $\sum^+$

Consists of all the strings of any length except a null string.

Ex: Let $\sum=\{a,b\}$

$\sum^+ = \{a,b,ab,ba,aaa,bbb,........\}$

Topic

Finite Automata

Finite State Systems (or) Finite Automata (FA)

Introduction

- ✓ The finite state system represents a mathematical model of a system with certain input.
- ✓ This model finally gives certain output.
- ✓ The input when is given to the machine it is processed by various states, these states are called as **intermediate states.**
- ✓ The FSM can change from one state to another in response to some inputs; the change from one state to another is called a ***transition***

Example: Control Mechanism of Elevator

Definition

A finite automata is a collection of 5 – tuples

$$M=(Q , \sum , \delta, q_0, F)$$

Where

Q -> Finite set of states which is nonempty

∑ -> Input Alphabet, indicates input set

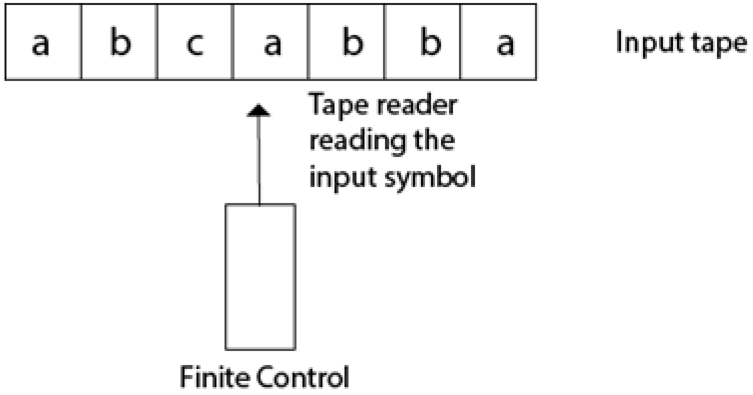
δ -> Transition function or mapping function. Using this function the next state can be determined

q_0 -> Starting State or initial state and $q_0 \in Q$

F -> Set of Final States

Finite Automata Model

The finite automata can be represented as follows



Input Tape

Linear tape having some number of cells.

Each input symbol is placed in each cell

Finite Control

✓ Contains numbers number of states and moves depends on the input symbol

✓ It decides the next state on receiving particular input from input tape.

Tape Reader

Reads the cells one by one from left to right and at a time only one input symbol is read

Acceptance of Strings and Languages

✓ The strings and languages can be accepted by a finite automata, when it reaches to a final state. There are two preferred notations for describing automata.

i) Transition Diagram

A transition diagram or state transition diagram is a directed graph which can be constructed as follows:

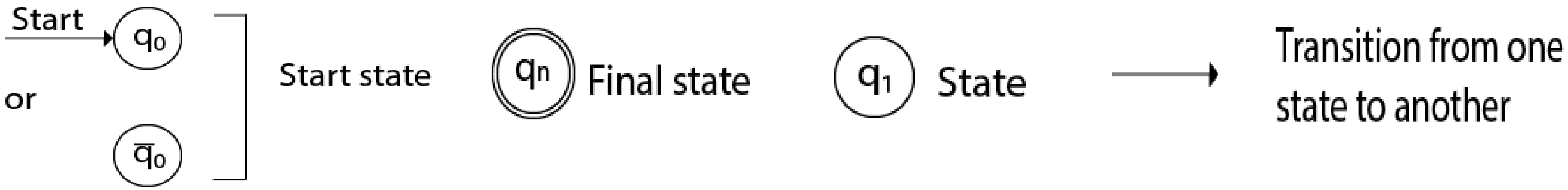
✓ There is a node for each state in Q, which is represented by the circle.

✓ There is a directed edge from node q to node p labeled a if δ(q, a) = p.

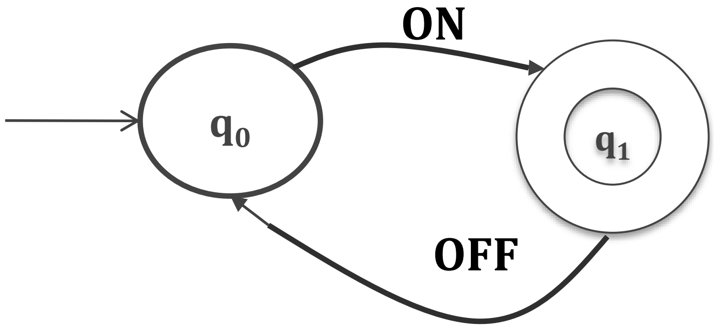
✓ In the start state, there is an arrow with no source.

✓ Accepting states or final states are indicating by a double circle.

Some Notations that are used in the transition diagram:



Example: Transition diagram for FAN Operation



ii) Transition Table

The transition table is basically a tabular representation of the transition function. It takes two arguments (a state and a symbol) and returns a state (the "next state").

- ✓A transition table is represented by the following things:
- ✓Columns correspond to input symbols.
- ✓Rows correspond to states.
- ✓Entries correspond to the next state.
- ✓The start state is denoted by an arrow with no source.
- ✓The accept state is denoted by a star.

Example: Transition Table for the above diagram

Input / States	OFF	ON
->q_0	-	q_1
*q_1	q_0	-

Applications

- ✓ Useful tool for the programs such as text editor and lexical analyzer.
- ✓ Pattern Matching
- ✓ File Searching Program
- ✓ Text Processing

Limitations

- ✓ It can recognize only simple language
- ✓ It delivers no output at all except an indication of whether the input is acceptable or not.
- ✓ FA only designed for decision making problems.

Topic

Deterministic Finite Automata (DFA)

Types of Finite Automata

There are two types of finite automata

i) Deterministic finite automata

ii) Non Deterministic finite automata

Deterministic finite automata

- ✓ Deterministic refers to the uniqueness of the computation.
- ✓ In DFA, there is only one path for specific input from the current state to the next state.
- ✓ DFA does not accept the null move, i.e., the DFA cannot change state without any input character.

Definition

A finite automata is a collection of 5 – tuples

$M=(Q, \Sigma, \delta, q_0, F)$ Where

Q -> Finite set of states which is nonempty

∑ -> Input Alphabet, indicates input set

δ -> Transition function or mapping function. Using this function the next state can be determined. $Q \times \Sigma \rightarrow Q$

q_0 -> Starting State or initial state and $q_0 \in Q$

F -> Set of Final States

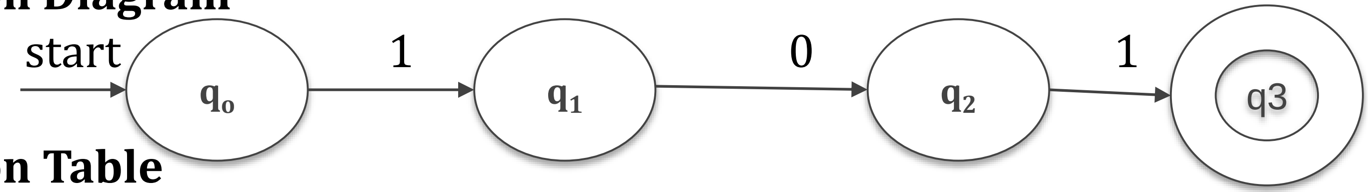
Example

Design a FA which accepts the only input 101 over the input set ∑={0,1}

Solution

L={101}

Transition Diagram



Transition Table

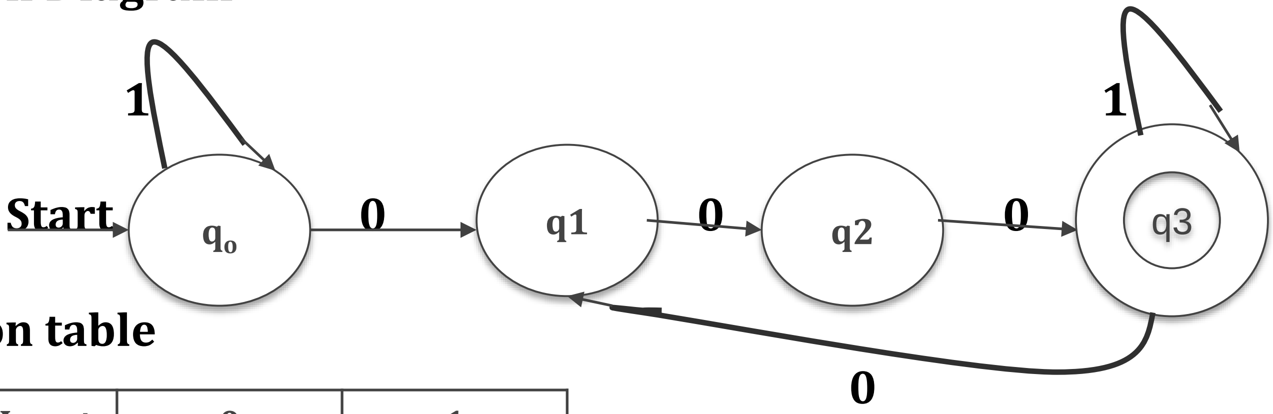
States \ Input	0	1
->q0	-	q1
q1	q2	-
q2	-	q3
*q3	-	-

Design FA with $\sum = \{0, 1\}$ accepts the set of all strings with three consecutive 0's.

Solution

L={000, 1000, 0001,111100011000,000001,…….}

Transition Diagram



Transition table

Input / States	0	1
->q_0	q_1	q_0
q_1	q_2	-
q_2	q_3	-
*q_3	q_1	q_3

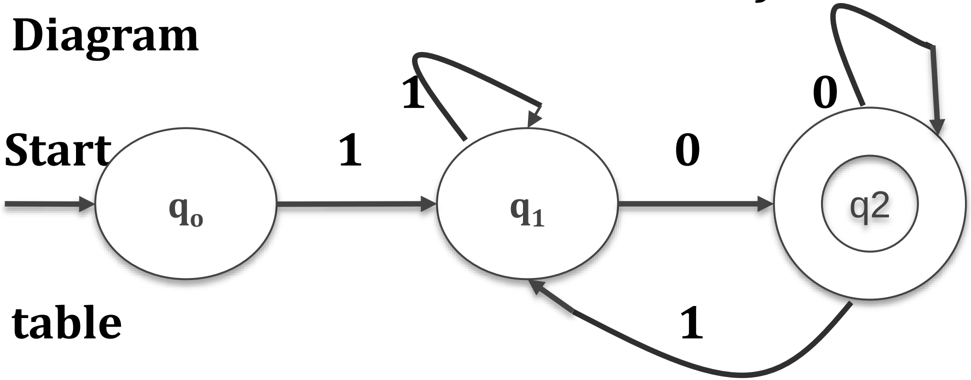
Example

Design a FA with ∑ = {0, 1} accepts the strings which start with 1 and end with 0.

Solution

L={10,111110, 100000,1 101 0 , 10100,……….}

Transition Diagram



Transition table

Input / States	0	1
$\rightarrow q_0$	-	q_1
q_1	q_2	q_1
$*q_2$	q_2	q_1

Topic

Non Deterministic Finite Automata (NDFA or NFA)

Introduction

For a single input symbol the finite automata goes to more than one state. The transition can exist in multiple states at the same time. The state may be a new state or an old state.

Definition

It has multiple transitions for a single input.

A finite automata is a collection of 5 – tuples

$M=(Q, \Sigma, \delta, q_0, F)$ Where

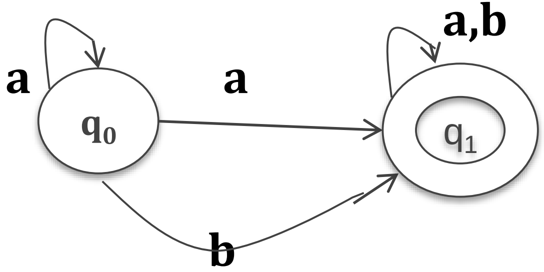
Q -> Finite set of states which is nonempty

∑ -> Input Alphabet, indicates input set

δ -> Transition function or mapping function. Using this function the next state can be determined. $Q \times \Sigma \rightarrow 2^Q$

q_0 -> Starting State or initial state and $q_0 \in Q$

F -> Set of Final States



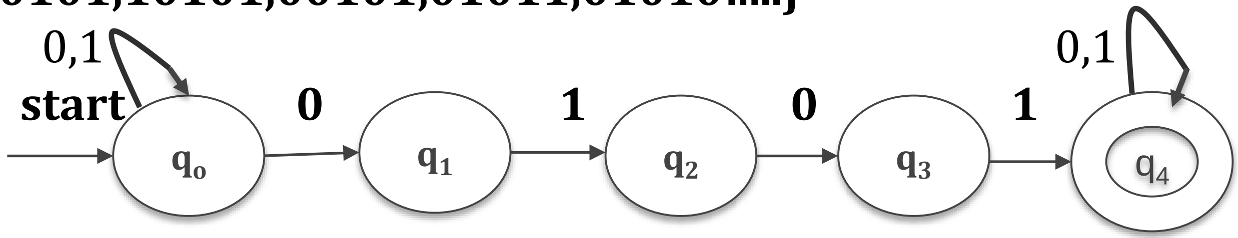
Input / States	a	b
->q_0	$\{q_0,q_1\}$	q_1
*q_1	q_1	q_1

Example

Construct a NFA for the language L={ consisting a substring 0101}

Solution

L={0101,10101,00101,01011,01010.....}



Transition Table

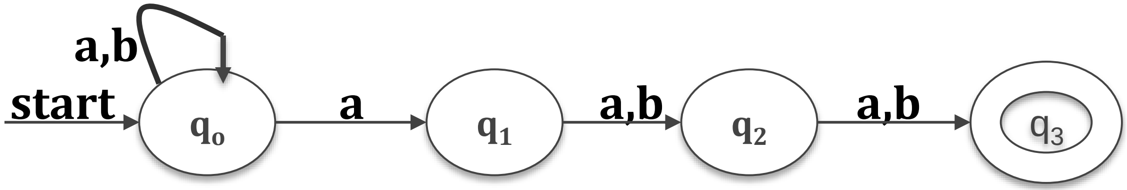
Input States	0	1
->q_0	{q_0,q_1}	q_0
q_1	-	q_2
q_2	q_3	-
q_3	-	q_4
*q_4	q_4	q_4

Example

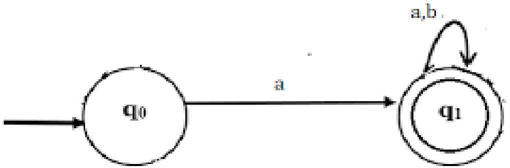
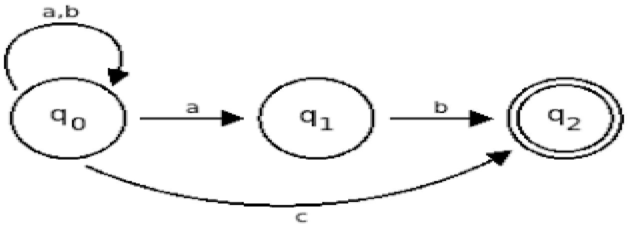
Construct a NFA for a language L which accepts all the string in which the third symbol from right end is always 'a' over ∑={a,b}

Solution

L={aaa,abb,baba,babb,aaba,…}



Difference between DFA and NFA

NFA	DFA
Every input string leads to the unique state of finite automata	For the same input there can be more than one next state
Conversion of regular expression to DFA is complex	Regular expression can be easily converted to NFA using Thompson's construction
The DFA requires more memory for storing the state information	The NFA requires more computations to match RE with input
Not possible to move to next state without reading any symbol	Move to next state without reading any symbol
More space allocation needed.	Less space needed.
Ex: 	Ex: 

Topic

Finite Automata with Epsilon Transitions

Nondeterministic Finite automata with ε

The ε – transitions are used simply to change one state to another

To make a transition without receiving an input symbol

$M=(Q, \sum, \delta, q_0, F)$

Q -> Finite set of states which is nonempty

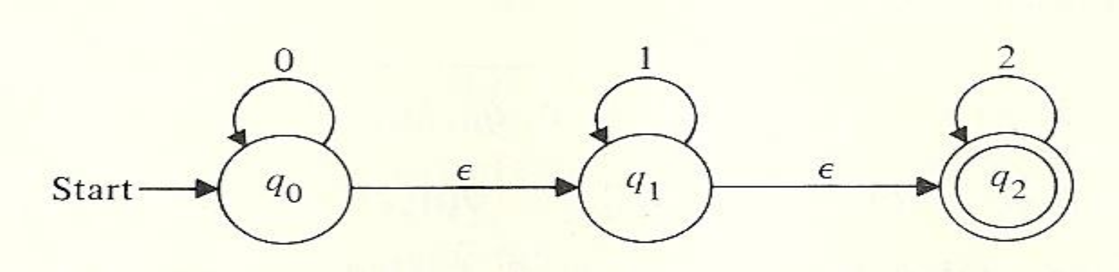
∑ -> Input Alphabet, indicates input set

δ -> Transition function or mapping function. Using this function the next state can be determined. $Q \rightarrow Q \times \sum \cup \{ \varepsilon \} \rightarrow 2^Q$

q_0 -> Starting State or initial state and $q_0 \in Q$

F -> Set of Final States

Example



Definition of ε – Closure

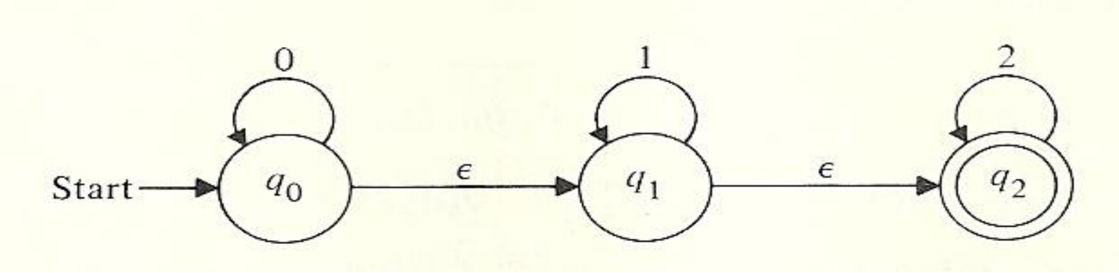
The ε – Closure (p) is a set of all states which are reachable from state p on ε – transitions such that:

i) ε – Closure(p) = p where pϵQ

ii) If there exist ε – Closure (p)={q} and δ(q, ε)=r then

ε – Closure(p) = {q,r}

Example



ε – Closure(q_0) = {q_0, q_1, q_2}

ε – Closure(q_1) = {q_1, q_2}

ε – Closure(q_2) = {q_2}