- **OBJECTIVE :** To know the different types of finite automata and regular languages

UNIT I FINITE AUTOMATA AND REGULAR EXPRESSIONS 9

Basic Definitions - Finite Automaton - DFA and NFA - Finite Automaton with -moves - Equivalence of NFA and DFA - Equivalence of NFAs with and without -moves - Regular Languages - Regular Expression - Equivalence of finite Automaton and regularexpressions-Minimization of DFA

Introduction

Strings, Alphabets and Languages

Symbol :

- ✶ A symbol is abstract entity.
- ✶ Letters and digits are examples of symbols.

String :

- ✶ A string or word is finite sequence of symbol.
- ✶ Example
  a, b and c are symbols and abcd is a string.
- ✶ The length of a string w is the number of symbols composing the string.
- ✶ It is denoted as $|w|$
- ✶ Example :
  abcd has length 4
  The empty string E, the string consisting of zero symbols $|E|=0$.
- ✶ A prefix of a string is any number of leading symbols of the string.
- ✶ Example :
  String abc has prefixes,
  E, a, ab, abc.
- ✶ A suffix is any number of trailing symbols of that string.
- ✶ Example :
  String abc has suffixes
  E, c, bc, abc
- ✶ A prefix or suffix of a string other than the string itself is called a proper prefix or suffix.
- ✶ The concatenation of two strings is the string formed by writing the first followed by the second without space.
- ✶ For eg.
  - ➢ Concatenation of dog and house is doghouse

- ➢ The empty string is the identity for concatenation operator
  $\in w = w \in = w$

Alphabet :

- ✶ An alphabet is a finite set of symbols.

Language :

- ✶ A language is a set of string of symbols from some one alphabet.
- ✶ The empty set φ and the set consisting of the empty string $\{\in\}$ are languages.
- ✶ The set of all string over a fixed alphabet Σ, this language is denoted by Σ^*
- ✶ Example
  1. $\Sigma = \{a\}$
     $\Sigma^* = \{\in, a, aa, aaa, ...\}$
  2. $\Sigma = \{0,1\}$
     $\Sigma^* = \{\in, 0, 1, 01, 10, 00, 11, 000, 000, ...\}$

Set Notation :

- ✶ Set is a collection of objects without repetition.
- ✶ Finite sets may be specified by two forms
  - i. listing their members between brachets.
  - ii. Set former
    $\{x | p(x)\}$
    $\{x \, in \, A | p(x)\}$
- ✶ If every member of A is a member of B, then we write $A \subseteq B$ and say A is contained in B.
- ✶ If $A \subseteq B$ but $A \neq B$, that is every member of A is in B and there is same member of B that is not in A, then we write $A \subseteq B$.
- ✶ A and B are equal if they have same members That is A = B iff $A \subseteq B$ nad $B \subseteq A$.

Operations on Sets :

1. $A \quad B$, the union of A and B is
   $\{x | x \text{ is in A of } x \text{ is in B}\}$
2. $A \quad B$, the intersection of A and B is
   $\{x | x \text{ is in A and } x \text{ is in B}\}$
3. A – B, the difference of A and B is
   $\{x | x \text{ is in A and } x \text{ is not in B}\}$
4. A x B, the Cartesian product of A and B, is the set of ordered pairs (a, b) such that a is in A and b is in B.
5. 2^A, is the power set of A is the set of all, subsets of A

Example :

Let A = {1, 2}

$B = \{2, 3\}$

1. $A \quad B = \{1, 2, 3\}$
2. $A \quad B = \{2\}$
3. $A - B = \{1\}$
4. $A \times B = \{(1, 2), (1, 3), (2, 2), (2, 3)\}$
5. $2^A = \{\phi\{1\}, \{2\}, \{1, 2\}\}$

- ✶ If A and B have n and m members respectively, then
  A x B has nm members
  2^A has 2^n members.

Relations :

- ✶ A binary relation is a set of pairs.
- ✶ The first component of each pair is chosen from a set called the domain and the second component of each pair is chosen from a set called a range.
- ✶ If R is a relation and (a, b) is a pair in R then we write a R b.

Properties of Relations :

- ✶ We say a relation R on set S is
  1. Reflexive
     if aRa for all a in S
  2. Irreflexive
     if aRa is false for all a in S
  3. Transitive
     if aRb and bRc imply aRc
  4. Symmetric
     if aRb implies bRa
  5. Asymmetic
     if aRb implies that bRa is false.
- ✶ Any asymmetric relation must be irreflexive.

Equivalence relation :

- ✶ A relation R that is
  - i. Reflexive
  - ii. Symmetric
  - iii. Transitive

    is said be an equivalence relation.

Closure of relations :

- ✶ The transitive closure of R, denoted R^t is defined by
  1) If (a, b) is in R, then (a, b) is in R^+
  2) if (a, b) is in R^t and (b, c) is in R then (a, c) is in R^+
  3) Nothing is in R^+, unless it follows from (1) and (2)
- ✶ The reflexive and transitive closure of R denoted R^+ is defined as
  $R^+ \cup \{(a, a) | a \text{ is in } s\}$

Example :

Let $R = \{(1, 2), (2, 2), (2, 3)\}$ be a relation on the set $\{1, 2, 3\}$

$R^{+} = \{(1, 2), (2, 2) (2, 3), (1, 3)\}$

$R^{*} = \{(1, 2), (2, 2), (2, 3), (1, 3), (1, 1) (3, 3)\}$

Graphs :

- ✶ A graph consist of a finite set of verticles V and a set of paris of vertices E called edges.
- ✶ Graph is denoted as $G = (V, E)$
- ✶ A path in a graph is a sequence of verticle $V_1, V_2, V_3, \ldots, V_k \; k \leq 1$, such that there is an edge (V_i, V_{i+1}) for each I $| \leq i \leq k$

Directed Graphs:

- ✶ A directed graph or digraph consists of a finite set of vertices V and a set of ordered pairs of vertices E called arcs.
- ✶ The arc from $V \rightarrow W$ is denoted as $V \rightarrow W$
- ✶ If $V \rightarrow W$ is an arc, we say

  $V \rightarrow \text{Predecessor of } W$

  $W \rightarrow \text{Successor of } V$

Trees:

- ✶ A tree is a digraph with the following properties
  1. Three is one vertex, called the root that has no predecessor and from which there is a path to every vertex.
  2. Each vertex other than the root has exactly one predecessor.
  3. The successor of each vertex is ordered from the left.
- ✶ If there is a path from vertex V_1 to Vertex V_2 then V_2 is said to be descendant of V_1 and V_1 is said to be an ancestor of V_2.

INDUCTIVE PROOFS

Inductive Proofs :

- ✶ Theorems can be proved by mathematical induction
- ✶ Let P(n) be a statement about a non negative integer n
- ✶ The Principal of mathematical induction is that P(n) follows from

  (a) P (o)

  (b) P (n-1) imples P(n) for $n \geq 1$
- ✶ Condition (a) is called the basis and condition (h) is called the inductive step.

Example 1 :

Prove by mathematical induction.

$$0^2 + 1^2 + 2^2 + \ldots + n^2 = \frac{n(n+1)(2n+1)}{6}$$

Solution

Let $P(n) = 1^2 + 2^2 + 3^2 + \ldots.. n^2 = \frac{n(n+1)(2n+1)}{6}$

Basic Step

for n = 0

L.H.S. $\sum_{i=0}^{n} i^2 = 0$

R.H.S. $\frac{n(n+1)(2n+1)}{6} = 0$

Inductive step

for n = n – 1

$\sum_{i=0}^{n-1} i^2 = \frac{(n)(n-1)(2n11)}{6}$ implies $\sum_{i=0}^{n} i^2 = \frac{n(n+1)(2n+1)}{6}$

Since

$$\sum_{i=0}^{n} i^2 = \sum_{i=0}^{n-1} i^2 + n^2$$

$$= \frac{n(n-1)(2n-1)}{6} + n^2$$

$$= \frac{(n^2-n)(2n-1)+6n^2}{6}$$

$$= \frac{2n^3-2n^2-n^2+n+6n^2}{6}$$

$$= \frac{2n^3+3n^2+n}{6}$$

$$= \frac{n(2n^2+3n+1)}{6}$$

$$= \frac{n(n+1)(2n+1)}{6}$$

L.H.S = R.H.S.

$\therefore$ Thus by induction it is true for all n

Example 2 :

Prove by mathematical induction.

$0 + 1 + 2 + \ldots. + n = \frac{n(n+1)}{2}$

Solution

Let $P(n) = 1 + 2 + 3 + n = \frac{n(n+1)}{2}$

Basis Step

for n = 0

L.H.S. $\sum_{i=0}^{n} i = 0$

R.H.S. $\sum_{i=0}^{n} i = \frac{n(n+1)}{2}$

$= 0$

Inductive step

for n = n – 1

$\sum_{i=0}^{n} i = \frac{(n-1)n}{2}$ implies $\sum_{i=0}^{n} i = \frac{n(n+1)}{2}$

Since

$$\sum_{i=0}^{n} i = \sum_{i=0}^{n} i + n$$

$$= \frac{(n-1)n}{2} + n$$

$$= \frac{(n-1)n + 2n}{2}$$

$$= \frac{n^2 - n + 2n}{2}$$

$$= \frac{n^2 + 1}{2}$$

$$= \frac{n(n+1)}{2}$$

L.H.S. = R.H.S.

Thus by induction it is true for all n.

Example 3 :

Prove if $x \geq 4$ then $2^x \geq x^2$ by mathematical inducations

Solution

Basic step :

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

$16 \geq 16$

x = 5 then $2^5 \geq 5^2$

$32 \geq 25$

Inductive step

put$x$ $= x + 1$

$\therefore 2^{x+1} \geq (x+1)^2$

$2^x.2 \geq (x+1)^2$

$2^x.2 \geq 2x^2 \geq (x1+)^2$

$2x^2 \geq x^2 + 2x + 1$

$$x^2 \geq 2x+1$$

$\div$ by x

$$x \geq 2+\frac{1}{x}$$

Since $x \geq 4, \frac{1}{x} \leq \frac{1}{4}$, where RHS = 2.25

$$2x^2 \geq (x+1) \text{ for } x \geq 4$$

INTRODUCTION TO FORMAL PROOF

Introduction to formal Proof :

- ✶ Formal proof is a step by step to solve the problem.
- ✶ In format proof
  We try to prove that statement B is true because statement A is true.
- ✶ The statement A is called hypothesis and B is called conclusion statement.

The four ways of Theorem Proving

(or)

Methods of formal proof.

(1) Deductive Proof
(2) Reduction Proof
(3) Other theorem forms
(4) Theorems that appear not to be if then statement.

Deductive Proof :

- ✶ A deductive proof consists of a sequence of statements whose truth leads us from some initial statement called the hypothesis or the given statement (s) to a conclusion statement.
- ✶ The theorem that is proved when we go from a hypothesis H to a conclusion C is the statement.
  "If H then C"
  We say that C is deducted from H
- ✶ The hypothesis may be true or false hypically depending on values of its parameters.

Theorem 1

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

Proof :

- ✶ The hypothesis H is $x \geq 4$
- ✶ The hypothesis has a parameter x and thus is neither true of false.

- eg. H is true for $x = 6$ and false $x = 2$
- The conclusion C is $2x \geq x^2$
- This statement also uses parameter x and is true for certain values of x and not others.
- The intuitive argument that tells the conclusion $2x \geq x^2$ will be true whenever $x \geq 4$
- The left side $2x$ doubles each time x increase by 1
- The right side x2 grows by the ration $\left(\frac{x+1}{x}\right)^2$
- Hence if $x \geq 4$ then $2x \geq x^2$, for all integers x ie $2x \geq x^2$ is deduced from $x \geq 4$

Theorem 2 :

If x is the sum of squares of four positive integers, then $2x \geq x^2$

	Statement	Justification
1	$x = a^2 + b^2 + c^2 + d^2$	Given
2	$a \geq 1, b \geq 1, c \geq 1, d \geq 1$	Given
3	$a^2 \geq 1, b^2 \geq 1, c^2 \geq 1, d^2 \geq 1$	(2) and properties of arithmetic
4	$x \geq 4$	(1) and (3) properties of arithmetic
5	$2^x \geq x^2$	(5) and Theorem (1)

Reduction to definitions

- If you are not sure how to start a proof convert all terms in the hypothesis to their definitions.

Theorem :

Let S be a finite subset of some infinite set Let T be the complement of S with respect to Then T is infinite.

Proof :

Original Statement	New Statement
S is finite	There is an integer n such |s| = n
U is infinite	for no integer P is |U| = p
T is the complement of S	SUT = U and $S \quad T = \phi$

- We use a common proof technique called "Proof by contradiction"
- In this proof, the contradiction of the conclusion is "T is finite"
- Let us assume T is finite,

  |T| = m for some integer m

  |S| = n
- Since SUT = U and $S \quad T=\phi$ the elements of U are exactly the elements of S and T
- Thus there must be n + m elements of U |U| = n + m
- The statement that U is finite contradicts the given statement that U is infinite
- Thus our assumption is contradiction
- So T is infinite

Other Theorem Forms :

- Ways of saying "if – Then"
  - The other ways in which if H then C might appear.
    1. H implies C
    2. H only if C
    3. C if H
    4. whenever H holds, C follows.

If – And – only – If statements :

- The statement of the form

  A if and only if B is actually two if – then statements.

  (i) if A then B and

  (ii) if B then A

Theorems that appear not to be if – then statements :

- Sometimes we find a theorem that appear not to have a hypothesis.
- An example is the well-known fact from the trigonometry.
- Theorem

  $\text{Sin}^2\theta+\text{Cos}^2\theta=1$

Additional Forms of Proof :

1. Proofs about sets.
2. Proofs by contradiction.
3. Proofs by counter example.

Proving Equivalences about sets :

- If E and F are two sets, then the statement E=F means the two sets represented are the same.
- Every element in the set epresented by E is in the set represented by F
- And every element in the set represented by F is in the set represented by E.

Example :

- ✶ The commutative law of union says that we can take the union of two sets R and S in either order.
  i.e., RUS = SUR
- ✶ The commutative law of union says E = F
- ✶ The proof of any statement that asserts the equality of two sets E = F, if follows the form of any if – and – only – if – proof
  1. Proof that if x is in E then x is in F
  2. Proof that if x is in F then x is in E

Theorem :

R (S T)=(R S) (R T)

Proof :

The two set of expressions involved are

E=R (S T)

R=(R S) (R T)

Steps in the part R (S T) ddd

	Statement	Justification
1	x is in R U(S T)	Given
2	x is in R or x is in S T	(1) and definition of union
3	x is in R or x in in both S and T	(2) and definition of intersection
4	x is in RUS	(3) and definition of union
5	x is in RUT	(4) and definition of union
6	x is in (R S) (R T)	(4), (5) and definition of intersection

Steps in the 'only-if' part.

	Statement	Justification
1	x is in (R S) (R T)	Given
2	x is in RUS	(1) and definition of intersection
3	x is in RUT	(1) and definition of intersection
4	x is in R or x is in both S and T	(2) and (3) reasoning about union
5	x is in R or x is in S T	(4) and definition intersection

6	*x* is in RU S T	(5) and definition of union.

The Contrapositive :

- ✶ The contrapositive of the statement "if H then C" is "if not C then not H"
- ✶ A statement and its contrapositive are either both true or both false.
- ✶ To see "if H then C" and if not C then not H are logically equivalent.
  There are four cases to consider
  1. H and C both true
  2. H true and C false
  3. C true and H false
  4. H and C both false

Proof by contradiction :

- ✶ Another way to prove a statement of the form.
  "if H then C is to prove the statement.
  - ➢ H and not C implies false hood.
- ✶ Start by assuming both the hypothesis H and the negation of the conclusion C
- ✶ Complete the proof by showing that something known to be false follows logically from H and C
- ✶ This form of proof is called proof by contradiction.

Counter example :

- ✶ Statements that have no parameters, or that apply to only a finite number of values of its parameter are called observations.
- ✶ It is easier to prove that a statement is not a theorem than to prove it is a theorem.

Example

All primes are odd.
More formally, we might say
if integer *x* is a prime, then *x* is odd

Example

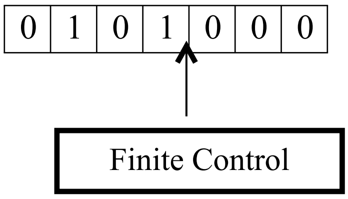
There is no pass of integers a and b such that
a mod b = b mod a

<u>BASIC DEFINITIONS DFA AND NFA</u>

Finite state systems :

- ✶ The finite Automaton is a mathematical model of a system with discute inputs and outputs.
- ✶ The system can be in a any one of a finite number of internal configuration or states.

- ✶ The state of the system summarixes the information concerning past inputs that is needed to determine the behavior of the system on subsequent inputs.
- ✶ The primary example of finite automaton is a switching circuit such as control unit of a computer.
- ✶ A switching circuit is composed of a finite number of gates each of which can be in one of two conditions usually 0 and 1
- ✶ The state of a switching network with n gates is thus any one of the 2^n assignments of 0 or 1 to the various gates.
- ✶ Text editors and the lexical analyzers found in most compiters are designed as finite state systems.

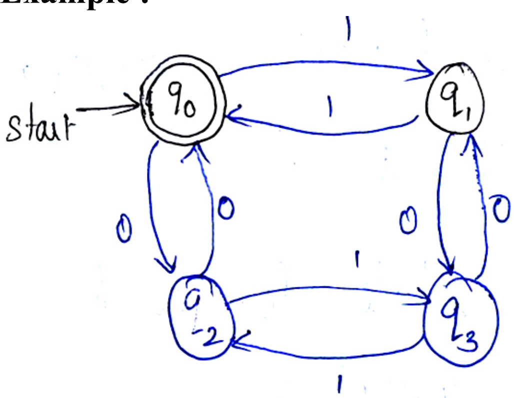


Basic Definitions :

Deterministic finite automata :

- ✶ A finite automator (FA) consists of a finite set of states and a set of transitions from state to state that occur on an inpul symbol chosen from an alphabet Σ
- ✶ For each input symbol there is exactly one transition out of each state.
- ✶ One state denoted q_0 is the initial state in which the transition starts.
- ✶ Some states are designated as final states or accepting states.
- ✶ The Deterministic finite automata can be represented as a transition diagram or transition table.
- ✶ The directed graph is used to represent the transition diagram.
    - ➢ The vertices of the graph correspond to the states of FA
    - ➢ If there is a transition from state q to state p on input a there is an are labeled a from state q to state p
    - ➢ The FA accepts a string *x* if the sequence of symbols of x leads start state to final state.

Example :



- ✶ The initial state, q_0 is indicated by the arrow labeled start.
- ✶ The final state indicated by the double circle.

- ✶ The FA accepts all string of 0's and 1's in which both the number 0's and the number of 1's are even.
- ✶ for eg.

The string accepted by the above FA, 11, 1100, 00, 0000, 1111, 110110

Formal Definition of DFA :

- ✶ We formally denote a finite automaton by a 5 tube.

M = $(Q, \Sigma, \delta, q_o, F)$

Q $\rightarrow$ is a finite set of states

$\Sigma \rightarrow$ is a finite input alphabet

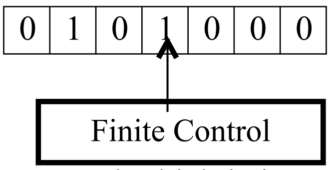
$q_o \rightarrow$ in Q is the initial state

$F \rightarrow F \subseteq Q$ is the set of final states

$\delta \rightarrow$ is the transition function mapping from $Q \times \Sigma$ to θ

ie $\delta(q, a)$ is a state for each state q and input symbol a

- ✶ FA is denoted as a finite control



- ✶ Finite control which is in same state from Q reads a sequence of symbols from Σ written on a tape.
- ✶ In one move, the FA in state q, scanning a symbol a enters state $\delta(q, a)$ and moves its head one symbol to the right.
- ✶ We define a function $\hat{\delta}$ from

$Q \times \Sigma^*$ to Q

- ✶ $\hat{\delta}(q, w)$ is the unique state p such that there is a path in the transition diagram from q to p labeled w
- ✶ Formally we define

  (1) $\hat{\delta}(q, t) = q$

  (2) for all string w and input symbols a

$\hat{\delta}(q, wa) = \delta(\hat{\delta}(q, w), a)$

- ✶ A string x is said to be accepted by FA

M = $(Q, \Sigma, \delta, q_o, F)$ if

$\delta(q_o, \boldsymbol{x}) = p$, for some P in F

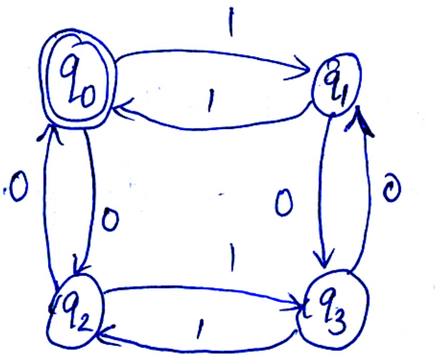
- ✶ The language accepted by FA, denoted as L(M)

L (M) = $\{\boldsymbol{x} | \delta(q_o, \boldsymbol{x}) \text{ is in } F\}$

- ✶ The language is a regular set if it is the set accepted by the same FA

Example :

Consider the transition diagram



★ This FA is denoted as

$M = (Q, \Sigma, \delta, q_o, F)$ where

$Q = \{q_0, q_1, q_2, q_3\}$

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

$q_0 = q_0$

$F = \{q_0\}$

δ is given as a transition table

s	0	1
q_0	q_2	q_1
q_1	q_2	q_1
q_1	q_3	q_0
q_2	q_0	q_3
q_3	q_1	q_2

★ To find the string accepted by DFA, suppose w = 110101 is input to M we have to find $\delta(q_0, 110101)$

$\delta(q_0, 1) = 9,$

$$\hat{\delta}(q_0, 110) = \delta(\delta(q\delta, 1), 1)$$
$$= \delta(q_1, 1)$$
$$= q_0$$

$$\hat{\delta}(q_0, 110) = \delta(\hat{\delta}(q_0, 11), 0)$$
$$= \delta(q_0, 0)$$
$$= q_2$$

$$\hat{\delta}(q_0, 1101) = \delta(\hat{\delta}(q_0, 110), 1)$$
$$= \delta(q_2, 1)$$
$$= q_3$$

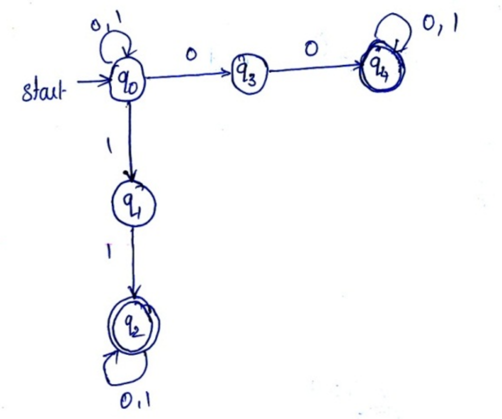
$\hat{\delta}(q_0, 11010) = \delta(\hat{\delta}(q_0, 1101), 0)$
$= \delta(q_3, 0)$
$= q_1$

$\hat{\delta}(q_0, 110101) = \delta(\hat{\delta}(q_0, 11010), 1)$
$= \delta(q_1, 1)$
$= q_0 \in F$

- Thus 110101 is accepted by the FA Thus 110101 is in L(M)
- L(M) = is the set of string with an even number of o's and an even number of 1's

Non – Deterministic finite Automata :

- A finite automata model to allow zero, one or more transitions from a state on the same input sympol. This new model is called a non deterministic finite Automata (NFA)
- Any set accepted by NFA can also be accepted by DFA.
- Example



- The input sequence a1, a2, a3 …. an is accepted by NFA, if the transition leads from the initial state of final state.

Formal Definition of NFA :

- Formally we denote a NFA by a 5 – tuple

$M = (Q, \Sigma, \delta, q_0, F)$
where
$Q \rightarrow$ set of states
$\Sigma \rightarrow$ is a finite input alphabet
$q_0 \rightarrow$ in Q is the initial state
$F \rightarrow F \subseteq Q$ is the set of final states

$\delta \rightarrow$ is the transition function mapping from $Q \times \Sigma$ to θ

$Q \times \Sigma \rightarrow 2^Q$

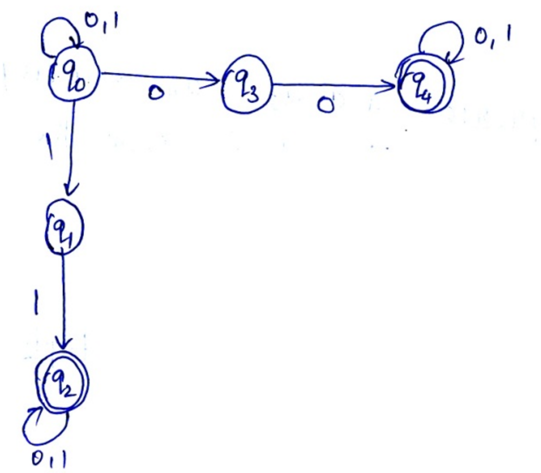
✶ The function δ can be extended to a function $\hat{\delta}$ mapping from $Q \times \Sigma^*$ to 2^Q and

(i) $\hat{\delta}(q, \in) = \{q\}$

(ii) $\hat{\delta}(q, wa) = \delta\left(\hat{\delta}(q, w), a\right)$

Example :

Consider the NFA



Let the input w = 01001

Solution :

$Q = \{q_0, q_1, q_2, q_3, q_4\}$

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

$q_0 = q_0$

$F = \{q_2, q_4\}$

δ:

	0	1
q_0	$\{q_0, q_0,\}$	$\{q_0, q_1,\}$
q_1	ϕ	$\{q_2\}$
q_2	$\{q_2\}$	$\{q_2\}$
q_3	$\{q_4\}$	ϕ
q_4	$\{q_4\}$	$\{q_4\}$

$\delta(q0, 01001) = q$

$\delta(q_0, 0) = \{q, q_3)$

$\delta(q_0, 01) \quad = \delta(\delta(q_0, 0), 1)$

$= \delta(\{q_0,q_3\},1)$

$= \delta(q_0,1) \quad \delta(q_3,1)$

$= \{q_0,q_1) \quad \phi$

$= \{q_0, q_1\}$

$\delta(q_0,010) \quad = \delta(\delta(q_0,01),0)$

$= \delta(\{q_0,q_1\},0)$

$= \delta(q_0,0) \quad \delta(q_1,0)$

$= \{q_0,q_3) \quad \phi$

$= \{q_0, q_3\}$

$\delta(q_0,0100) \quad = \delta(\hat{\delta}(q_0,010),1)$

$= \delta(\{q_0,q_3\},0)$

$= \delta(q_0,1) \quad \delta(q_3,0)$

$= \{q_0,q_3) \quad \phi \, \{q_4\}$

$= \{q_0, q_3, q_4\}$

$\delta(q_0,01001) \quad = \delta(\hat{\delta}(q_0,0100),1)$

$= \delta(\{q_0,q_3,q_4\},1)$

$= \delta(q_0,1) \quad \delta(q_3,1) \quad \delta(q_4,1)$

$= \{q_0,q_1) \quad \phi \quad \{q_4\}$

$= \{q_0, q_1, q_4\}$

$\delta(q_0,01001)$ contains a state in F so the input string 01001 is accepted by the NFA.

EQUIVALENCE DFA AND NFA

The Equivalence of DFA's and NFA's

- Since every DFA is an NFA it is clear that the class of languages accepted by NFA's includes the regular sets.
- for every NFA we can construct an equivalent DFA.

Theorem :

Let L be a set accepted by NFA, then there exists a DFA that accepts L.

Proof :

Let M = (Q,Σ,δ,q_0,F) be an NFA which accepts L.

Define a DFA.

$M1=(Q^1,\Sigma^1,\delta^1,q_0{}^1,F^1)$

- The states of M^1 are all the subsets of the set of states of M

  i.e) $Q^1 = 2^Q$
- F^1 is the set of all states in Q^1containing a final state of M.
- An element of Q^1will be denoted by $[q_1, q_2, \ldots\ldots\ldots q_1)$ are in Q.

- 2is a single state of the DFA corresponding to the set of states of the NFA
- $q_0^1 = [q_0]$
- $\delta^1 ([q_1], a) = [p_1]$

  iff $\delta(q1, a) = \{p1\}$

  $\delta 1 ([q_1, q_2, \ldots q_i], a) = [p_1, p_2, \ldots. p_j]$

  iff $\delta 1 ([q_1, q_2, \ldots q_i], a) = [p_1, p_2, \ldots. p_j]$
- It is easy to show by induction on the length of the input string x that

  $\delta^1 (q_0{}^1, x) = [q_1, q_2, \ldots q_j]$

  iff

  $\delta^1 (q_0, x) = [q_1, q_2, \ldots q_j]$

Basis :

The result is trivial for $1\ x\ 1 = 0$, since $q_0^1 = [q_0]$ and x must be $\in$

$\delta^1 ([q_0], \in = [q_0]$

iff

$\delta\ (q_0 \in) = q_0$

Induction :

- Suppose that the hypothesis is true for inputs of length m or less
- Let xa be a string of length m + 1 with a in Σ then

  $\delta^1 (q_0{}^1, x\mathrm{a}) = \delta^1 (\delta^1 (q_0{}^1, x), \mathrm{a})$

By the inductive hypothesis

$\delta^1 (q_0{}^1, x) = [p_1, p_2, \ldots p_j]$

iff

$\delta(q_0, x) = [p_1, p_2, \ldots p_j]$

$\delta^1 (q_0, x\mathrm{a}) = \delta^1 (\delta^1 (q_0{}^1, x), \mathrm{a})$

$= \delta^1 [p_1, p_2, \ldots.. p_j], \mathrm{a})$

$= \{r_1, r_2, \ldots r_k\}$

iff

$\delta^1 (q_0, x\mathrm{a}) = \delta(\delta(q_0, x), \mathrm{a})$

$= \delta^1 [p_1, p_2, \ldots.. p_j], \mathrm{a})$

$= \{r_1, r_2, \ldots r_k\}$

Thus

iff

$\delta^1 (q_0{}^1, x\mathrm{a}) = [r_1, r_2, \ldots r_k]$

iff

$\delta(q_0, x\mathrm{a}) = [r_1, r_2, \ldots r_k]$

which establishes the inductive hypothesis

★ $\delta^1 (q_0{}^1, x)$ is in F^1 exactly when $\delta(q_0, x)$ contains a state in F

L (M) = L (M)

PROBLEM ON NFA TO DFA

Example 1:

1. Let M = $\left(\{q_0, q_1\}\{0,1\}, \delta, q_0, \{q_1\} \text{be an}\right)$

NFA where $\delta(q_0, 0) = \{q_0, q_1)\}$

$\delta(q_0, 1) = \{q_1\}$

$\delta(q_0, 0) = \phi$

$\delta(q_0, 1) = \{q_0, q_1)\}$

Solution :

We can construct a DFA

$M^1 = (Q^1, \{0, 1\}, \delta^1, [q_0], F)$

Q^1 = all subsets of $\{q^0, q^1)$

$Q^1 = \{[q_0], [q_1], [q_0, q_1], \phi\}$

F^1 = Set of states of Q^1 containing state in F

$F^1 = \{ [q_1], [q_0, q_1]\}$

Transition Table : δ^1

	0	1
[q0]	$[q_{0,} q_{1,}]$	$[q_1]$
$[q_1]$	ϕ	$[q_0, q_1]$
$[q_0, q_1]$	?	?
ϕ	ϕ	ϕ

To find $\delta^1 ([q_0, q_1], 0$

$\delta(\{q_0, q_1\}, 0) = \delta(q_0, 0) \quad \delta(q_1, 0)$

$= \{q_0, q_1\} \quad \phi$

$= \{q_0, q_1\}$

$\delta 1([q_0, q_1], 0) = [q_0, q_1]$

To find $\delta^1 ([q_0, q_1], 1$

$\delta(\{q_0, q_1\}, 1) = \delta(q_0, 1) \quad \delta(q_1, 1)$

$= \{q_1\} \quad \{q_0, q_1\}$

$= \{q_0, q_1\}$

$\delta^1([q_0, q_1], 1) = [q_0, q_1]$

$\therefore \quad M^1 = (Q^1, \Sigma^1, \delta^1, q_0{}^1, F^1)$

$Q^1 = \{ [q_0], [q_1], [q_0, q_1], \phi \}$

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

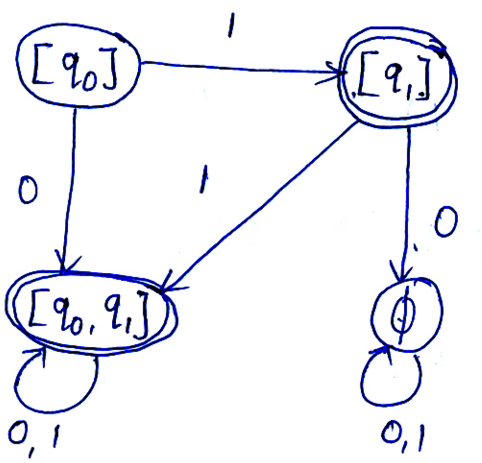
$q_0{}^1 = [q_0]$

$F^1 = \{ [q_1], [q_0, q_1] \}$

δ^1

	0	1
$[q_0]$	$[q_0, q_1]$	$[q_1]$
$[q_1]$	ϕ	$[q_0, q_1]$
$[q_0, q_1]$	$[q_0, q_1]$	$[q_0, q_1]$
ϕ	ϕ	ϕ

Transition Diagram



Examples 2 :

Consider the following NFA

	a	b
q_0	$\{q_0, q_1\}$	$\{q_2\}$

q_1	$\{q_1\}$	$\{q_0\}$
$[q_2]$	$\{q_0\}$	$\{q_1, q_2\}$

Construct an equivalent DFA

Solution :

We construct DFA M^1

$M^1 = (Q^1, \Sigma^1, \delta^1, q_0{}^1, F^1)$

$Q^1 = \{ [q_0], [q_1], [q_2], [q_0, q_1], [q_0, q_2], [q_1, q_2], [q_0, q_1, q_2], \phi \}$

$\Sigma = \{a, b\}$

$q_0{}^1 = [q_0]$

$F^1 = \{ [q_2], [q_0, q_1], [q_1, q_2], [q_0, q_1, q_2] \}$

δ^1

	a	b
$[q_0]$	$[q_0, q_1]$	$[q_2]$
$[q_1]$	$[q_1]$	$[q_0]$
$[q_2]$	$[q_0]$	$[q_1, q_2]$
$[q_0, q_1]$	?	?
$[q_0, q_2]$	?	?
$[q_1, q_2]$	?	?
$[q_0, q_1, q_2]$	?	?

To find $\delta^1([q_0, q_1], a$

$$\delta(\{q_0, q_1\}, a) = \delta(q_0, a) \quad \delta(q_1, a)$$
$$= \{q_0, q_1\} \quad \{q_1\}$$
$$= \{q_0, q_1\}$$
$$\delta^1([q_0, q_1], a) = [q_0, q_1]$$

To find $\delta^1([q_0, q_1], b$

$$\delta(\{q_0, q_1\}, b) = \delta(q_0, b) \quad \delta(q_1, b)$$
$$= \{q_2\} \quad \{q_0\}$$
$$= \{q_0, q_2\}$$
$$\delta^1([q_0, q_1], b) = [q_0, q_2]$$

To find $\delta^1([q_0,q_2],a$

$$\delta(\{q_0,q_2\},a) = \delta(q_0,a) \quad \delta(q_2,a)$$
$$= \{q_0,q_1\} \quad \{q_0\}$$
$$= \{q_0, q_1\}$$
$$\delta^1([q_0,q_2],a) = [q_0,q_1]$$

To find $\delta^1([q_0,q_2],b$

$$\delta(\{q_0,q_2\},b) = \delta(q_0,b) \quad \delta(q_2,b)$$
$$= \{q_2\} \quad \{q_1,q_2\}$$
$$= \{q_0, q_1\}$$
$$= [q_1,q_2]$$
$$\delta^1([q_0,q_2],b) = [q_1,q_2]$$

To find $\delta^1([q_1,q_2],a$

$$\delta(\{q_1,q_2\},a) = \delta(q_1,a) \quad \delta(q_2,a)$$
$$= \{q_1\} \quad \{q_0\}$$
$$= \{q_0, q_1\}$$
$$\delta^1([q_1,q_2],a) = [q_0,q_1]$$

To find $\delta^1([q_1,q_2],b$

$$\delta(\{q_1,q_2\},b) = \delta(q_1,b) \quad \delta(q_2,b)$$
$$= \{q_0\} \quad \{q_1,q_2\}$$
$$= \{q_0, q_1, q_2\}$$
$$\delta^1([q_1,q_2],b) = [q_0,q_1,q_2]$$

To find $\delta^1([q_0,q_1,q_2],a$

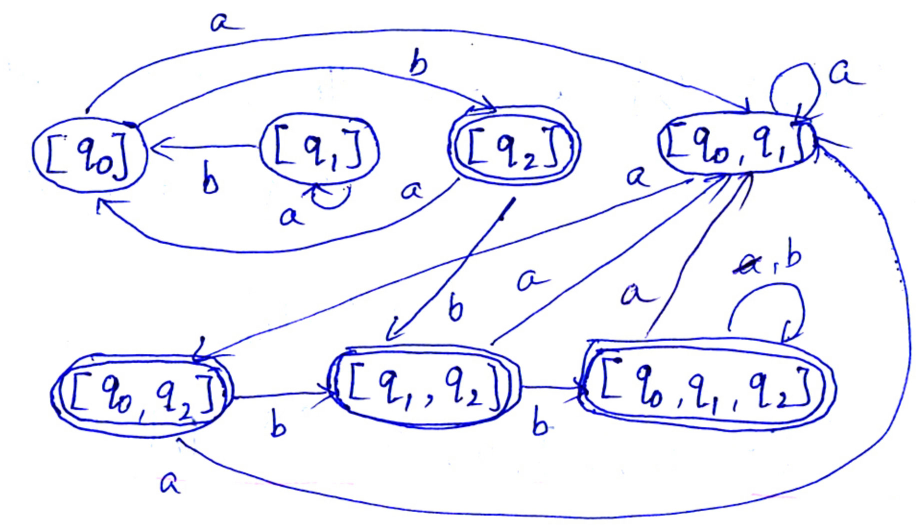
$$\delta(\{q_0,q_1,q_2\},a) = \delta(q_0,a) \quad \delta(q_1,a) \quad \delta(q_2,a)$$
$$= \{q_0,q_1\} \quad \{q_1\} \quad \{q_0\}$$
$$= \{q_0, q_1,\}$$
$$\delta^1([q_0,q_1,q_2],a) = [q_0,q_1,]$$

To find $\delta^1([q_0,q_1,q_2],b)$

$$\delta(\{q_0,q_1,q_2\},b) = \delta(q_0,b) \quad \delta(q_1,b) \quad \delta(q_2,b)$$
$$= \{q_2\} \quad \{q_0\} \quad \{q_1,q_2\}$$
$$= \{q_0, q_1, q_2\}$$
$$\delta^1([q_0,q_1,q_2],b) = [q_0,q_1,q_2]$$

	a	b
q_0	$[q_0, q_1]$	$[q_2]$

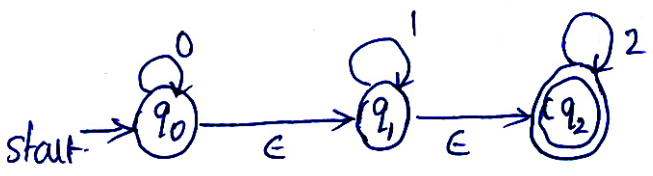
q_1	$[q_1]$	$[q_0]$
$[q_2]$	$[q_0]$	$[q_1, q_2]$
$[q_0, q_1]$	$[q_0, q_1]$	$[q_0, q_2]$
$[q_0, q_2]$	$[q_0, q_1]$	$[q_1, q_2]$
$[q_1, q_2]$	$[q_0, q_1]$	$[q_0, q_1, q_2]$
$[q_0, q_1, q_2]$	$[q_0, q_1]$	$[q_0, q_1, q_2]$



FINITE AUTOMATA WITH ∈-MOVES

Finite Automata with ∈ Moves :

- ✶ The NFA may be extended to include transitions on the empty input ∈
- ✶ For eg.



- We say an NFA accepts a string w if there is some labeled w from the initial state to a final state of course edges labeled $\in$ may be included in the path although the $\in$'s do not appear explicitly in w.
- For eg. the word 002 is accepted by the NFA by the path $q_0 - q_0 - q_0 - q_1 - q_2 - q_2$ with arcs labeled 0, 0, $\in, \in, 2$

Formal definition :

- A NFA with $\in -$ moves to be a 5 – tuple,

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

  δ is the transition function maps

  $Q \times (\Sigma \quad \{\in\}$ to 2^Q
- $\delta(q, a)$ will consist of all states P such that there is a transition labeled a from q to p where a is either $\in$ or a symbol in Σ

	0	1	2	$\in$
q_0	$\{q_0\}$	ϕ	ϕ	$\{q_1\}$
q_1	ϕ	$\{q_1\}$	ϕ	$\{q_2\}$
$[q_2]$	ϕ	ϕ	$\{q_2\}$	ϕ

$\in$ - closue (q)

$\in$-closue (q) is the set of all vertices p such that there is a path from q to p labeled

Example :

$\in$ - closure (q_0) = $\{q_0, q_1, q_2\}$

- ie) the path consisting of q_0 alone is a from q_0 to q_0 with all arcs labeled $\in$.
- Path $q_0 - q_1$, shows that q_1 is in $\in -$ closure. (q_0)
- Path $q_0 - q_1 - q_2$ shows that $q_2$is in Σ-closure.
- $\hat{\delta}$ can be defined as follows :

  1. $\hat{\delta}(q, \in) = \in - \text{closure}(q)$
  2. for w in Σ^* and a in Σ

     $\hat{\delta}$ (q, wa) = $\in$ - closure (p)

     Where p = $\{p \mid \}$

     For some p in $\delta(\hat{\delta}(q, w)$
  3. $\delta(R, a) = \quad \delta(q, a)$

     q in R
  4. $\hat{\delta}(R, W) = \quad \hat{\delta}(q, w)$

q in R

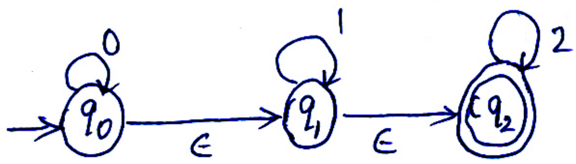
- ✶ We can define L (M) as

  The language accepted by M (Q,Σ,δ,q_0,F) to be

  $\{w|$ ^{q} $\hat{\delta}(q,w)$ contains a state in F$\}$

- ✶ Example :

  Consider the NFA



Find $\hat{\delta}(q_0, q_1)$

$\hat{\delta}$ $(q_0, 01) = \in-\text{closure}(\delta(\hat{\delta}(q_0,0),1)$

$\hat{\delta}$ $(q_0, 0) = \in-\text{closure}(\delta(\hat{\delta}(q_0,\in),0)$

$\hat{\delta}$ $(q_0,$ $^{\in}$ $01) = \in-\text{closure}(q_0)$

$= \{q_0, q_1, q_2\}$

$\hat{\delta}$ $(q_0, 01) = \in-\text{closure}(\delta(\hat{\delta}(q_0,\in),0)$

$= \in-\text{closure}(\delta(\{q_0,q_1,q_2),0)$

$= \in-\text{closure}(\delta(q_0,0) \quad \delta(q,0) \quad \delta(q2,0))$

$= \in-\text{closure}(\delta\{q_0\} \quad \phi \quad \phi)$

$= \{q_0, q_1, q_2\}$

$\hat{\delta}$ $(q_0, 01) = \in-\text{closure}(\delta(\hat{\delta}(q_0,0),1)$

$= \in-\text{closure}(\delta(\{q_0,q_1,q_2),1)$

$= \in-\text{closure}(\delta(q_0,1) \quad \delta(q_1,1) \quad \delta(q_2,1))$

$= \in-\text{closure}(\phi \quad \{q_1\} \quad \phi)$

$= \in-\text{closure}\{q_1\}$

$= \{q_1, q_2\}$

EQUIVALENCE OF NFA's WITH AND WITHOUT ∈-MOVES

- ❖ **Prove that "A language accepted by some NFA with ∈- moves iff L is accepted by without ∈- moves NFA**

Theorem :

If L is accepted by an NFA with ∈-transitions then L is accepted by an NFA without ∈- transition.

Proof :

Let M = (Q, $\Sigma, \delta, q_0, F)$ be an NFA with $\in$- transitions.

Now we have to define NFA without $\in$- move M'

$M^1 = (Q^1, \Sigma^1, \delta^1, q^0, F^1)$

Where $F^1 = F \cup \{q0\}$ if $\in$- closure (q_0) contains a state of F

$= F$ Otherwise

- ✶ It is easy show by induction on the length of the input string .We have to prove

$\delta^1(q_0, \boldsymbol{x}) = \hat{\delta}(q_0, \boldsymbol{x})$

- ✶ This statement may not be true for $\boldsymbol{x} = \in$

$\delta^1(q0, \in) = \{q_0\}$

while $\delta(q_0, \in) = \in -\text{closure}(q_0)$

- ✶ Therefore we begin our induction at 1

Basis :

$|x| = 1$

Then x is a symbol a

$\delta 1(q_0, a) = \delta(q_0, a)$ by the definition of δ^1

Induction :

$|x| > 1$ Let x = wa

$\delta 1(q0, wa) = \delta 1(\delta(q0, w), a)$

$= \delta^1(p, a)$

$= \delta^1(q, a)$

q in p

$= \delta(q, a)$

q in p

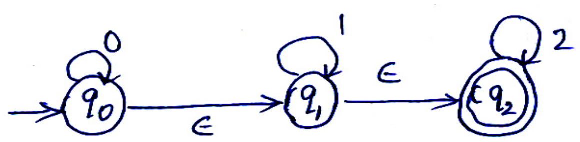
$= \delta(p, a)$

$= \delta(\hat{\delta}(q_0, w), a = \hat{\delta}(q_0, wa)$

<u>CONVERSION OF NFA WITH $\in$-MOVES TO NFA WITHOUT $\in$-MOVES</u>

- ❖ **CONVERT THE NFA WITH $\in$-MOVES TO NFA WITHOUT $\in$-MOVES**

Consider the NFA with $\in$ - moves



Find an equivalent NFA without $\in$- moves

Solution :

Given NFA with $\in$- moves

$M = (\{q_0, q_1, q_2\}, \{0, 1, 2, \in\}, \delta, q_0, \{q_2\})$

Now we have to define NFA without $\in$- move

$M1 = (Q1, \Sigma, \delta 1, q0, F1)$

$Q = \{q_0, q_1, q_3\}$

$\Sigma = \{0, 1, 2\}$

$F^1 = \{q_0, q_2\}$ $\in$- closure (q0) = {q0, q1, q2} contains a state of F.

δ^1

	0	1	2
q_0	$\{q_0, q_1, q_2\}$	$\{q_0, q_1\}$	$\{q_2\}$
q_1	ϕ	$\{q_1, q_2\}$	$\{q_2\}$
q_2	ϕ	ϕ	$\{q_2\}$

$\in-\text{closure}(q_1) = \{q_1, q_2\}$

$\in-\text{closure}(q_2) = \{q_2\}$

$\delta(q0, \in) = \in-\text{closure}(q_0)$

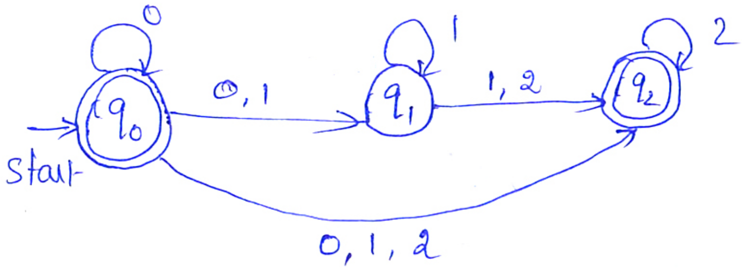
$= \{q_0, q_1, q_2\}$

$$
\begin{aligned}
\delta^1(q0, 0) &= \in-\text{closure}(\delta(q_0, 0)) \\
&= \in-\text{closure}(\delta(\delta(q_0, \in), 0) \\
&= \in-\text{closure}(\delta(\{q_0, q_1, q_2\}, 0) \\
&= \in-\text{closure}(\delta(q_0, 0) \quad \delta(q_1, 0) \quad \delta(q_2, 0)) \\
&= \in-\text{closure}(\delta(\{q_0\} \quad \phi \quad \phi) \\
&= \{q_0, q_1, q_2\}
\end{aligned}
$$

$$
\begin{aligned}
\delta^1(q_0, 1) &= \in-\text{closure}(\delta(q_0, 1)) \\
&= \in-\text{closure}(\delta(\delta(q_0, \in), 1) \\
&= \in-\text{closure}(\delta(\{q_0, q_1, q_2\}, 1) \\
&= \in-\text{closure}(\delta(q_0, 1) \quad \delta(q_1, 1) \quad \delta(q_2, 1)) \\
&= \in-\text{closure}(\phi \quad (\{q_1\} \quad \phi) \\
&= \{q_1, q_2\}
\end{aligned}
$$

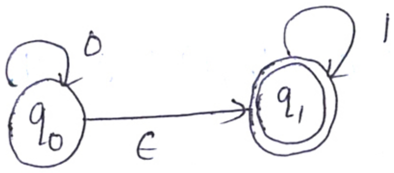
$$
\begin{aligned}
\delta^1(q_0, 2) &= \in-\text{closure}(\delta(q_0, 2)) \\
&= \in-\text{closure}(\delta(\delta(q_0, \in), 2) \\
&= \in-\text{closure}(\delta(\{q_0, q_1, q_2\}, 2) \\
&= \in-\text{closure}(\delta(q_0, 2) \quad \delta(q_1, 2) \quad \delta(q_2, 2))
\end{aligned}
$$

$$= \in -\text{closure}(\phi \quad \phi \quad \{q_2\}$$
$$= \{q_2\}$$
$$\delta^1(q_1,0) \quad = \in -\text{closure}(\delta(q_1,0))$$
$$= \in -\text{closure}(\delta(\delta(q_1,\in),0)$$
$$= \in -\text{closure}(\delta(\{q_1,q_2\},0)$$
$$= \in -\text{closure}(\delta(q_1,0) \quad \delta(q_2,0))$$
$$= \in -\text{closure}(\phi \quad \phi)$$
$$= \phi$$
$$\delta^1(q_1,0) \quad = \in -\text{closure}(\delta(q_1,0))$$
$$= \in -\text{closure}(\delta(\delta(q_1,\in),1)$$
$$= \in -\text{closure}(\delta(\{q_1,q_2\},1)$$
$$= \in -\text{closure}(\delta(q_1,1) \quad \delta(q_2,1)))$$
$$= \in -\text{closure}(\{q_1\} \quad \phi)$$
$$= \{q_1, q_2\}$$
$$\delta^1(q_1,2) \quad = \in -\text{closure}(\delta(q_1,2))$$
$$= \in -\text{closure}(\delta(\delta(q_1,\in),2)$$
$$= \in -\text{closure}(\delta(\{q_1,q_2\},2)$$
$$= \in -\text{closure}(\delta(q_1,2) \quad \delta(q_2,2))$$
$$= \in -\text{closure}(\phi \quad \{q_2\})$$
$$= \{q_2\}$$
$$\delta^1(q_2,0) \quad = \in -\text{closure}(\delta(q_2,\in),0)$$
$$= \in -\text{closure}(\delta(\{q_2),0)$$
$$= \in -\text{closure}(\phi)$$
$$= \phi$$
$$\delta^1(q_2,1) \quad = \in -\text{closure}(\delta(q_2,\in),1)$$
$$= \in -\text{closure}(\delta(\{q_2),1)$$
$$= \in -\text{closure}(\phi)$$
$$= \phi$$
$$\delta^1(q_2,2) \quad = \in -\text{closure}(\delta(q_2,\in),2)$$
$$= \in -\text{closure}(\delta(\{q_2),2)$$
$$= \in -\text{closure}(q_2)$$
$$= \phi$$



Example 2 :

Construct a NFA without $\in$- moves from NFA with $\in$- moves



Solution :

$\in-$closure $(q_0) = \{q_0, q_1\}$

$\in-$closure $(q_1) = \{q_1\}$

Let Given NFA with $\in-$moves

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

We have to find M', NFA without $\in-$move

$M = (Q, \Sigma, \delta^1, q_0, F^1)$

$F1 = \{q0, q1\}$ $\in -$closure$(q0)$ contains a state in F

$F^1 = F \quad \{q_0\}$

$\delta^1 (q_0, \in) = \in -$closure$(q_0)$

$= \{q_0, q_1\}$

$\delta^1 (q_0, 0) = \in -$closure$(\delta^1 (\delta^1 (q_0, \in), 0)$

$= \in -$closure$(\delta^1 (\{q_0, q_1,\}, 0)$

$= \in -$closure$(\delta(\{q_0, 0) \quad \delta(q_0, 0)$

$= \in -$closure $(\{q_0), \quad \phi)$

$= \{q_0, q_1\}$

$\delta^1 (q_0, 1) = \in -$closure$(\delta^1 (\delta^1 (q_0, \in), 1)$

$= \in -$closure$(\delta^1 (\{q_0, q_1,\}, 1)$

$= \in -$closure$(\delta(\{q_0, 1) \quad \delta(q_1, 1)$

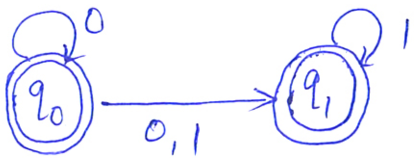
$= \in -$closure $(\phi \quad \{q_1))$

$= \{q_1\}$

$\delta^1 (q_1, 0) = \in -$closure$(\delta^1 (\delta^1 (q_1, \in), 0)$

$$
\begin{aligned}
&= \in\!-\text{closure}(\delta^1(\{q_1\}),0)\\
&= \in\!-\text{closure}(\phi)\\
&= \phi\\
\delta^1(q_1,1) &= \in\!-\text{closure}(\delta^1(\delta^1(q_1,\in),1)\\
&= \in\!-\text{closure}(\delta^1(\{q_2\}),1)\\
&= \in\!-\text{closure}(q_1)\\
&= \{q_1\}
\end{aligned}
$$

	0	1
q_0	$\{q_0, q_1\}$	$\{q_1\}$
q_1	ϕ	$\{q_1\}$



EQUIVALENCE OF FINITE AUTOMATA AND REGULAR EXPRESSIONS

Equivalence of finite automata and regular expressions :

- ✶ The languages accepted by finite automata are the languages denoted by regular expressions.
- ✶ For every regular expression there is an equivalent NFA with $\in$- transitions.

Theorem 1:

Let r be a regular expression. Then there exists an NFA with $\in$- transitions that accepts L (r)

Proof :

We show by induction on the number of operator in the regular expression r that

there exists an NFA with $\in$- transitions having one final state and no transitions out of this final state such that

L (M) = L (r)

Basis :

For regular expressions having zero operators.

- ✶ The expression r must be $\in, \phi$, or a for some a in Σ

Induction :

One or more operators in the regular expression.

- ✶ Assure that the theorem is true for regular expression with fewer that I operators.

  $i \geq 1$

- ✶ Let r have I operators.

Case 1 :

$r = r1 + r2$

- ✶ Both r_1 and r_2 must have fewer than i operator
- ✶ Thus there are 2 NFA's

  $M1 = (Q_1, \Sigma_1, \delta_1, q_1, \{f_1\}$ with

  $= L(M_1) = L(r_1)$

- ✶ Assume Q_1 and Q_2 are disjoint

  Let $q_0 \rightarrow$ be a new initial state

  $f_0 \rightarrow$ be a new initial state

- ✶ Now we can construct NFA with

  $M = Q1, \quad Q2 \quad \{q0, f0\}, \Sigma, \quad \Sigma 2, \delta, q0, \{f0\})$

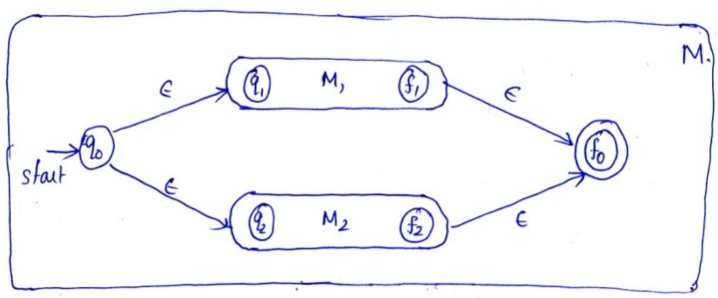
where δ is defined by

(i) $\delta(q_0, \in) = \{q_1, q_2\}$

(ii) $\delta(q, a) = \delta_1(q, a)$ for q in $Q_1 - \{f_1\}$ and a in $\Sigma_1 \quad \{\in\}$

(iii) $\delta(q, a) = \delta_2(q, a)$ for q in $Q_2 - \{f_2\}$ and a in $\Sigma_2 \quad \{\in\}$

(iv) $\delta_1(f_1, \in) = \delta_2(f_2, \in) = \{f_0\}$



- ✶ Hence

  $L(M) = L(M_1) \cup L(M_2)$

Any path in the transition diagram of M from q_0 to f_0 must begin by going to either q_1 or q_2 on $\in$. If the path goes to q_1 it may follow any path in M_1 to f_1 and then go to f_0 on $\in$

Hence $L(M) = L(M_1) \cup L(M_2)$

Case 2 :

$r = r_1 r_2$

Let M_1 and M_2 be 2 NFA's

$M_1 = Q_1, \Sigma_1, \delta_1, q_1, \{f_1\})$ with

$L(M_1) = L(r_1)$

and

$M_2 = Q_2, \Sigma_2, \delta_2, q_2, \{f_2\})$ with

$L(M_2) = L(r_2)$

- ✶ Assume Q1 and Q2 are disjoint
- ✶ No we can construct NFA with

$L(M) = L(r)$

$M = (Q, \quad Q_2, \Sigma, \quad \Sigma_2, \delta, \{q_1\}, \{f_2\})$

where

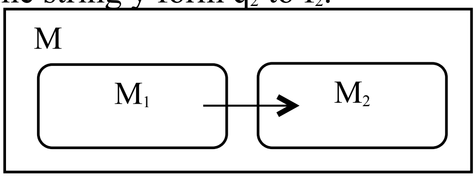
δ is defined as

(i) $\delta(q,a) = \delta_1(q_1,a)$ for q in $Q_1 - \{f_1\}$ and a in $\Sigma_1 \quad \{\in\}$

(ii) $\delta(q,a) = \delta(q_1,a)$ for q in $Q_2 - \{f_1\}$ and a in $\Sigma_2 \quad \{\in\}$

(iii) $\delta(f_1, \in) = \{q_2\}$

- ✶ Every path in M from q1 to f_2 is a path labeled by some string *x* form q1 to f_1 followed by the edge from f1 to q_2 labeled $\in$ followed by a path labeled by some string y form q_2 to f_2.



- ✶ Thus

$L(M) = L(M_1)\, L(M_2)$

ie. $L(M) = \{xy \mid x \text{ is in } L(M_1) \text{ and } y \text{ is in } L(M_2)\}$

Case (iii)

$r = r_1^*$

Let $M_1 = (Q_1, \Sigma_1, \delta_1, q_1, \{f_1\})$ with

$L(M_1) = L(r_1)$

Now we can construct NFA

Let $M_1 = (Q_1 \{q_0, f_0\}, \Sigma_1, \delta_1, q_0, \{f_0\})$ with

$L(M) = L(r)$

where

$q_0 \rightarrow$ be an new initial state

$f_0 \rightarrow$ be an new final state

and δ is given by

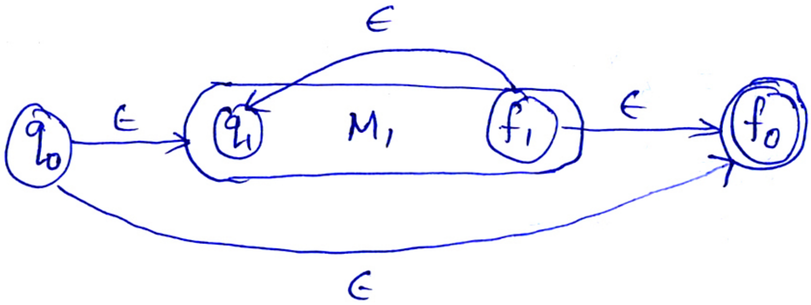
(i) $\delta(q_0, \in) = \delta(f_1, \in) = \{q_1, f_0\}$

(ii) $\delta(q,a) = \delta_1(q,a)$ for q in $Q_1 - \{f_1\}$ and a in $\Sigma_1 \quad \{\in\}$

- ✶ Any path from q_0 to f_0 consists either of a path from q_0 to q_1 on $\in$ followed by some number of paths from q_1 to $f_1$and back to q_1 on $\in_1$ each labeled by a string in $L(M_1)$ followed by a path from q_1 to f_1 on a string in $L(M_1)$ then to f0 on $\in$

Hence :

$L(M) = L(M_1)$



Theorem 2

If L is accepted by DFA, then L is denoted by the regular expression.

Proof :

Let L be the set accepted by accepted by the DFA

$M = (\{q_1, \ldots\ldots\ldots, q_n\}, \Sigma, \delta, q_1, F)$

Let

$R_{ij}^{k} \rightarrow$ denote the set of all string x such that $\delta(q_i, \boldsymbol{x}) = q_j$ and $\delta(q_i, y) = q_l$ for any that is a prefix of x, then $l = k$

i.e.) R_{ij}^{k} = is the set of all strings that take the FA from state qi to qj without going though any state numberd higher than k.

- ✶ we can define R_{ij}^{k} = is the set of all strings that take the FA from state q_i to q_j without going through any state numbered higher than k.
- ✶ We can define R_{ij}^{k} recursively.

$$R_{ij}^{k} = R_{ik}^{k-1}(R_{kk}^{k-1})^* R_{kj}^{k-1} \quad R_{ij}^{k-1}$$

- ✶ The inputs of R_{ij}^{k} are, either

1) in R_{ij}^{k-1}

(or)

2) Composed of a string R_{ik}^{k-1} followed by zero of more strings in R_{kk}^{k-1} followed by a string R_{kj}^{k-1}

- ✶ We must show that for each i, j and k, there exists a regular expression r_{ij}^{k} denoting the language R_{ij}^{k}
- ✶ We can show by induction on k.

Basis :

$k = 0$

R^{0}_{ij} is a finite set of strings, each of which is either $\in$ or a single symbol.

Induction :

- ✶ The recursive formula R^{k}_{ij} involves only the regular expression operators
  a) union
  b) concatenation
  c) closure
- ✶ By the induction hypothesis, for r^{k}_{ij}, we may select the regular expression
  $r^{k}_{ij}, = r^{k-1}_{ij} (r^{k-1}_{kk})^* r^{k-1}_{kj} + r^{k-1}_{ij}$
- ✶ We can conclude

$$R^{n}_{ij}$$

L (M) = q_j in F

Thus L (M) is denoted by the regular expn.

$r^{n}_{ij1} + r^{n}_{ij2} + ... + r^{n}_{ijp}$ where $F = \{q_{ji}, q_{j2}, ... q_{jp}\}$

CONVERSION OF REGULAR EXPRESSION TO FINITE AUTOMATA

Construct an NFA for the regular express on 01* + 1 (or) (0 (1*)) + 1

Solution :

Given regular expression

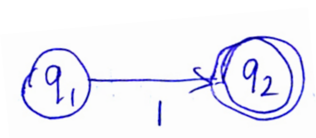
r = 01* + 1

It is of the form $r = r_1 + r_2$ where

$r_1 = 01^*$

$r_2 = 1$

The NFA for $r_2 = 1$



Now r_1 is of the form

$r1 = r_3 r_4$

where $r_3 = 0$

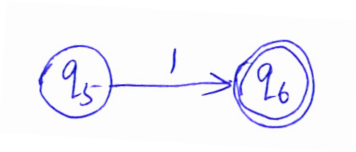
$r_4 = 1^*$

The NFA for $r_3 = 0$

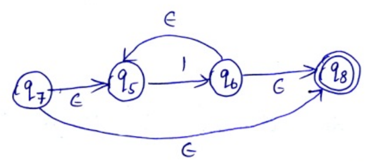
$r_4 = r_5{*}$

$r_5 = 1$

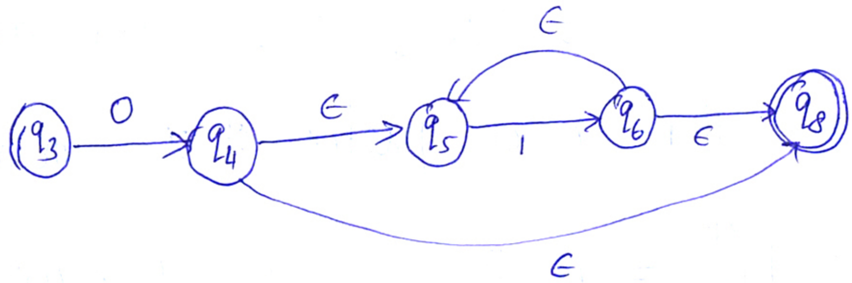
The NFA for $r_5 = 1$



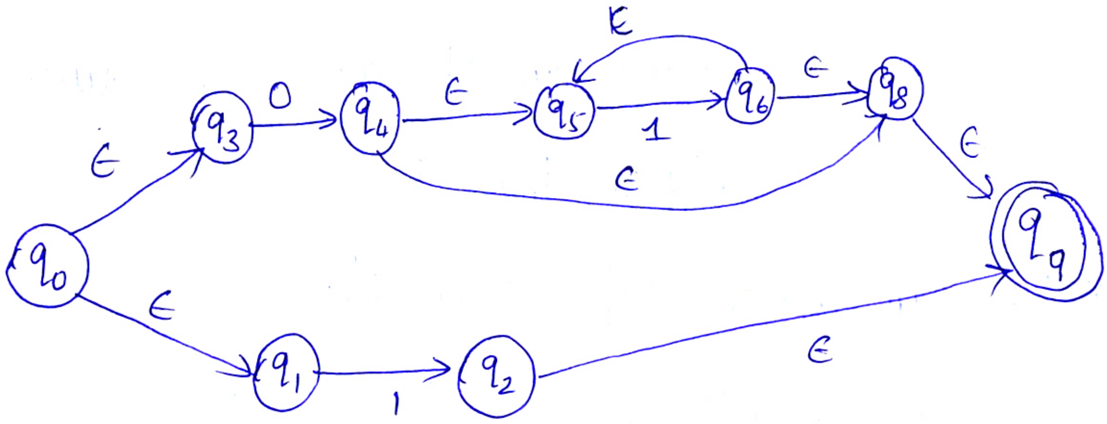
The NFA for $r_4 = r_5{*} = 1{*}$



The NFA for $r_1 – r_3r_4 = 01{*}$



The NFA for $r = r_1 + r_2$



<u>CONVERSION OF FINITE AUTOMATA TO REGULAR EXPRESSION</u>

Convent the following DFA to a regular expression

Solution :

Find R_{ij}^{k} for i, j, k

Let k = 0

$$R_{ij}^{o}=\begin{cases}\{a|\delta(q_i,a)=q_j & \text{if } i\neq j\\ \{a|\delta(q_i,a)=q_j\} \quad \{\in\} & \text{if } i=j\end{cases}$$

$r_{11}^{0}=\in+1$

$r_{12}^{0}=0$

$r_{21}^{0}=\phi$

$r_{22}^{0}=\in+0+1$

$R_{ij}^{k},=R_{ik}^{k-1}(R_{kk}^{k-1})^{*}R_{kj}^{k-1} \quad R_{ij}^{k-1}$

$r_{ij}^{k}=r_{ik}^{k-1}(r_{kk}^{k-1})^{*}r_{kj}^{k-1}+r_{ij}^{k-1}$

For K = 1

$R_{11}^{(1)}=r_{11}^{0}(r_{11}^{0})^{*}r_{11}^{0}+r_{11}^{0}$

$=(\in+1)(\in+1)^{*}(\in+1)+(\in+1)$

$=1^{*}+(\in+1)$

$=1^{*}$

$r_{12}^{(1)}=r_{11}^{0}(r_{11}^{0})^{*}r_{12}^{0}+r_{12}^{0}$

$=(\in+1)(\in+1)^{*}0+0$

$=1^{*}+0$

$=1^{*}0$

$r_{21}^{(1)}=r_{21}^{0}(r_{11}^{0})^{*}r_{11}^{0}+r_{21}^{0}$

$=\phi(\in+1)^{*}(\in+1)+\phi$

$=\phi 1^{*}+\phi$

$=\phi$

$r_{22}^{(1)}=r_{21}^{0}(r_{11}^{0})^{*}r_{12}^{0}+r_{22}^{0}$

$=\phi(\in+1)^{*}(0)+\in+0+1$

$=\phi 1^{*}+0+\in+0+1$

$=\phi+\in+0+1$

$=\in+0+1$

Let K = 2

$r_{11}^{(2)}=r_{12}^{1}(r_{22}^{1})^{*}r_{21}^{1}+r_{11}^{1}$

$=1^{*}0\,(\in+0+1)^{*}\phi+1^{*}$

$=\phi+1^{*}$

$=1^{*}$

$r_{12}^{(2)}=r_{12}^{1}(r_{22}^{1})^{*}r_{22}^{1}+r_{12}^{1}$

$= 1^*0\ (\in+0+1)^*(\in+0+1)+1^*$

$= 1^*0(0+1)^*+1^*0$

$= 1^*0(0+1)^*$

$r_{21}^{(2)} = r_{22}^{1}\ (r_{22}^{1})^* r_{21}^{1} + r_{21}^{1}$

$= (\in+0+1)(\in+0+1)^*\phi+\phi$

$= (0+1)^*\ \phi+\phi$

$=\phi$

$r_{22}^{(2)} = r_{22}^{1}\ (r_{22}^{1})^* r_{22}^{1} + r_{22}^{1}$

$= (\in+0+1)(\in+0+1)^*(\in+0+1)+(\in+0+1)$

$= (0+1)^* + (\in+0+1)$

$= (0+1)^*$

L (M) = $r_{12}^{(2)}$

= 1* 0 (0 + 1)*

$\boxed{L\ (M) = 1^*\ 0\ (0+1)^*}$

MINIMIZATION OF DFA

Minimization of DFA :

- There is unique minimum state DFA for every regular set.

Theorem :

- The minimum state automata accepting a language L is unique upto an isomorphism ie) renaming of the states.

Proof :

Any DFA M = $(Q, \Sigma, \delta, q_0, F)$ accepting L defines an equivalence relation that is a refinement of R_L

- Thus the number of states of M is greater than or equal to the number of states of M^1
- If equality holds then each of the states of M can be identified with one of the states of M^1
- Let q be a state of M there must be some *x* in Σ^* such that $\delta(q_0, \boldsymbol{x}) = q$, otherwise q would be removed from Q
- Identify q with the state $\delta^1(q_0{}^1, \boldsymbol{x})$ of M^1
- The identification will be consistent.
- If $\delta(q_0, \boldsymbol{x}) = \delta(q_0, y) = q$, then x and y are in the same equivalence class of R_L.

  Thus $\delta^1(q_0{}^1, \boldsymbol{x}) = \delta^1(q_0{}^1, y)$

A Minimization Algorithm :

- ✶ There is a simple method for finding the minimum state DFA M^1, equivalent a given DFA $M = (Q, \Sigma, \delta, q0, F)$
- ✶ Let $\equiv$ be the equivalence relation on the states of M.
- ✶ If $P \equiv q$, we say p is equivalent to q.

  $P \equiv q$ iff for each input string x

  $\delta(p, x)$ is an accepting state.

  $\delta(q, x)$ is an accepting state.
- ✶ We say that p is distinguishable from q if there exists on x such that

  $\delta(p, x)$ is an F and

  $\delta(q, x)$ is not in F and vice versa.

Procedure :

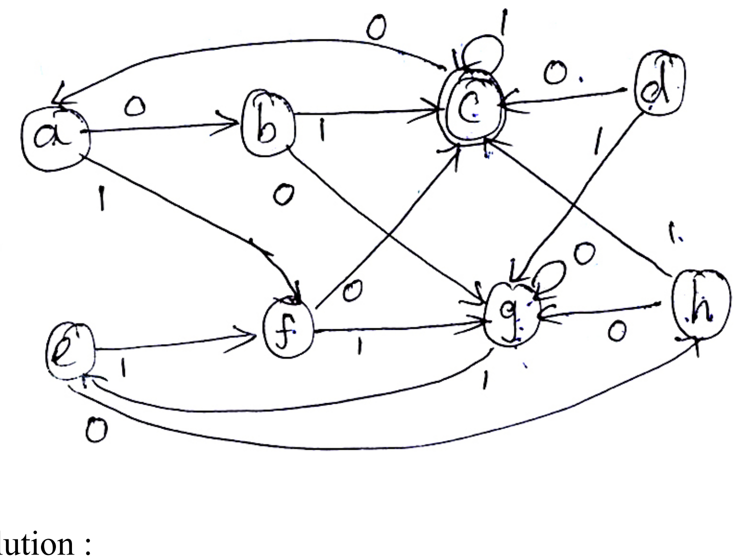
- ✶ An X is placed in the table each time a of states that cannot be equivalent.
- ✶ Initially an X is placed in each entry corresponding to one find state and one non – final state.
- ✶ Next for each pair of states P and q that are not distinguishable, we consider.

  $r = \delta(p, a)$

  $s = \delta(q, a)$ for each input symbol a
- ✶ If states r and s have been shown to be distinguishable by some string x, then p and q are distinguishable by string ax.
- ✶ Thus if the string (r,s) in the table has an x, an x is placed at the entry (p, q)

Example :

Minimize the following DFA



Solution :

b	x		
c	x	x	
d	x	x	x

	a	b	c	d	e	f	g
e		x	x	x			
f	x	x	x		x		
g	x	x	x	x	x	x	
h	x		x	x	x	x	x

- Intially an x placed in each entry corresponding to one final state and one final state.
  The entries
  (c, a), (c, b), (c, d), (c, e), (c, f), (c, g), (c, h)
- Now the unmarked pairs are
  (a, b) (b, d) (d, e) (l, f) (f, g) (g, h)
  (a, d) (b, e) (d, f) (e, g) (f, h)
  (a, e) (b, f) (d, g) (e, b)
  (a, f) (b, g) (d, b)
  (a, h)

(a, b)

$$\left.\begin{array}{l}\delta(a,0)=b\\ \delta(b,0)=g\end{array}\right\}\text{ (b, g) is unmarked}$$

$$\left.\begin{array}{l}\delta(a,1)=f\\ \delta(b,1)=c\end{array}\right\}\text{ (f, c) is marked, so mark (a, b)}$$

(a, b)

$$\left.\begin{array}{l}\delta(a,0)=b\\ \delta(d,0)=c\end{array}\right\}\text{ (b, c) is marked so mark (a, d)}$$

(a, l)

$$\left.\begin{array}{l}\delta(a,0)=b\\ \delta(l,0)=h\end{array}\right\}\text{ (b, g) is unmarked}$$

$$\left.\begin{array}{l}\delta(a,1)=f\\ \delta(l,1)=f\end{array}\right\}\text{ (f, f) is unmarked}$$

(a, f)

$$\left.\begin{array}{l}\delta(a,0)=b\\ \delta(f,0)=c\end{array}\right\}\text{ (b, c) is marked so mark (a, f)}$$

(a, g)

$\left.\begin{array}{l}\delta(a,0)=b\\ \delta(g,0)=g\end{array}\right\}$ (b, g) is unmarked

$\left.\begin{array}{l}\delta(a,1)=f\\ \delta(g,1)=e\end{array}\right\}$ (f, e) is unmarked

(a, h)

$\left.\begin{array}{l}\delta(a,0)=b\\ \delta(h,0)=g\end{array}\right\}$ (b, g) is unmarked

$\left.\begin{array}{l}\delta(a,1)=f\\ \delta(h,1)=c\end{array}\right\}$ (f, c) is marked. So mark (a, h)

(b, d)

$\left.\begin{array}{l}\delta(b,0)=g\\ \delta(d,0)=c\end{array}\right\}$ (g, c) is marked. so mark (b, d)

(b, e)

$\left.\begin{array}{l}\delta(b,0)=g\\ \delta(e,0)=h\end{array}\right\}$ (g, h) is unmarked

$\left.\begin{array}{l}\delta(b,1)=c\\ \delta(e,1)=f\end{array}\right\}$ (c, f) is marked. So mark (b, e)

(b, f)

$\left.\begin{array}{l}\delta(b,0)=g\\ \delta(f,0)=c\end{array}\right\}$ (g, c) is marked. So mark (b, f)

(b, g)

$\left.\begin{array}{l}\delta(b,0)=g\\ \delta(g,0)=g\end{array}\right\}$ (g, g) is unmarked

$\left.\begin{array}{l}\delta(b,1)=c\\ \delta(g,1)=e\end{array}\right\}$ (e, e) is marked. So place an x in (b, g)

(b, h)

$\left.\begin{array}{l}\delta(b,0)=g\\ \delta(h,0)=g\end{array}\right\}$ (g, g) is unmarked

$\left.\begin{array}{l}\delta(b,1)=c\\ \delta(h,1)=c\end{array}\right\}$ (c, c) is unmarked.

(d, e)

$$\left.\begin{array}{l}\delta(d,0)=c\\ \delta(e,0)=h\end{array}\right\}\text{ (c, h) is unmarked. So place x in (d, e)}$$

(d, f)

$$\left.\begin{array}{l}\delta(d,0)=c\\ \delta(f,0)=c\end{array}\right\}\text{ (c, c) is unmarked}$$

$$\left.\begin{array}{l}\delta(d,1)=g\\ \delta(f,1)=g\end{array}\right\}\text{ (g, g) is unmarked.}$$

(d, g)

$$\left.\begin{array}{l}\delta(d,0)=c\\ \delta(g,0)=g\end{array}\right\}\text{ (c, g) is marked. So place an x in (d, g)}$$

(d, h)

$$\left.\begin{array}{l}\delta(d,0)=c\\ \delta(g,0)=g\end{array}\right\}\text{ (c, g) is marked. So place an x in (d, g)}$$

(d, h)

$$\left.\begin{array}{l}\delta(d,0)=c\\ \delta(h,0)=g\end{array}\right\}\text{ (c, g) is marked. So place an x in (d, h)}$$

(e, f)

$$\left.\begin{array}{l}\delta(e,0)=h\\ \delta(f,0)=c\end{array}\right\}\text{ (h, c) is marked. so place x in (e, f)}$$

(e, h)

$$\left.\begin{array}{l}\delta(e,0)=h\\ \delta(h,0)=g\end{array}\right\}\text{ (h, g) is unmarked.}$$

$$\left.\begin{array}{l}\delta(e,1)=f\\ \delta(h,1)=c\end{array}\right\}\text{ (f, c) is marked. So place an x in (e, h)}$$

(e, g)

$$\left.\begin{array}{l}\delta(e,0)=h\\ \delta(g,0)=g\end{array}\right\}\text{ (h, g) is unmarked}$$

$$\left.\begin{array}{l}\delta(e,1)=f\\ \delta(g,1)=e\end{array}\right\}\text{ (f, e) is marked. So place x in (e, g)}$$

(f, g)

$$\left.\begin{array}{l}\delta(f,0)=c\\ \delta(g,0)=g\end{array}\right\}$$ (c, g) is marked. So place an x in (f, g)

(f, h)

$$\left.\begin{array}{l}\delta(f,0)=c\\ \delta(h,0)=g\end{array}\right\}$$ (c, g) is marked. So place an x in (f, h)

(g, h)

$$\left.\begin{array}{l}\delta(g,0)=g\\ \delta(h,0)=g\end{array}\right\}$$ (g, g) is unmarked

$$\left.\begin{array}{l}\delta(g,1)=e\\ \delta(h,1)=c\end{array}\right\}$$ (e, c) is marked. So place x in (g, h)

Now the unmarked place are,

(a, e) (b, h) (d, f) (a, g)

(a, e)

$$\left.\begin{array}{l}\delta(a,0)=b\\ \delta(e,0)=h\end{array}\right\}$$ (b, h) is unmarked

$$\left.\begin{array}{l}\delta(a,1)=f\\ \delta(e,1)=f\end{array}\right\}$$ (f, f) is unmarked.

(a, g)

$$\left.\begin{array}{l}\delta(a,0)=b\\ \delta(g,0)=g\end{array}\right\}$$ (b, g) is marked. So pace an x in (a, g)

(b, h)

$$\left.\begin{array}{l}\delta(b,0)=g\\ \delta(h,0)=g\end{array}\right\}$$ (g, g) is unmarked

$$\left.\begin{array}{l}\delta(b,1)=c\\ \delta(h,1)=c\end{array}\right\}$$ (c, c) is unmarked.

(d, f)

$$\left.\begin{array}{l}\delta(d,0)=c\\ \delta(f,0)=c\end{array}\right\}$$ (c, c) is unmarked

$$\left.\begin{array}{l}\delta(d,1)=g\\ \delta(f,1)=g\end{array}\right\}$$ (g, g) is unmarked.

On completion of the table, we conclude that the equivalent states are

$a \equiv e \qquad b \equiv h \qquad d \equiv f$

Algorithm :

begin

for p in F and q in Q-F do mark (p, q) for each pair of disfinct states (p, q)in F x F or (Q – F) x (Q – F) do

if for some input symbol a

$(\delta(p,a),\delta(q,a)$ is marked then begin mark (p, q)

recursively mark all unkarked pairs on the lists of other paurs that are marked at this step.

end.

else

for all input symbols a do

put (p, q) on the list for $(\delta(p,a),\delta(q,a)$ unless

$(\delta(p,a) = \delta(q,a)$

end.

Method II

For the above same problem, the DFA can be minimized using DFA minimization algorithm.

Solution :

Transition table :

	0	1
a	b	f
b	g	c
c	a	c
d	c	g
e	h	f
f	c	g
f	c	g
g	g	c
h	g	c

Step 1

The pair of status (b, h) are same state transition. So b and h are equivalent state.

$b \equiv h$

	0	1
a	b	f
[b,h]	g	c
*c	a	c
d	c	g
e	h	f
f	c	g
g	c	g

Step 2

The pair of status (d, f) are same state transition.

d≡f

	0	1
a	b	f
[b,h]	g	c
*c	a	c
[d,f]	c	g
e	h	f
f	h	f
g	g	e

⇒

	0	1
a	[b, h]	c
c	a	c
[d, f]	c	g
e	[b, h]	[d, f]
g	g	e

Step e

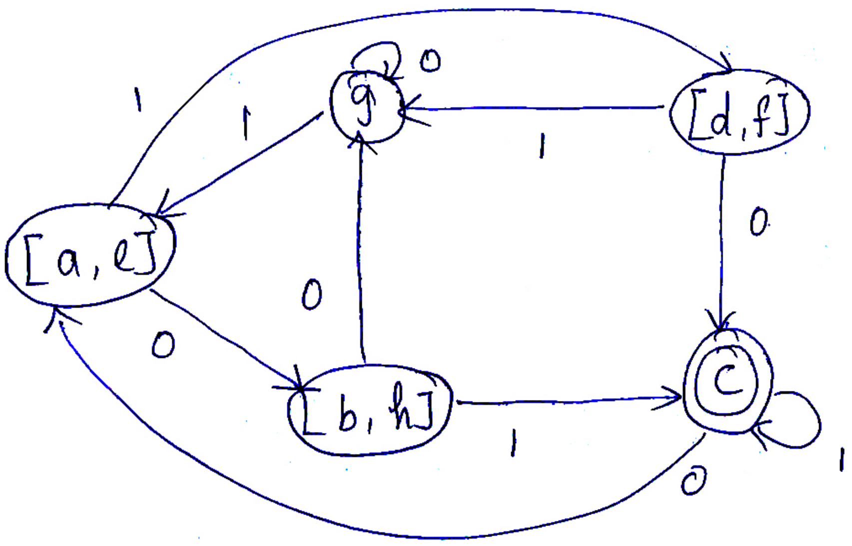
The pair of states (a, e) are same state transition diagram.

∴a≡e

	0	1
[a, e]	[b, h]	[d, f]
[b, h]	g	c
c	[a, e]	c

[d, f]	c	g
g	g	[a, e]

- ✶ Thus the minimized FA is



PUMPING LEMMA FOR REGULAR SETS

Pumping lemma for regular sets :

- ✶ Pumping lemma is a powerful for proving certain languages non-regular.
- ✶ It is also useful for developing algorithms to answer whether the language accepted by FA is finite or infinite.
- ✶ If languages is regular it is accepted by a DFA M = $Q,\Sigma,\delta,q0,F)$ with some particular number of states n.
- ✶ Consider an input of n or more symbols a, a_2, … a_m, $m \geq n$ and for i = 1, 2 … m let $\delta(q_0,a,a_2....a_i)=q_i$
- ✶ It is not possible for each of the n + 1 states q_0, q_1, …..q_n to be distinct, since there are only n different states.
- ✶ Thus there are two integers j and k.
  $O \leq j < k \leq n$, such that $q_j = q_k$
- ✶ The path labeled a, a_2…. a_m in the transition diagram of M.

- ✶ Since j<k, the string a_j + 1 …. a_k is of length atleast 1, and since $k \leq n$, its length is no more than m.
- ✶ If qm is in F, that is a, a_2 … a_m is in L (M) then a, a_2… a_j then a, a2 … a_ja_{k+1} … a_m is also in L (M)
- ✶ Since there is a path from q_0 to q_m that goes through q_j but not around the loop.

$$\delta(q_0, a, \ldots a_j a_{k+1} \ldots a_m) = \delta(\delta(q_0, a_1 \ldots a_j), a_{n+1} \ldots a_m$$
$$= \delta(q_j, a_{k+1} \ldots a_m)$$
$$= \delta(q_k, a_{k+1} \ldots a_m)$$
$$= q_m$$

- ✶ Similary we can go around the loop, as many times.
- ✶ Thus a, .. a_j $(a_{j+1} \ldots a)^i$ $a_{k+1} \ldots a_m$ is in L (M)

Applications of pumbing lemma :

- ✶ The pumping lemma is useful in proving that certain languages are not regular.
  1. Select the language L you wish to prove non-regular and assume that L is regular.
  2. The adversary picks n, the constant mentioned in the pumping lemma.
  3. Select a string z in L. The choice may depend implicitly on the value of n.
  4. The adversary breaks z into u, v and w subject to the constraints that $|uv| \leq n$ and $|v| \geq 1$.
  5. Achieve a contradiction to the pumping lemma by showing for any u, v and w determined by the adversary, that there exists an i for which uvw is not in L. I may then be concluded that L is not regular you selection of i may depend on n, u, v and w.
  6. The formal statement of the pumping lemma,
     $(\forall L)(\Im n)(\forall z)[z \text{ in } L \text{ and} |z| \geq n$ imphess
     and $(\forall_i)$ uviw is in L))]

Lemma :

Let L be a regular set. Then there is constant n, such if z is any word in L and $|z| \geq$ n, we may write z = uvw in such a way that

$|uv| \leq n$

$|v| \geq 1$ and

for all $i \geq 0 \; uv^i w$ is in L

Proof :

Z is a, $a_2 \ldots a_m$

u = $a_1, a_2 \ldots a_j$

v = $a_{j+1} \ldots a_k$

w = $a_{k+1} \ldots a_m$

- ✶ The pumping lemma states that if a regular set contains a long string, z, then it contains infinite set of string of the form uv^iw.

Problems based on pumping lemma :

1. Show that the language L = $\{a^n b^n \mid n \geq 1\}$ is not regular

Solution :

(a) Assume that L = $\{a^n b^n \mid n \geq 1\}$ is a regular languages.

(b) Let n be the integer, ie) no of states in the pumping lemma.

(c) Take one string z from L

$Z = a^i b^i$ such that $|z| \geq n$

(4) Break z into 3 strings.

$z = uvw$

$z = a^i b^i$

and mak the assumptions that

$uv = a^m$

$v = aj$

$w = a^{i-m} b^i$

$= a^i b^i$

our assumption is correct

i.e) $|uv| \leq n\,\varepsilon$

$|v| \geq 1$

since both the conditions are true the string uv^kw is also in L

$uv^kw = uvv^{k-1}w$

$= a^m a^{j(k-1)} a^{i-m} b^i$

$= a^{m+j(k-1)} b^i$

put k = 0

$uv^kw = a^{i-j} b^i \neq a^i b^i$

put k = 2

$uv^kw = a^{i+j} b^i \neq a^i b^i$

Since for k = 0, 2, the strings that does not belong to the language L. So the language is not regular.

2. Show that L $\left\{0^{i^2} \mid i \text{ is an integer } i \geq 1\right\}$ is not regular.

Solution :

1. Assume L = $\left\{0^{i^2} \mid i \text{ is an integer } i \geq 1\right\}$ is regular.
2. Let n be the integer, ie. no. of states in the pumping lemma.
3. Take one string Z

   Let Z = O^{n^2}
4. Break z into z = uvw.

   uv = O^{m2}

   V = O^{j^2}

   W = $O^{n2} - m^2$

   uvw = $O^{m^2} O^{n^2} - m^2$

   $= O^{n^2}$

   Our assumption $|uv| \leq n\,\varepsilon |v| \geq 1$ is correct.

Since both candidates are true, the string uv^kw is also in L

$uv^kw = uvv^{k-1}w.$

$$= O^{m^2} O^{j^2(k-1)} O^{n^2-m^2}$$

$$= O^{m^2} O^{j^2(k-1)} O^{n^2-m^2}$$

for k = 0

$$uvkw = O^{m^2} O^{-j^2} O^{n^2-m^2}$$

$$= O^{n^2} - j^2 \neq O^{n^2}$$

for k = 0, 2 the strings $vu^k w$ does not belong to the language L. so the language is not regular.