Course on

Formal Language & Automata Theory

Department of CSE, GIET University, Gunupur, Odisha

Outline of the Lecture

- Regular set and Regular Expression
- Operators in Regular expressions
- Finite Automata from Regular expression
- Arden's theorem
- Regular expression from Finite Automata
- Pumping Lemma for Regular languages
- Closure properties of Regular languages

Regular set and Regular Expression

Regular Expression(RE): It can be thought of as a algebraic description of DFA and NDFA. It is a sequence of characters that forms a search pattern, mainly for use in pattern matching with strings, or string matching.

Regular Set: The set which can be represented by RE.

Regular language(RL): A language is said to be Regular if there is a RE for that language.(later we will discuss more clearly)

Regular Exp	Regular set
a	{a}
a+b+c	{a,b,c}
ab+ba	{ab,ba}
ab(d+e)	{abd,abe}
a*	{λ,a,aa,aaa……}
abc*de	(abde,abcde,abccde,…}
(a+b)*	{λ,a,b,aa,ab,ba,bb,aaa,aab,bba……..}

Regular Expression(RE):

If Σ is an alphabet, then the class of RE over Σ is defined as follow

1. The symbol Ɛ and φ are RE
2. If a ∈ Σ then a is a RE
3. Union of two RE is a RE(R1+R2)
4. Concatenation of two RE is RE ie (R1R2)
5. If R is a RE then R* is a RE.
6. If R is a RE then (R) is RE
7. Nothing else will be RE unless it is constructed from point 1 to point 6

Example:

Note: a+b means either a or b

1. Write the RE for the language over Σ =(0),that recognize even number of 0
2. Write the RE for the language over Σ=(1) that recognize odd number of 1
3. Write the RE for the language over Σ=(0,1) that recognize even 0 followed by even 1.
4. Write the RE for the language over Σ=(0,1) that recognize substring 010

Note: write the RE for all DFA which we designed.

Write the RE over Σ= (0, 1) for the following

1. Even number of 0
2. Accept the string 1000 only
3. Started with 0.
4. Ended with 1.
5. Started with 0 and ended with 1
6. Odd number of 1
7. Started with 00
8. Accept the string having substring 010
9. Accept the string having substring 1101
10. Accept the string having substring 1100
11. Even number of 0 and even number of 1
12. Even number of 0 and odd number of 1
13. Odd number of 0 and even number of 1
14. odd number of 0 and odd number of 1
15. String length module 3 =0(length is multiple of 3)
16. Decimal representation of string modulo 3 =0

Write the RE over Σ= (a, b) for the following

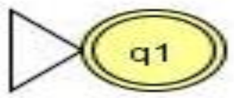
1. Accept the string a^*
2. Accept the string b^+
3. Even number of a
4. Even number of a followed by even number of b
5. Even number of a followed by odd number of b
6. Odd number of a followed by even number of b
7. Odd number of a followed by odd number of b
8. Accept the string having at least 3 a
9. Accept the string having at least 3 b
10. Accept the string having at least 3 a and at least 2 b
11. Exactly three a and exactly 2 b

Algebra for RE: If P, Q, R are RE then following holds

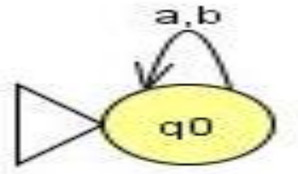
1. $\varphi+R=R$
2. $\varphi R=R\varphi=\varphi$
3. $\varepsilon R=R\varepsilon=R$
4. $\varepsilon^*=\varepsilon$
5. $\Phi^*=\varepsilon$
6. $R+R=R$
7. $R^*R^*=R^*$
8. $RR^*=R^*R=R^+$
9. $(R^*)^*=R^*$
10. $\varepsilon+RR^*=R^*=\varepsilon R^*R$
11. $(PQ)^*P=P(QP)^*$
12. $(P+Q)^*=(P^*Q^*)^*=(P^*+Q^*)^*$
13. $(P+Q)R=PR+QR$
14. $R(P+Q)=RP+RQ$

Building Finite Automata from Regular expression

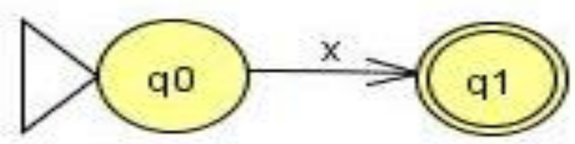
1. There is an FA that accepts the language defined by regular expression Ɛ; i.e., the language {Ɛ}



2. There is an FA defined by the regular expression ∅ ; i.e., the language with no words, which is ∅. [Let ∑ ={a,b}].



3. There is an FA that accepts the language L defined by the regular expression x ; i.e., L = { x } , where x ∈ Σ, so language L consists of only a single word and that word is the single letter x



- **Based on Rule 2, we get following definition of FA:**

If there is an FA called FA_1 that accepts the language defined by the regular expression r_1 and there is an FA called FA_2 that accepts the language defined by the regular expression r_2 , then there is an FA called FA_3 that accepts the language defined by the regular expression r_1+r_2.

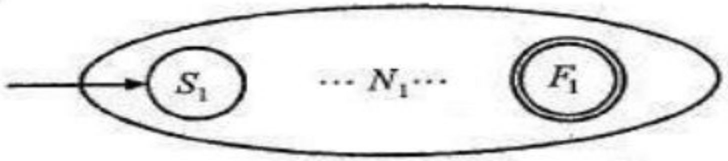


FIGURE 2(a) NFA for regular set R_1

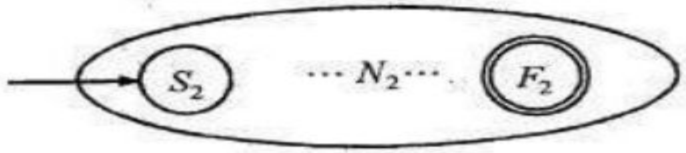
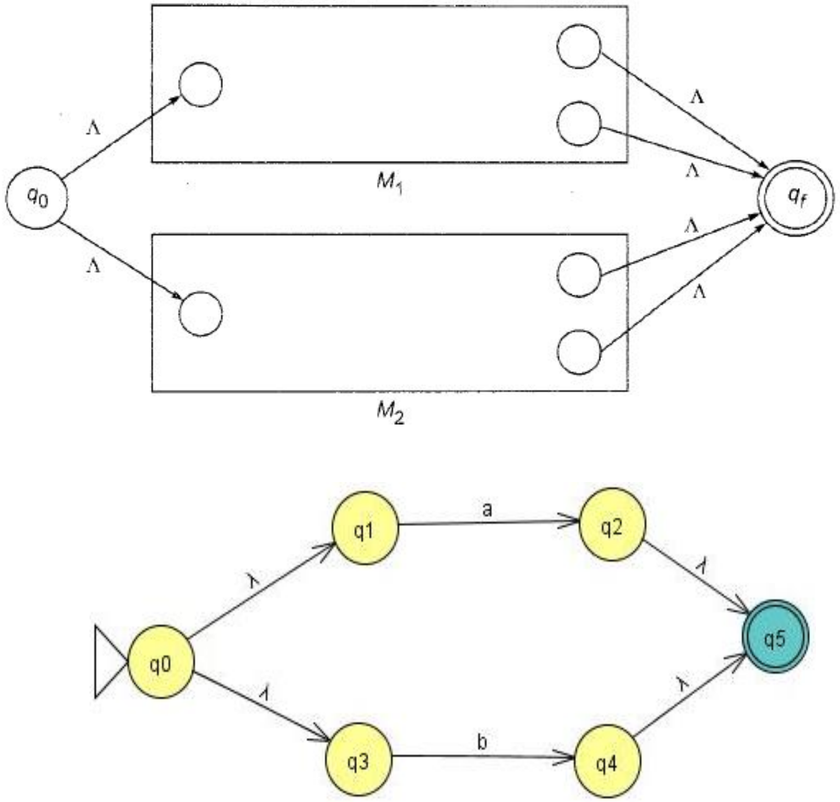


FIGURE 2(b) NFA for regular set R_2



- **Based on Rule 3, we get following definition of FA:**

  If there is an FA called FA_1 that accepts the language defined by the regular expression r1and there is an FA called FA_2 that accepts the language defined by the regular expression r_2, then there is an FA called FA_3 that accepts the language defined by the regular expression $r_1 r_2$, which is the concatenation.

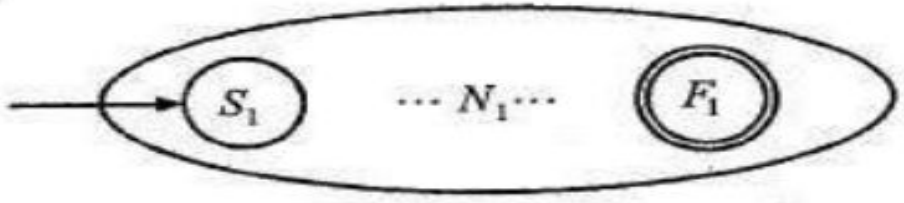


FIGURE 2(a) NFA for regular set R_1

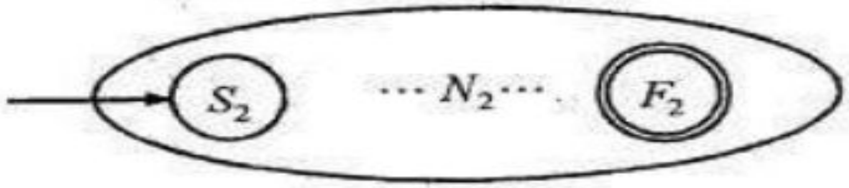
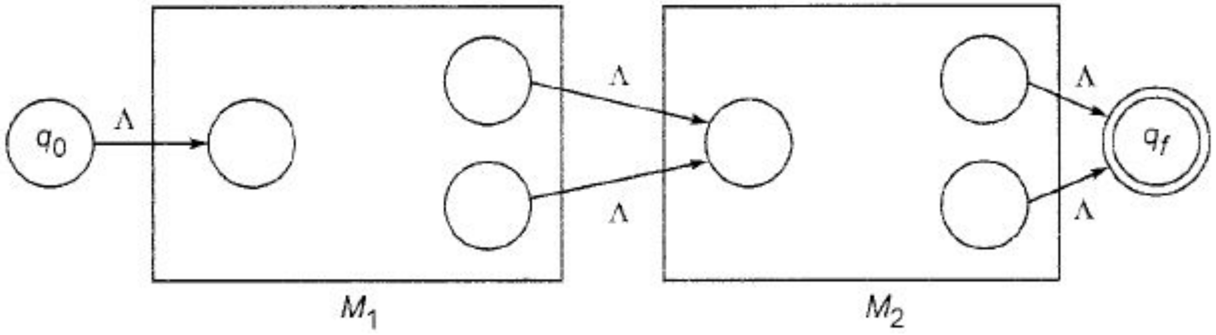
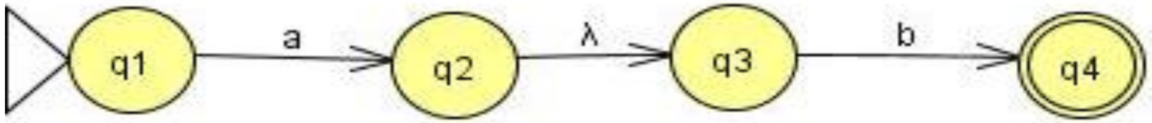
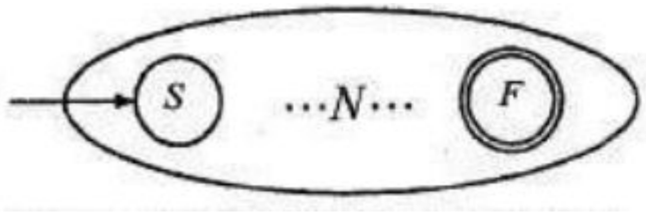


FIGURE 2(b) NFA for regular set R_2

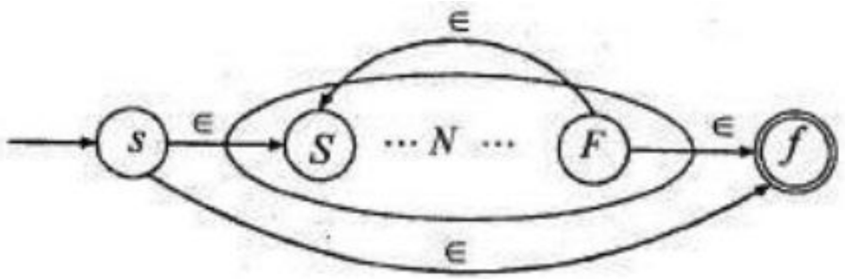




NDFA accepting *L*/Kleene star.



NDFA accepting L*/Kleene star



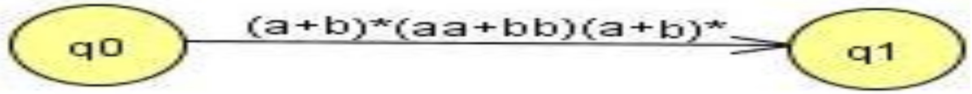
Design FA from the given RE

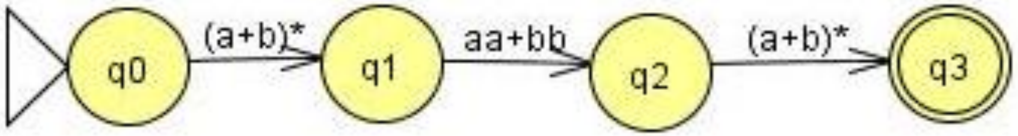
1. a+b
2. (a+b)c
3. (a+b)*
4. (a+b)*c

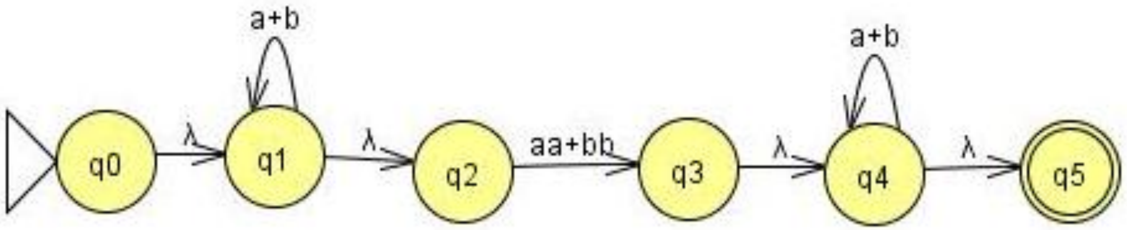
1. $(0+1)^*(00+11)(0+1)^*$

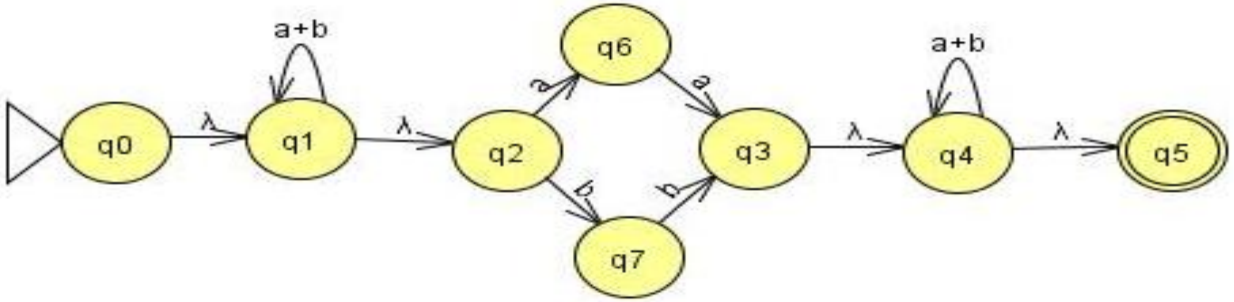
Design FA from the given RE

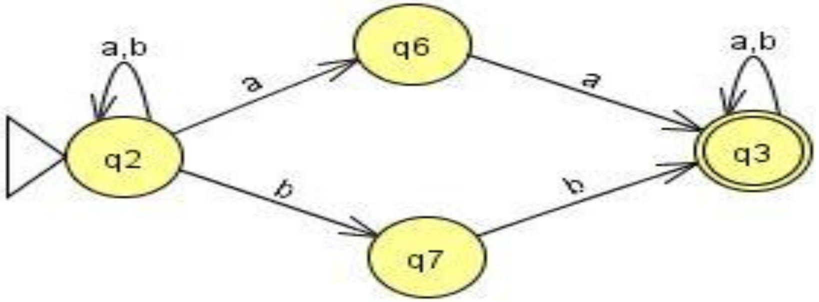
1. (a+b)*(aa+bb)(a+b)*











Arden's Theorem:

Let P and Q are RE over Σ and P does not contain Ɛ, then following equivalents R=Q+RP---(A) has an unique solution R=QP*

Proof: To prove this theorem we put the value of R in equation A. Hence , we get Q+ RP = Q+(Q+RP)P

= Q+QP+RPP

= Q+QP+RP2

= Q+QP+QP2+….QPi+RPi+1

= Q(Ɛ +P +P2+…+Pi)+RP(i+1)

Q+RP = Q(Ɛ +P +P2+…+P^i)+RP^{i+1}

for all i>=0 …..(B)

Let us assume *w* be the string of length in the set R. then *w* belongs to the equation B. But as P does not contain Ɛ , hence RP^{i+1} does not contain any string of length less than i+1 and so *w* does not belongs to the set RP^{i+1}. This means *w* belongs to the QP*.

Show that

(1+00*1)+(1+00*1) (0+10*1)* (0+10*1)= 0*1(0+10*1)*

(1+00*1) [(Ɛ +(0+10*1)*(0+10*1)] By R-14

(1+00*1)(0+10*1)* By R-10

(Ɛ+00*)1(0+10*1)* By R13

0*1(0+10*1)* By Arden's Theorem

10	Ɛ+RR*=R*= ƐR*R
11	(PQ)*P=P(QP)*
12	(P+Q)*=(P*Q*)*=(P*+Q*)*
13	(P+Q)R=PR+QR
14	R(P+Q)=RP+RQ

Show that

Ɛ+1*(011)*(1*(011)*)*=(1+011)*

=(1*(01)*)* as Ɛ+PP*=P*

=(1+01)* as (p*+Q*)*=(P+Q)*

10 Ɛ+RR*=R*= ƐR*R
11 (PQ)*P=P(QP)*
12 (P+Q)*=(P*Q*)*=(P*+Q*)*
13 (P+Q)R=PR+QR
14 R(P+Q)=RP+RQ

Finite Automata to Regular Expression

- We can use Arden's theorem to find the regular expression from a given finite automata.
- Assumptions;

  Finite automata has only one initial state.

  No λ move

Consider the incoming arrows for a node

- From this graph we can write equations as

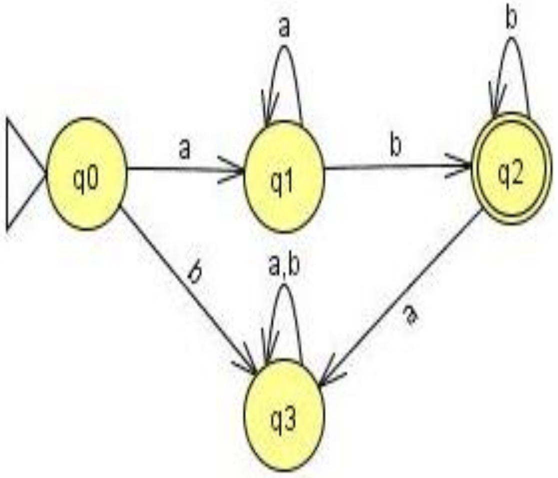
  q0= λ (No incoming arrow on q0)

  q1=q0.a + q1.a (from q0,a and q1,b)

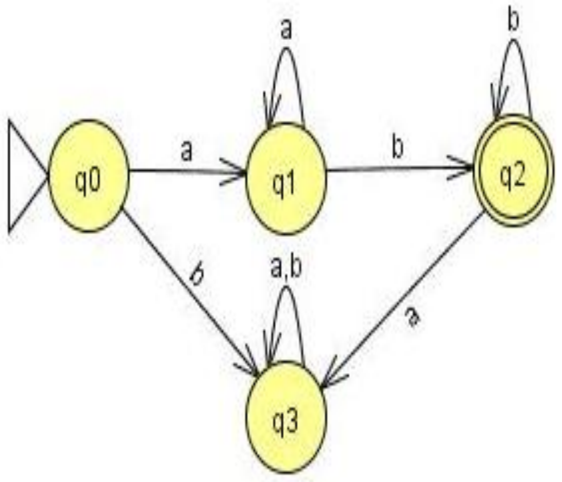
  q2=q1.b+q2.b (from q1,b and q2,b)

  q3 = q0.b + q3.a +q3.b +q2.a

  (from q0,b; from q3,b; from q3,a; from q2,a)



q0= Ɛ (1)
q1=q0.a + q1.a (2)
q2=q1.b+q2.b (3)
q3 = q0.b + q3.a +q3.b +q2.a (4)



q1= Ɛ a+q1a = a+q1a=Q+RP=>R=QP*
q1= aa*
q2=aa*b+q2b = Q+RP => R=QP*
q2=aa*bb*

Now q2 is a final state therefore our conversion ends here.

Example

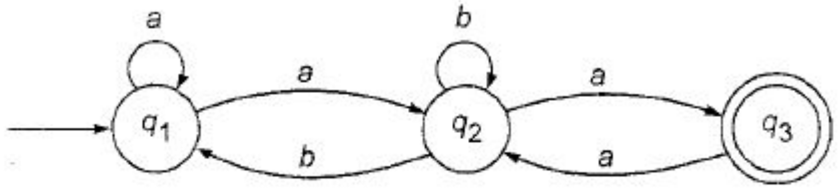


Fig. 5.13 Transition system of Example 5.8.

ql = q1a + **q2b** + Ɛ------- (1)
q2 = q1a + q2b + q3a . (2)
q3 = q2a (3)

$$q_2 = q_1a + q_2b + q_2aa$$
$$= q_1a + q_2(b + aa)$$
$$= q_1a(b + aa)^*$$

Substituting q_2 in q_1, we get

$$q_1 = q_1a + q_1a(b + aa)^*b + \Lambda$$
$$= q_1(a + a(b + aa)^*b) + \Lambda$$

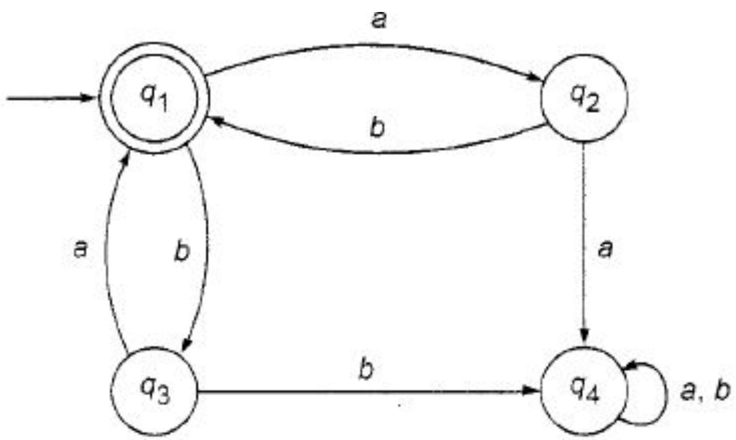
Hence,

$$q_1 = \Lambda(a + a(b + aa)^*b)^*$$
$$q_2 = (a + a(b + aa)^*b)^* a(b + aa)^*$$
$$q_3 = (a + a(b + aa)^*b)^* a(b + aa)^*a$$

Since q_3 is a final state, the set of strings recognized by the graph is given by

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

Example



$q1 = q2b + q3a + \mathcal{E}$-------- (1)

$q2 = q1a$ (2)

$q3 = q1b$ (3)

$q4=q2a+q3b+q4a+q4b$ (4)

$$q_1 = q_2b + q_3a + \Lambda$$

$$q_2 = q_1a$$

$$q_3 = q_1b$$

$$q_4 = q_2a + q_3b + q_4a + q_4b$$

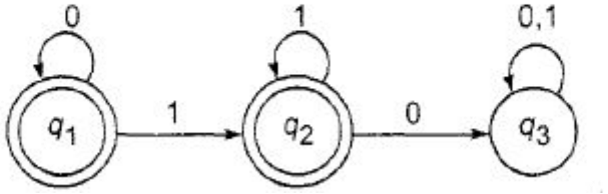
As q_1 is the only final state and the q_1-equation involves only q_2 and q_3, we use only q_2- and q_3-equations (the q_4-equation is redundant for our purposes). Substituting for q_2 and q_3, we get

$$q_1 = q_1ab + q_1ba + \Lambda = q_1(ab + ba) + \Lambda$$

By applying Theorem 5.1, we get

$$q_1 = \Lambda(ab + ba)^* = (ab + ba)^*$$

Example



q_1 = q_1**0+** Ɛ------- (1)
$q_2 = q_1 1+q_2 1$ (2)
$q_3=q_2 0+q_3 0+q_3 1$ (3)

By applying Theorem 5.1 to the $\mathbf{q}_1$-equation, we get

$$\mathbf{q}_1 = \Lambda\mathbf{0}^* = \mathbf{0}^*$$

So,

$$\mathbf{q}_2 = \mathbf{q}_1\mathbf{1} + \mathbf{q}_2\mathbf{1} = \mathbf{0}^*\mathbf{1} + \mathbf{q}_2\mathbf{1}$$

Therefore,

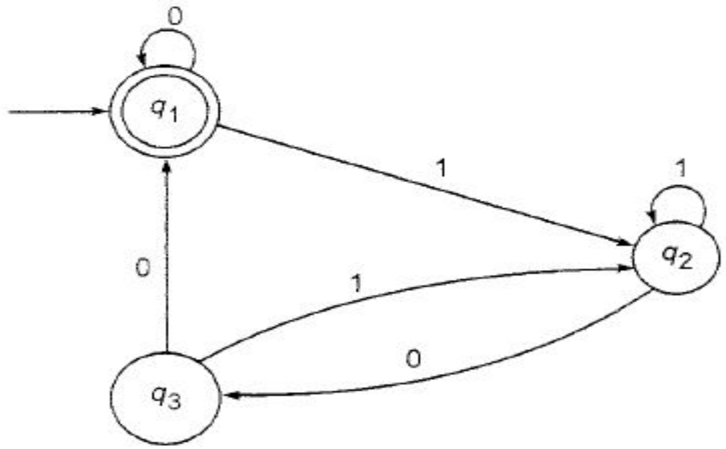
$$\mathbf{q}_2 = (\mathbf{0}^*\mathbf{1})\mathbf{1}^*$$

As the final states are $\mathbf{q}_1$ and $\mathbf{q}_2$, we need not solve for $\mathbf{q}_3$:

$$\mathbf{q}_1 + \mathbf{q}_2 = \mathbf{0}^* + \mathbf{0}^*(\mathbf{11}^*) = \mathbf{0}^*(\Lambda + \mathbf{11}^*) = \mathbf{0}^*(\mathbf{1}^*) \quad \text{by } I_9$$

The strings represented by the transition graph are $\mathbf{0}^*\mathbf{1}^*$. We can interpret the

Example



$q_1 = \mathbf{q_1 0} + q_3 0 + \mathcal{E}$------- (1)

$q_2 = q_1 1 + q_2 1 + q_3 1$ (2)

$q_3 = q_2 0$ (3)

So,

$$q_2 = q_1 1 + q_2 1 + (q_2 0)1 = q_1 1 + q_2(1 + 01)$$

By applying Theorem 5.1, we get

$$q_2 = q_1 1(1 + 01)^*$$

Also,

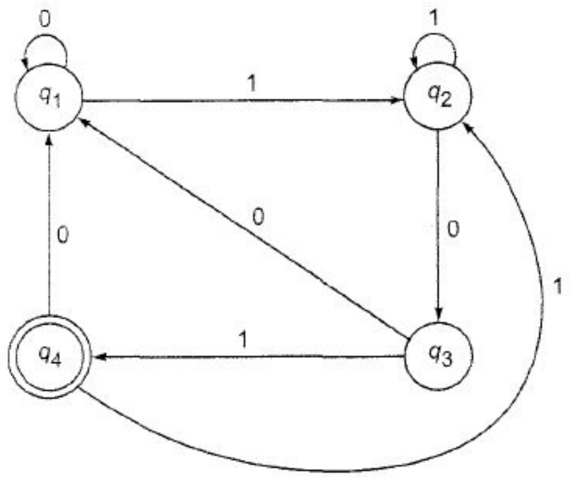
$$q_1 = q_1 0 + q_3 0 + \Lambda = q_1 0 + q_2 00 + \Lambda$$
$$= q_1 0 + (q_1 1(1 + 01)^*)00 + \Lambda$$
$$= q_1(0 + 1(1 + 01)^* 00) + \Lambda$$

Once again applying Theorem 5.1, we get

$$q_1 = \Lambda(0 + 1(1 + 01)^* 00)^* = (0 + 1(1 + 01)^* 00)^*$$

As q_1 is the only final state, the regular expression corresponding to the given diagram is $(0 + 1(1 + 01)^* 00)^*$.

Example



Solution

There is only one initial state, and there are no Λ-moves. So, we form the equations corresponding to q_1, q_2, q_3, q_4:

$$q_1 = q_10 + q_30 + q_40 + \Lambda$$

$$q_2 = q_11 + q_21 + q_41$$

$$q_3 = q_20$$

$$q_4 = q_31$$

Now,

$$q_4 = q_31 = (q_20)1 = q_201$$

Thus, we are able to write q_3, q_4 in terms of q_2. Using the q_2-equation, we get

$$q_2 = q_11 + q_21 + q_2011 = q_11 + q_2(1 + 011)$$

By applying Theorem 5.1, we obtain

$$q_2 = (q_11)(1 + 011)^* = q_1(1(1 + 011)^*)$$

From the q_1-equation, we have

$$q_1 = q_10 + q_200 + q_2010 + \Lambda$$
$$= q_10 + q_2(00 + 010) + \Lambda$$
$$= q_10 + q_11(1 + 011)^* (00 + 010) + \Lambda$$

Again, by applying Theorem 5.1, we obtain

$$q_1 = \Lambda(0 + 1(1 + 011)^* (00 + 010))^*$$

$$q_4 = q_201 = q_11(1 + 011)^* 01$$
$$= (0 + 1(1 + 011)^*(00 + 010))^*(1(1 + 011)^* 01)$$

Design a DFA over Σ =(a, b) , which recognize the language having even number of a and even number of b. And convert to corresponding RE

Pumping Lemma for Regular languages:

Theorem:(Pumping Lemma). If L is an infinite regular language, then there exists a number p (Pumping length) such that for all w € L with |w|>=p, w can be written as xyz satisfying the following:

1. For all i >=0 xy^iz € L
2. |y|>0
3. |xy|<=p

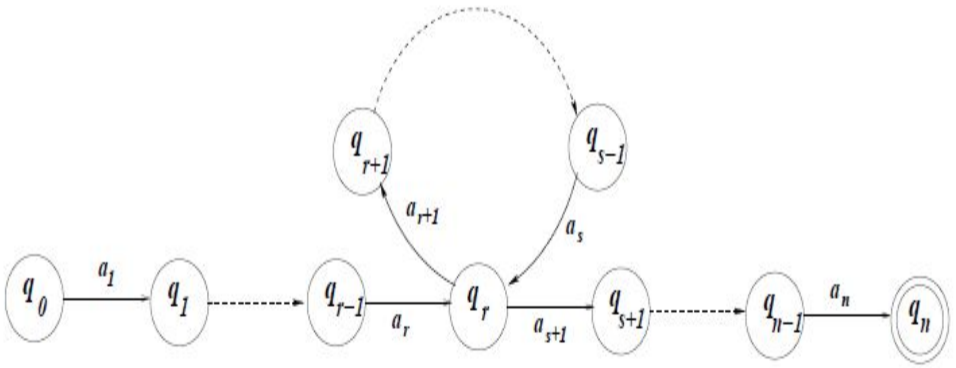
Proof:

Let M=(Q, ∑, δ, q0,F) be a DFA. Let w=$a_1,a_2,a_3,\ldots a_n$,n>p be a string.

Let q_1,q_2,---q_{n+1} be the sequence of state that M enters while processing.

So $q_{i+1}= \delta(q_i,a_i)$, 1<=i<=n be the sequence of length n+1 which is at least p+1.

Among the sequence of state two must be in same state.(By pigeonhole principle)



We call the first of these q_j and second is q_l because q_l occurs among the first p+1place in the sequence.
We have l<=p+1.
Now let w= $a_1,a_2,a_3,\ldots a_{j-1}$
X=a1,a2,…a_{j-1}
y = a_j,----a_{l-1}
Z= a_l,----a_n
As x takes M from q_1 to q_j, Y takes M for q_j to q_j, Z takes q_{j+1} to q_{n+1} which is acceptance state
Hence M must accept xy^iz for all I
And l<=p+1 so |xy|<=p, as we reach in final state xy^iz € L

Pumping Lemma Example:

Prove that $a^n b^n$ is not regular

Proof:

Let us assume $a^n b^n$ is Regular language.

w= $a^n b^n$, $|w|=2n>n$, where n is the pumping length

By pumping lemma w=xyz with $|xy|<=n$, $|y| \geq 1$

We need to find $x y^i z$

Case 1: y has 'a' ie y= a^k, k>0

Case 2 y has 'b' ie y= b^k, k>0

Case 3 y has both a and b ie y=$a^k b^l$, k,l >=1

Case 1: y has 'a' ie y= a^k, k>0

$xy^2z= a^{n-k}a^ka^kb^n =a^{n+k}b^n \notin L$

Hence what we assumed $a^n b^n$ is Regular language is wrong. Hence $a^n b^n$ is not Regular language.

Similarly we can do for other two case.

Prove that a^p is not regular, where p is a prime number

Let us assume $\mathbf{a^p}$ is Regular language.

w= $\mathbf{a^p}$,|w|=p>=m , where m is the pumping length

w= a^p= xyz, where |xy|<=p, |y|>0

Now evaluate $xy^{p+1}z$

Length($xy^{p+1}z$)= Length($xy^p yz$)= Length(xyz)+Length(y^p)=

p+p(length(y)

=p+p.m

=p(1+m) which is not prime number.

Hence what we assumed $\mathbf{a^p}$ is Regular language is wrong.

So $\mathbf{a^p}$ is not Regular language

Prove that a^{i^2} i>=1is not regular

Proof:

Let a^{i^2} is regular

W= a^{n^2} ,|w|=n^2>n by pumping lemma

W=xyz with |xy|<=n, |y|>0

Now |xy^2z|=

Thank You