UNIT- II

REGULAR EXPRESSIONS

3

REGULAR LANGUAGES AND FINITE AUTOMATA

After going through this chapter, you should be able to understand :

- Regular sets and Regular Expressions
- Identity Rules
- Constructing FA for a given REs
- Conversion of FA to REs
- Pumping Lemma of Regular sets
- Closure properties of Regular sets

3.1 REGULAR SETS

A special class of sets of words over S, called regular sets, is defined recursively as follows. (Kleene proves that any set recognized by an FSM is regular. Conversely, every regular set can be recognized by some FSM.)

1. Every finite set of words over S (including ϵ, the empty set) is a regular set.
2. If A and B are regular sets over S, then $A \cup B$ and AB are also regular.
3. If S is a regular set over S, then so is its closure S*.
4. No set is regular unless it is obtained by a finite number of applications of definitions (1) to (3).

i.e., the class of regular sets over S is the smallest class containing all finite sets of words over S and closed under union, concatenation and star operation.

Examples:

i) Let $\Sigma = \{a,b\}$ then the set of strings that contain both odd number of a's and b's is a regular set.

ii) Let $\Sigma = \{0\}$ then the set of strings $\{0,00,000,.....\}$ is a regular set.

iii) Let $\Sigma = \{0,1\}$ then the set of strings $\{01,10\}$ is a regular set.

3.2 REGULAR EXPRESSIONS

The languages accepted by FA are regular languages and these languages are easily described by simple expressions called regular expressions. We have some algebraic notations to represent the regular expressions.

Regular expressions are means to represent certain sets of strings in some algebraic manner and regular expressions describe the language accepted by FA.

If Σ is an alphabet then regular expression(s) over this can be described by following rules.

1. Any symbol from $\Sigma, \in$ *and* ϕ are regular expressions.
2. If r_1 and r_2 are two regular expressions then *union* of these represented as $r_1 \cup r_2$ or $r_1 + r_2$ is also a regular expression
3. If r_1 and r_2 are two regular expressions then *concatenation* of these represented as $r_1 r_2$ is also a regular expression.
4. The Kleene closure of a regular expression r is denoted by r^* is also a regular expression.
5. If r is a regular expression then (r) is also a regular expression.
6. The regular expressions obtained by applying rules 1 to 5 once or more than once are also regular expressions.

Examples :

(1) If $\Sigma = \{a, b\}$, then

(a) a is a regular expression (Using rule 1)
(b) b is a regular expression (Using rule 1)
(c) $a + b$ is a regular expression (Using rule 2)
(d) b^* is a regular expression (Using rule 4)
(e) ab is a regular expression (Using rule 3)
(f) $ab + b^*$ is a regular expression (Using rule 6)

(2) Find regular expression for the following

(a) A language consists of all the words over $\{a, b\}$ ending in b.

(b) A language consists of all the words over $\{a, b\}$ ending in bb.

(c) A language consists of all the words over $\{a, b\}$ starting with a and ending in b.

(d) A language consists of all the words over $\{a, b\}$ having bb as a substring.

(e) A language consists of all the words over $\{a, b\}$ ending in aab.

Solution : Let $\Sigma = \{a, b\}$, and

All the words over $\Sigma = \{\in, a, b, aa, bb, ab, ba, aaa, \ldots\} = \Sigma^*$ *or* $(a + b)^*$ *or* $(a \cup b)^*$

(a) Regular expression for the given language is $(a + b) * b$

(b) Regular expression for the given language is $(a + b) * bb$

(c) Regular expression for the given language is $a (a + b) * b$

(d) Regular expression for the given language is $(a + b) * aa$ or $aa (a + b) *$ or $(a + b) * bb (a + b) *$

(e) Regular expression for the given language is $(a + b) * aab$

The table below shows some examples of regular expressions and the language corresponding to these regular expressions.

Regular expression	Meaning
(a + b)*	Set of strings of a's and b's of any length including the NULL string.
(a + b)*abb	Set of strings of a's and b's ending with the string abb.
ab (a + b)*	Set of strings of a's and b's starting with the string ab.
(a + b)*aa (a + b)*	Set of strings of a's and b's having a sub string aa.
a*b*c*	Set of strings consisting of any number of a's (may be empty string also) followed by any number of b's (may include empty string) followed by any number of c's (may include empty string).
$a^+b^+c^+$	Set of strings consisting of at least one 'a' followed by string consisting of at least one 'b' followed by string consisting of at least one 'c'.
aa*bb*cc*	Set of strings consisting of at least one 'a' followed by string consisting of at least one 'b' followed by string consisting of at least one 'c'.
(a + b)* (a + bb)	Set of strings of a's and b's ending with either a or bb.
(aa) * (bb)*b	Set of strings consisting of even number of a's followed by odd number of b's.
(0 + 1) * 000	Set of strings of 0's and 1's ending with three consecutive zeros(or ending with 000)
(11)*	Set consisting of even number of 1's

TABLE: Meaning of regular expressions

Example 1 : Obtain a regular expression to accept a language consisting of strings of a's and b's of even length.

Solution :

String of a's and b's of even length can be obtained by the combination of the strings aa, ab, ba, and bb. The language may even consist of an empty string denoted by $\in$. So, the regular expression can be of the form

$$(aa + ab + ba + bb)^*$$

The * closure includes the empty string.

Note : This regular expression can also be represented using set notation as

$$L(r)=\{(aa + ab + ba + bb)^n | n \geq 0\}$$

Example 2 : Obtain a regular expression to accept a language consisting of strings of a's and b's of odd length.

Solution :

String of a's and b's of odd length can be obtained by the combination of the strings aa, ab, ba and bb followed by either a or b. So, the regular expression can be of the form

$$(aa+ab+ba+bb)^* (a+b)$$

String of a's and b's of odd length can also be obtained by the combination of the strings aa, ab, ba and bb preceded by either a or b. So, the regular expression can also be represented as

$$(a+b)(aa+ab+ba+bb)^*$$

Note : Even though these two expressions are seems to be different, the language corresponding to those two expressions is same. So, a variety of regular expressions can be obtained for a language and all are equivalent.

Example 3 : Obtain a regular expression such that $L(r) = \{W \mid W \in \{0,1\}^*$ with at least three consecutive 0's $\}$.

Solution :

An arbitrary string consisting of 0's and 1's can be represented by the regular expression.

$$(0 + 1)^*$$

This arbitrary string can precede three consecutive zeros and can follow three consecutive zeros. So, the regular expression can be written as

$$(0+1)^*000(0+1)^*$$

Note : Using the set notation the regular expression can be written as

$$L(r) = \{(0 + 1)^m 000 (0 + 1)^n \mid m \geq 0 \text{ and } n \geq 0\}$$

Example 4 : Obtain a regular expression to accept strings of a's and b's ending with 'b' and has no substring aa.

Solution :

Note : The statement "strings of a's and b's ending with 'b' and has no substring aa" can be restated as "string made up of either b or ab". Note that if we state something like this, the substring aa will never occur in the string and the string ends with 'b'. So, the regular expression can be of the form

$$(b + ab)^*$$

But, because of * closure, even null string is also included. But, the string should end with 'b'. So, instead of * closure, we can use positive closure '+'. So, the regular expression to accept strings of a's and b's ending with 'b' and has no substring aa can be written as

$$(b+ab)^*$$

The above regular expression can also be written as

$$(b+ab)\,(b+ab)^*$$

Note : Using the set notation this regular expression can be written as

$$L(r)=\{(b + ab)^n | n\geq 1\}$$

Example 5 : Obtain a regular expression to accept strings of 0's and 1's having no two consecutive zeros.

Solution :

The first observation from the statement is that whenever a 0 occurs it should be followed by 1. But, there is no restriction on the number of 1's. So , it is a string consisting of any combination of 1's and 01's. So, the partial regular expression for this can be of the form

$$(1+01)^*$$

No doubt that the above expression is correct. But, suppose the string ends with a 0. What to do? For this, the string obtained from above regular expression may end with 0 or may end with $\in$ (i. e., may not end with 0). So, the above regular expression can be written as

$$(1+01)^*(0+\in)$$

Example 6 : Obtain a regular expression to accept strings of a's and b's of length ≤ 10.

Solution :

The regular expression for this can be written as

$$\in + a + b + aa + ab + ba + bb + \ldots\ldots + bbbbbbbbba + bbbbbbbbbb$$

But, using ..... in a regular expression is not recommended and so we can write the above expression as

$$(\in + a + b)^{10}$$

Example 7 : Obtain a regular expression to accept strings of a's and b's starting with 'a' and ending with 'b'.

Solution :

Strings of a's and b's of arbitrary length can be written as $(a+b)^*$

But, this should start with 'a' and end with 'b'. So, the regular expression can be written as

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

Hierarchy of Evaluation of Regular Expressions

We follow the following order when we evaluate a regular expression.

1. Parenthesis
2. Kleene closure
3. Concatenation
4. Union

Example 1: Consider the regular expression $(a + b)^* aab$ and describe the all words represented by this.

Solution :

$(a + b)^* aab$ = {All words over $\{a,b\}$} aab (Evaluating $(a + b)^*$ first)

= $\{\in, a, b, aa, bb, ab, ba, aaa, \ldots\}\, aab$

= {All words over $\{a, b\}$ ending in aab}

Example 2: Consider the regular expression $(a^* + b^*)^*$ and explain it.

Solution : We evaluate a^* and b^* first then $(a^* + b^*)^*$.

$(a^* + b^*)^*$ = (All the words over $\{a\}$ + all the words over $\{b\}$)*

= ($\{\in, a, aa, \ldots\}$ or $\{\in, b, bb, \ldots\}$)*

$= (\{\in, a, b, aa, bb, \ldots\})^*$

$= \{\in, a, b, aa, bb, ab, ba, aaa, bbb, abb, baa, aabb, \ldots\}$

= {All the words over $\{a, b\}$ }

$\equiv (a + b)^*$

So, $(a^* + b^*)^* \equiv (a + b)^*$

3.3 IDENTITIES FOR REs

The two regular expressions P and Q are equivalent (denoted as P = Q) if and only if P represents the same set of strings as Q does. For showing this equivalence of regular expressions we need to show some identities of regular expressions.

Let P, Q and R are regular expressions then the identity rules are as given below

1.	$\in R = R \in = R$	
2.	$\in^* = \in$	$\in$ is null string
3.	$(\phi)^* = \in$	ϕ is empty string.
4.	$\phi R = R\phi = \phi$	
5.	$\phi + = R = R$	
6.	$R + R = R$	
7.	$RR^* = R^* R = R^+$	
8.	$(R^*)^* = R^*$	
9.	$\in + RR^* = R^*$	
10.	$(P + Q)R = PR + QR$	
11.	$(P + Q)^* = (P^* Q^*) = (P^* + Q^*)^*$	
12.	$R^*(\in + R) = (\in + R)R^* = R^*$	
13.	$(R + \in)^* = R^*$	
14.	$\in + R^* = R^*$	
15.	$(PQ)^* P = P(QP)^*$	
16.	$R^* R + R = R^* R$	

3.3.1 Equivalence of two REs

Let us see one important theorem named Arden's Theorem which helps in checking the equivalence of two regular expressions.

Arden's Theorem : Let P and Q be the two regular expressions over the input set Σ. The regular expression R is given as

$$R = Q + RP$$

Which has a unique solution as $R = QP^*$

Proof : Let, P and Q are two regular expressions over the input string Σ.
If P does not contain $\in$ then there exists R such that

$$R = Q + RP \qquad \ldots (1)$$

We will replace R by QP* in equation 1.
Consider R. H. S. of equation 1.

$$= Q + QP^*P$$

$$= Q(\in + P^*P)$$

$$= QP^* \qquad \because \in + R^*R = R^*$$

Thus $R = QP^*$

is proved. To prove that $R = QP^*$ is a unique solution, we will now replace L.H.S. of equation 1 by Q + RP. Then it becomes

$$Q + RP$$

But again R can be replaced by Q + RP.

$$\therefore \quad Q + RP = Q + (Q + RP)P$$

$$= Q + QP + RP^2$$

Again replace R by Q + RP.

$$= Q + QP + (Q + RP)P^2$$

$$= Q + QP + QP^2 + RP^3$$

Thus if we go on replacing R by Q + RP then we get,

$$Q + RP = Q + QP + QP^2 + \ldots\ldots + QP^i + RP^{i+1}$$

$$= Q(\in + P + P^2 + \ldots\ldots P^i) + RP^{i+1}$$

From equation 1,

$$R = Q(\in + P + P^2 + \ldots\ldots + P^i) + RP^{i+1} \qquad \ldots (2)$$

Where $i \geq 0$

Consider equation 2,

$$R = Q\underbrace{(\in + P + P^2 + \ldots\ldots + P^i)}_{P^*} + RP^{i+1}$$

$$\therefore \quad R = QP^* + RP^{i+1}$$

Let w be a string of length i.

In RP^{i+1} has no string of less than i + 1 length. Hence w is not in set RP^{i+1}. Hence R and QP^* represent the same set. Hence it is proved that

R = Q + RP has a unique solution.

$R = QP^*$.

Example 1 : Prove $(1+00*1)+(1+00*1)(0+10*1)*(0+10*1) = 0*1(0+10*1)*$

Solution : Let us solve L.H.S. first,

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

We will take $(1+00*1)$ as a common factor

$$1+00*1\ \underbrace{(\in +(0+10*1)*(0+10*1))}$$

$(\in +R*R)$ where $R = (0+10*1)$

As we know, $(\in +R*R) = (\in +RR*) = R*$

$\therefore (1+00*1)((0+10*1)*)$ out of this consider

$$\underbrace{(1+00*1)}\ (0+10*1)*$$

Taking 1 as a common factor

$(\in +00*)1(0+10*1)*$

Applying $\in +00* = 0*$

$0*1(0+10*1)*$

= R. H. S.

Hence the two regular expressions are equivalent.

Example 2 : Show that $(0*1*)* = (0+1)*$

Solution : Consider L. H. S.

$= (0*1*)*$

$= \{\in, 0, 00, 1, 11, 111, 01, 10, ...\}$

= { any combination of 0's, any combination of 1's, any combination of 0 and 1, $\in$ }

Similarly,

R. H. S.

$= (0+1)*$

$= \{\in, 0, 00, 1, 11, 111, 01, 10, \ldots\ldots\}$

$= \{ \in$, any combination of 0's, any combination of 1's, any combination of 0 and 1 $\}$

Hence, L. H. S. = R. H. S. is proved.

3.4 RELATIONSHIP BETWEEN FA AND RE

There is a close relationship between a finite automata and the regular expression we can show this relation in below figure.

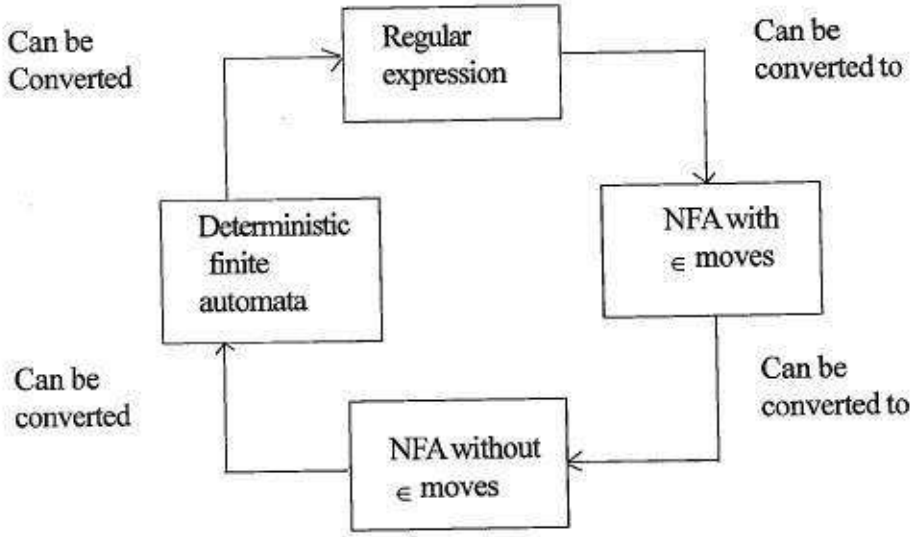


FIGURE : Relationship between FA and regular expression

The above figure shows that it is convenient to convert the regular expression to NFA with $\in$ moves. Let us see the theorem based on this conversion.

3.5 CONSTRUCTING FA FOR A GIVEN REs

Theorem : If r be a regular expression then there exists a NFA with $\in$ - moves, which accepts $L(r)$.

Proof : First we will discuss the construction of NFA M with $\in$ - moves for regular expression r and then we prove that $L(M) = L(r)$.

Let r be the regular expression over the alphabet Σ.

Construction of NFA with $\in$ - moves

Case 1 :

(i) $r = \phi$

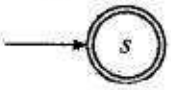
NFA $M = (\{s, f\}, \{\}, \delta, s, \{f\})$ as shown in Figure1 (a)

(No path from initial state s to reach the final state f.)

Figure 1 (a)

(ii) $r = \in$

NFA $M = (\{s\}, \{\}, \delta, s, \{s\})$ as shown in Figure 1 (b)

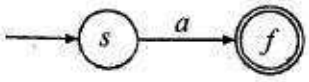


(The initial state s is the final state)

Figure 1 (b)

(iii) $r = a$, for all $a \in \Sigma$,

NFA $M = (\{s, f\}, \Sigma, \delta, s, \{f\})$



(One path is there from initial state s to reach the final state f with label a.)

Figure 1 (c)

Case 2 : $|r| \geq 1$

Let r_1 and r_2 be the two regular expressions over Σ_1, Σ_2 and N_1 and N_2 are two NFA for r_1 and r_2 respectively as shown in Figure 2 (a).

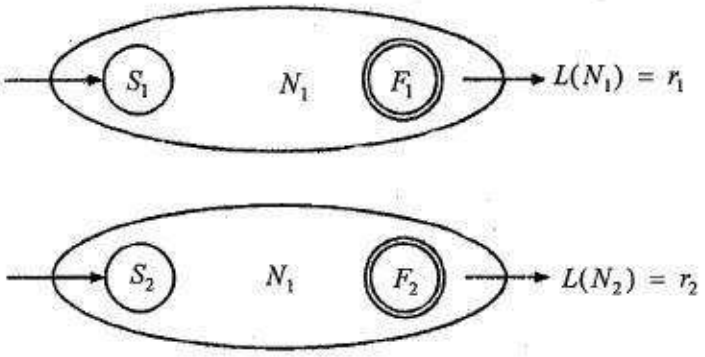


Figure 2 (a) NFA for regular expression r_1 and r_2

Rule 1 : For constructing NFA M for $\mathbf{r} = \mathbf{r}_1 + \mathbf{r}_2$ or $\mathbf{r}_1 \cup \mathbf{r}_2$

Let s and f are the starting state and final state respectively of M.

Transition diagram of M is shown in Figure 2 (b).

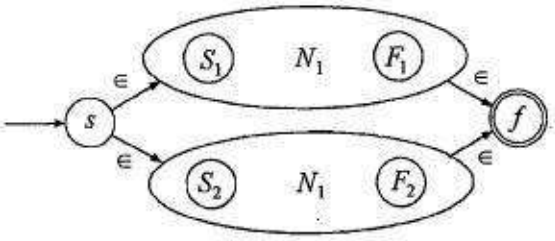


Figure 2 (b) NFA for regular expression $r_1 + r_2$

$$L(M) = \in L(N_1) \in \text{ or } \in L(N_2) \in$$
$$= L(N_1) \text{ or } L(N_2) = r_1 \text{ or } r_2$$

So, $r = r_1 + r_2$

$M = (Q, \Sigma_1 \cup \Sigma_2, \delta, s, \{f\})$, where Q contains all the states of N_1 and N_2.

Rule 2 : For regular expression $\mathbf{r} = \mathbf{r}_1\mathbf{r}_2$, NFA M is shown in Figure2 (c).

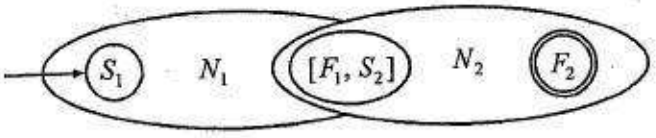


Figure 2 (c) NFA for regular expression r_1r_2

The final state (s) of N_1 is merged with initial state of N_2 into one state $[F_1S_2]$ as shown above in Figure2 (c).

$$L(M) = L(N_1) \text{ followed } L(N_2)$$
$$= L(N_1)\ L(N_2) = r_1r_2$$

So, $r = r_1r_2$

$M = (Q, \Sigma_1 \cup \Sigma_2, \delta, S_1, \{F_2\})$, where Q contains all the states of N_1 and N_2 such that final state(s) of N_1 is merged with initial state of N_2.

Rule 3 : For regular expression $\mathbf{r} = \mathbf{r_1^*}$, NFA M is shown in Figure2 (d)

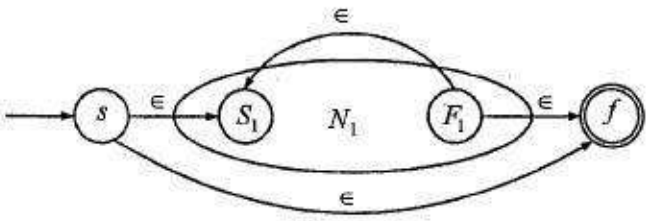


Figure 2 (d) NFA for regular expression for r_1^*

$$L(M) = \{\in, L(N_1), L(N_1)\ L(N_1), L(N_1)L(N_1)L(N_1), ...\}$$

$$= L(N_1)^*$$

$$= r_1^*$$

$M = (\{s, f\} \cup Q_1, \Sigma_1, \delta, s, \{f\})$, where Q_1 is the set of states of N_1.

Rule 4 : For construction of NFA M for $r = r_1^+$, M is shown in Figure 2 (e).

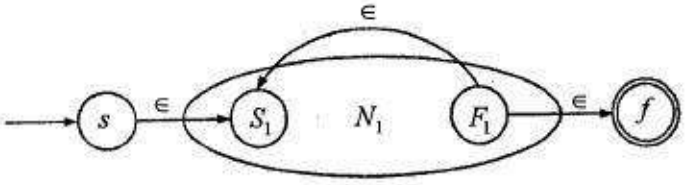


Figure 2(e) NFA for regular expression for r_1^+

$$L(M) = \{L(N_1), L(N_1)L(N_1), L(N_1)L(N_1)L(N_1), ...\}$$

$$= L(N_1)^+ = r_1^+$$

$M = (\{s, f\} \cup Q_1, \Sigma_1, \delta, s, \{f\})$, where Q_1 is the set of states of N_1.

Example 1 : Construct NFA for the regular expression a + ba *.

Solution : The regular expression

$r = a + ba*$ can be broken into r_1 and r_2 as

$r_1 = a$

$r_2 = ba*$

Let us draw the NFA for r_1, which is very simple.

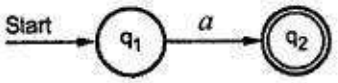


FIGURE 1: For r_1

Now, we will go for $r_2 = ba*$, this can be broken into r_3 and r_4 where $r_3 = b$ and $r_4 = a*$. Now the case for concatenation will be applied. The NFA will look like this r_3 will be shown in figure2.

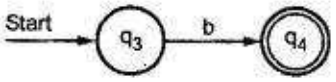


FIGURE 2: For r_3

and r_4 will be shown as

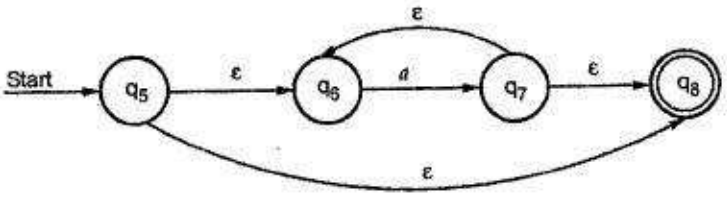


FIGURE 3 : For r_4

The r_2 will be $r_2 = r_3.r_4$

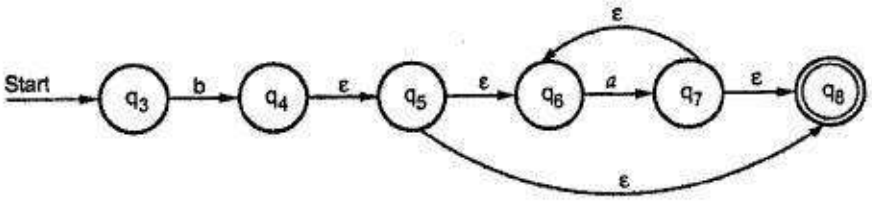


FIGURE 4 : For r_2

Now, we will draw NFA for $r = r_1 + r_2$ i. e. $a + ba*$

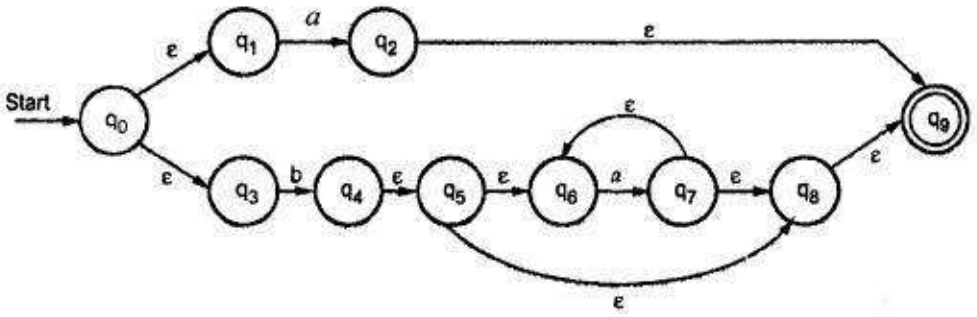


FIGURE 5 : NFA for $r = r_1 + r_2$ i. e. $a + ba*$

Example 2 : Construct NFA with $\in$ moves for the regular expression $(0+1)*$.

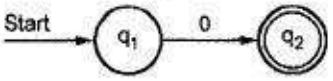
Solution : The NFA will be constructed step by step by breaking regular expression into small regular expressions.

$$r_3 = (r_1 + r_2)$$

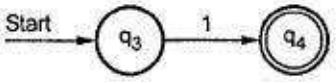
$$r = r_3^*$$

where $r_1 = 0$, $r_2 = 1$

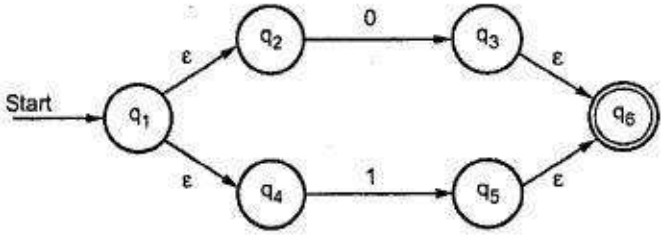
NFA for r_1 will be



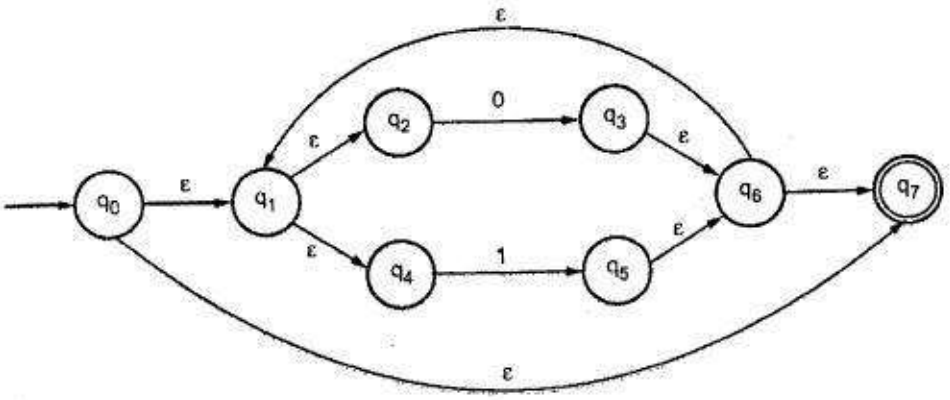
NFA for r_2 will be



NFA for r_3 will be



And finally



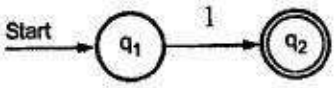
Example 3 : Construct NFA for the language having odd number of one's over the set $\Sigma = \{1\}$.

Solution : In this problem language L is given, we have to first convert it to regular expression. The r. e. for this L is written as r.e. = 1 (11)*.

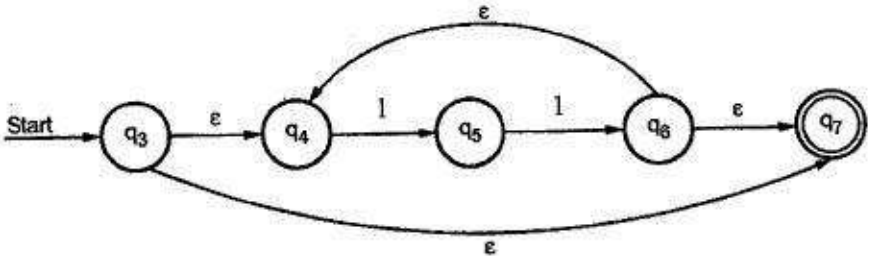
The r is now written as

$$r = r_1 \; r_2$$

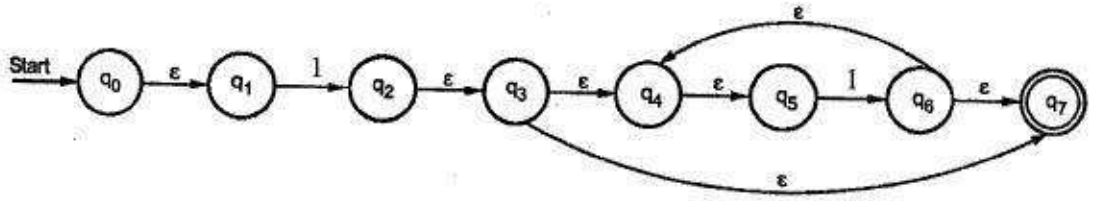
NFA for $r_1 = 1$ is



NFA for $r_2 = (11)^*$



The final NFA is



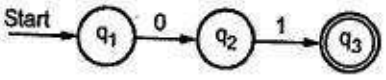
Example 4 : Construct NFA for the r. e. $(01+2^*)0$.

Solution : Let us design NFA for the regular expression by dividing the expression into smaller units

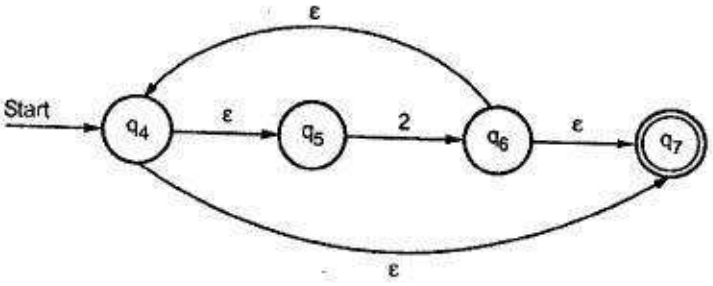
$$r = (r_1 + r_2)r_3$$

where $r_1 = 01$, $r_2 = 2^*$ and $r_3 = 0$

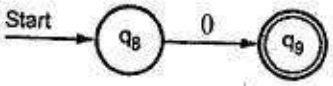
The NFA for r_1 will be



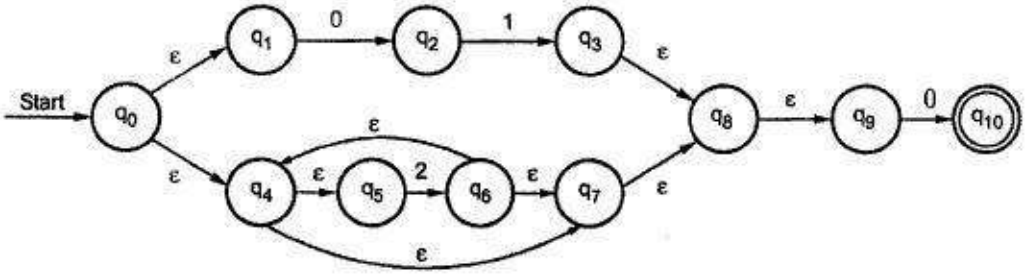
The NFA for r_2 will be



The NFA for r_3 will be



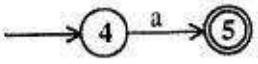
The final NFA will be



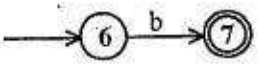
Example 5 : Obtain an NFA which accepts strings of a's and b's starting with the string ab.

Solution : The regular expression corresponding to this language is ab (a + b) *.

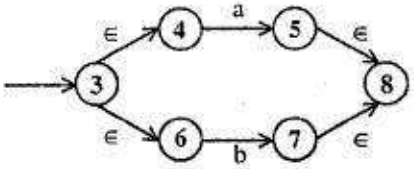
Step 1 : The machine to accept 'a' is shown below.



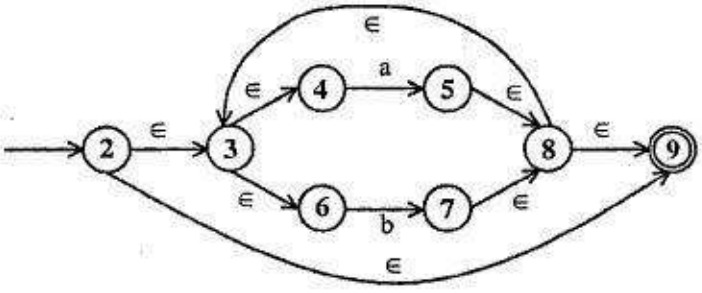
Step 2 : The machine to accept 'b' is shown below.



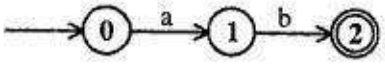
Step 3 : The machine to accept (a + b) is shown below.



Step 4 : The machine to accept (a + b)* is shown below .



Step 5 : The machine to accept ab is shown below.



Step 6 : The machine to accept ab (a + b)* is shown below .

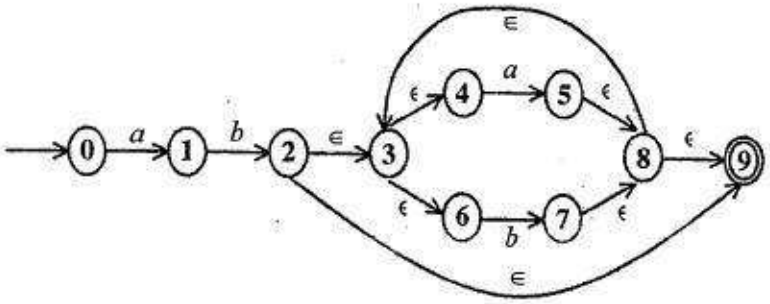
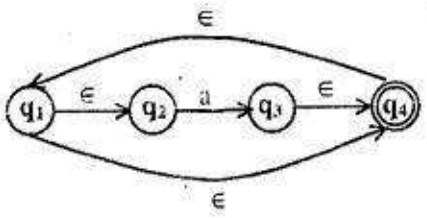


FIGURE : To accept the language (ab (a+b)*)

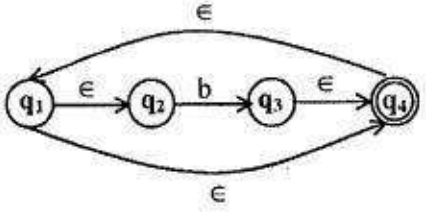
Example 6: Obtain an NFA for the regular expression $a^* + b^* + c^*$

Solution :

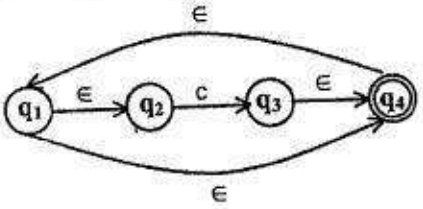
The machine corresponding the regular expression a* can be written as



The machine corresponding the regular expression b* can be written as



The machine corresponding the regular expression c* can be written as



The machine corresponding the regular expression $a^* + b^* + c^*$ is shown in below figure.

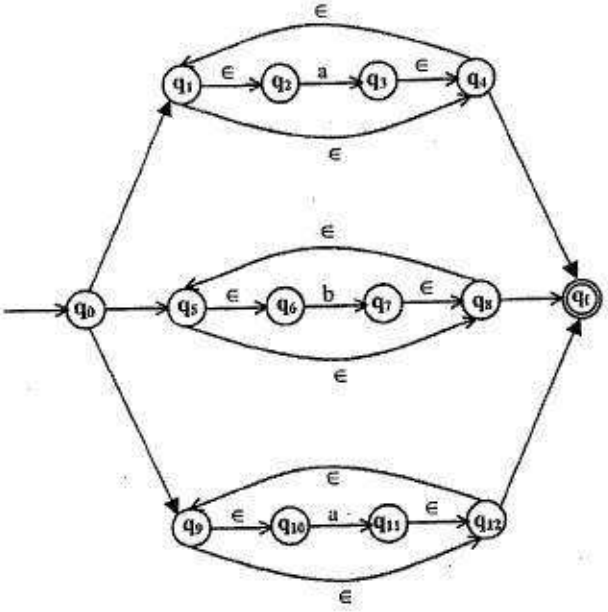
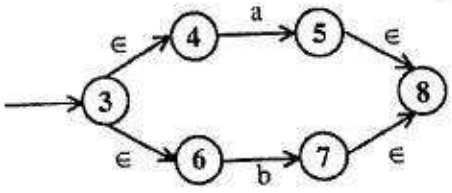


FIGURE: To accept the language $(a^* + b^* + c^*)$

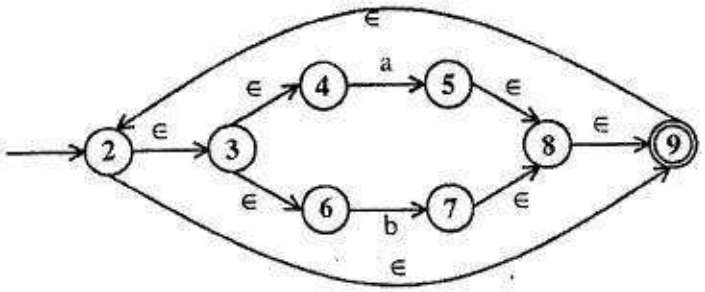
Example 7 : Obtain an NFA for the regular expression $(a+b)^* aa(a+b)^*$

Solution :

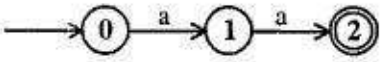
Step 1 : The machine to accept (a + b) is shown below.



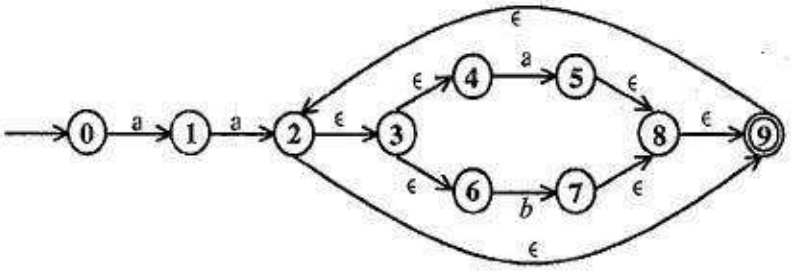
Step 2 : The machine to accept (a + b) * is shown below.



Step 3 : The machine to accept aa is shown below.



Step 4 : The machine to accept aa (a + b)* is shown below .



Step 5 : The machine to accept (a + b) * aa (a + b)* is shown in below figure.

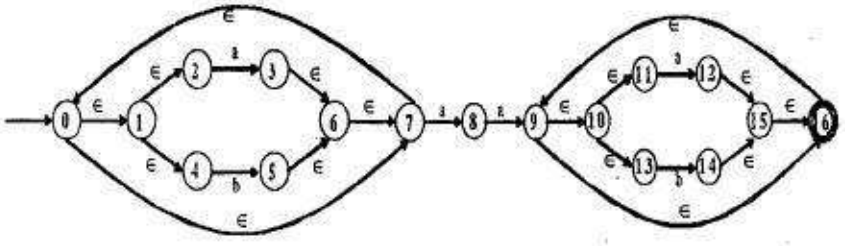


FIGURE : NFA to accept (a + b)* aa (a + b)*

Example 8 : Construction of DFA equivalent to a regular expression $(0+1)^*(00+11)(0+1)^*$ and also find the reduced DFA.

Solution : Given regular expression is $(0+1)^*(00+11)(0+1)^*$

Step 1 : (Construction of transition graph for NFA without $\in$ – moves).
First of all construct the transition graph with $\in$ using the construction rules

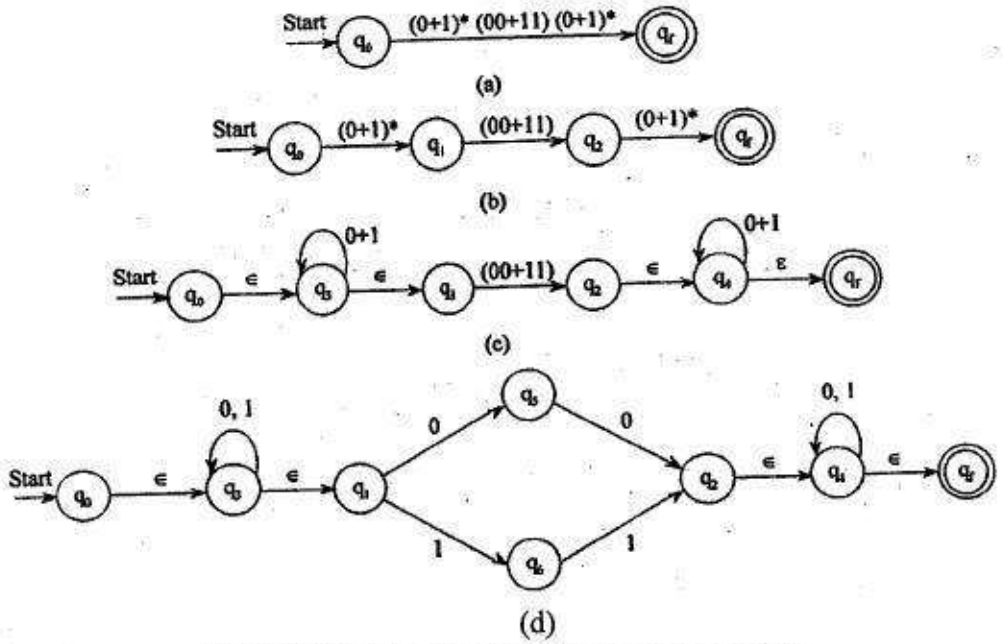


FIGURE: NFA for the given Regular Expression

Transition graph for NFA without $\in$ – moves is :

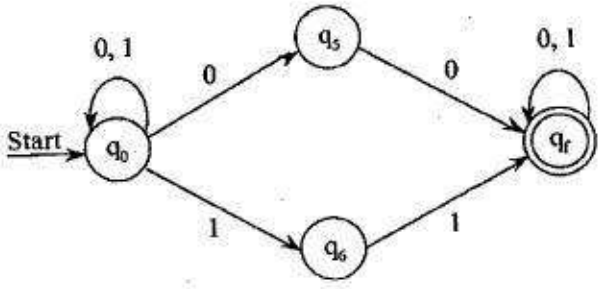


FIGURE : NFA without $\in$ - moves

Step 2 : We construct the transition table for NFA as given in below table :

	0	1
→ q_0	$\{q_0, q_5\}$	$\{q_0, q_6\}$
q_5	$\{q_f\}$	-
q_6	-	$\{q_f\}$
(q_f)	$\{q_f\}$	$\{q_f\}$

FIGURE: NFA Transition Table

Step 3 : Construct DFA table for NFA.

	Input	
States	0	1
→ $\{q_0\}$	$\{q_0, q_5\}$	$\{q_0, q_6\}$
$\{q_0, q_5\}$	$\{q_0, q_5, q_f\}$	$\{q_0, q_6\}$
$\{q_0, q_6\}$	$\{q_0, q_5\}$	$\{q_0, q_6, q_f\}$
$(\{q_0, q_5, q_f\})$	$\{q_0, q_5, q_f\}$	$\{q_0, q_6, q_f\}$
$(\{q_0, q_6, q_f\})$	$\{q_0, q_5, q_f\}$	$\{q_0, q_6, q_f\}$

FIGURE: DFA Transition Table

The state diagram for the successor table is the required DFA as shown in below figure .

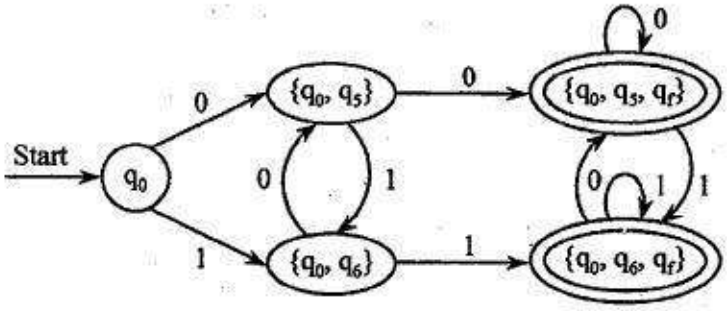


FIGURE: Required DFA for Regular expression $(0+1)^*(00+11)(0+1)^*$

As q_f is the only final state of NFA, $\{q_0, q_5, q_f\}$ and $\{q_0, q_6, q_f\}$ are the final states of DFA.

Reduce the Number of States of above DFA

As the rows corresponding to $\{q_5, q_5, q_f\}$ and $\{q_0, q_6, q_f\}$ are identical and delete the last row $\{q_0, q_6, q_f\}$.

States	Input 0	1
→ $\{q_0\}$	$\{q_0, q_5\}$	$\{q_0, q_6\}$
$\{q_0, q_5\}$	$\{q_0, q_5, q_f\}$	$\{q_0, q_6\}$
$\{q_0, q_6\}$	$\{q_0, q_5\}$	$\{q_0, q_6, q_f\}$
$\{q_0, q_5, q_f\}$	$\{q_0, q_5, q_f\}$	$\{q_0, q_6, q_f\}$

FIGURE : Reduced Transition Table of DFA

The reduced DFA transition diagram is,

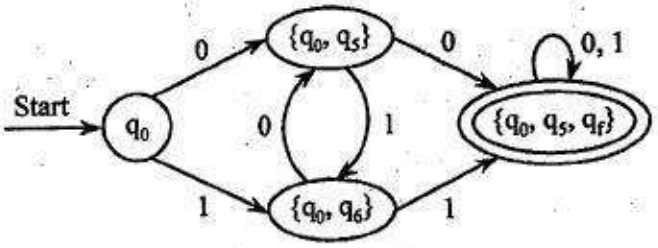


FIGURE : Reduced DFA for Regular Expression $(0+1)^*(00+11)(0+1)^*$

3.6 CONVERSION OF FA TO RE

Theorem : If L is accepted by a DFA, then L is denoted by a regular expression.

Proof : Let L be the set accepted by the DFA,

$$M = (\{q_1, q_2, \ldots\ldots q_n\}, \Sigma, \delta, q_1, F)$$

Let R_{ij}^{k} denote the set of all strings x such that $\delta(q_i, x) = q_j$ and if $\delta(q_i, y) = q_l$ for any y that is a prefix (initial segment) of x, other than x or $\in$, then $l \le k$, i.e., R_{ij}^{k} is the set of all strings that take the finite automaton from state q_i to state q_j without going through any state numbered higher than k.

R_{ij}^{k} can be defined recursively as,

$$R_{ij}^{k} = R_{ij}^{k-1}\left(R_{kk}^{k-1}\right)^{*} R_{kj}^{k-1} \cup R_{ij}^{k-1} \qquad \text{..... (1)}$$

$$R_{ij}^{0} = \begin{cases} \{a / \delta(q_i, a) = q_j & \text{if } i \neq j \\ \{a / \delta(q_i, a) = q_j\} \cup \{\in\} & \text{if } i = j \end{cases}$$

To show that for each i, j and k, there exists a regular expression R_{ij}^{k} denoting the language R_{ij}^{k} i.e., by applying induction on k.

Basis Step :

If $(k = 0)$, R_{ij}^{0} is a finite set of strings each of which is either $\in$ or a single symbol.

r_{ij}^{0} can be expressed as,

$$r_{ij}^{0} = a_1 + a_2 + \ldots\ldots + a_p \; (or \; r_{ij} = a_1 + a_2 + \ldots\ldots + a_p + \in \; if \; i = j)$$

Where, $\{a_1, a_2, \ldots, a_p\}$ is the set of all symbols 'a' such that $\delta(q_i, a) = q_j$.

If there are no such a's, then ϕ (or $\in$ in the case i = j) serves as r_{ij}^{0}.

Induction :

The recursive formula for R_{ij}^{k} given in (1) clearly involves only the regular expression operators.

By induction hypothesis, for each 1 and m, a regular expression r_{lm}^{k-1} such that,

$$L\left(r_{lm}^{k-1}\right) = R_{lm}^{k-1}$$

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

Which completes the induction.

To complete proof observe that $L(M) = \bigcup_{q_j \in F} R_{1j}^{n}$

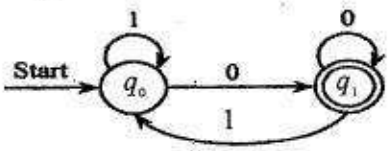
Since R_{1j}^{n} denotes the labels of all paths from q_1 to q_j.

$\therefore$ $L(M)$ is denoted by regular expression,

$$L(M) = r_{1j_1}^{n} + r_{1j_2}^{n} + r_{1j_p}^{n}$$

Where, $F = \{q_{j_1}, q_{j_2}, \ldots . q_p\}$

Example 1: Write equivalent regular expression for the following deterministic finite automaton.



Solution : A table is constructed as shown in below Table (K starts from 0 to number of states in the design) and the entries are calculated according to theorem.

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

r_{ij}^0 values are calculated as, $r_{ij}^0 = \begin{cases} \{a / \delta(q_i,a) = q_j\} & \text{if } i \neq j \\ \{a / \delta(q_i,a) = q_j\} & \text{if } i = j \end{cases}$

r_{11}^0: $\delta(q_0,0) = q_1$ not satisfying above condition

$\delta(q_0,1) = q_0$ satisfying above condition and $\in$ is default added because i = j condition.

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

r_{12}^0: $\delta(q_0,0) = q_1$ satisfying condition $(\because i \neq j)$

$\delta(q_0,1) = q_0$ not satisfying condition

$r_{12}^0 = 0$

r_{21}^0: $\delta(q_1,0) = q_1$ not satisfying

$\delta(q_1,1) = q_0$ satisfying condition

$\therefore r_{21}^0 = 1 (i \neq j)$

r_{22}^0: $\delta(q_1,0) = q_1$ satisfying condition

$\delta(q_1,1) = q_0$ not satisfying condition

$\therefore r_{22}^0 = 0+ \in (i = j)$

r_{11}^1 : Where k = 1 we have to apply,

$r_{ij}^k = r_{ik}^{k-1} \left(r_{kk}^{k-1}\right)^* \left(r_{kj}^{k-1}\right) \cup r_{ij}^{k-1}$

$r_{11}^1 = r_{11}^0 \left(r_{11}^0\right)^* \left(r_{11}^0\right) \cup r_{11}^0$

Considering values from table (k = 0),

$r_{11}^{1} = (1+\epsilon)(1+\epsilon)^*(1+\epsilon) \cup (1+\epsilon)$

Applying $(\epsilon + r r^*) = r^*$

$= 1+\epsilon((1+\epsilon)^*(1+\epsilon)+\epsilon)$

$= (1+\epsilon)(1+\epsilon)^*$ $(\because (1+\epsilon^*) = 1^*)$

$= (1+\epsilon)\,1^*$

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

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

$= 0((1+\epsilon)(1+\epsilon)^*+\epsilon)$ $(\because \epsilon + r\,r^* = r^*)$

$= 0\,(1+\epsilon)^*$

$= 0\,1^*$

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

$= 1\,(1+\epsilon)^*(1+\epsilon)+1$

$= 1((1+\epsilon)^*(1+\epsilon)+\epsilon)$

$= 1\,(1+\epsilon)^*$

$= 1\,1^*$

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

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

$= 1\,1^*0+0+\epsilon$

Now the complete construction of regular expression is, in the given FA the starting state is q_0 and final state q_1. Write expressing from starting to all final states by taking k as total number of states.

r_{12}^{2} is final term to construct regular expression.

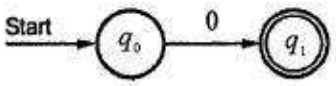
$r_{12}^{2} = (r_{12}^{1})(r_{22}^{1})^*(r_{22}^{1}) \cup (r_{12}^{1})$

$= 0\;1^*(1\,1^*0+0+\epsilon)^*(1\,1^*0+0+\epsilon)+0\,1^*$

$= 0\;1^*((1\,1^*0+0+\epsilon)^*(1\,1^*0+0+\epsilon)+\epsilon)$

$= 0\;1^*(1\,1^*0+0+\epsilon)^*$ $(\because \epsilon + r\,r^* = r^*)$

Example 2: Construct the regular expression for the finite automata given in below figure.



Solution :

	k = 0
r_{11}	$\in$
r_{12}	0
r_{21}	ϕ
r_{22}	$\in$

In above table, we have calculated the values as r_{ij} will indicate the set of all the input string from q_i to q_j. If i = j then we add $\in$ with the input string. If $i \neq j$ and there is no path from q_i to q_j then we add ϕ.

Let us compute r_{11}^0

r_{11}^0 where $i = 1, j = 1, k = 0$. There is no path from q_i to q_j but i = j. So we add $\in$ in the k = 0 column at r_{11}^0 row.

Similarly

r_{12}^0 = The input from q_0 to q_1

$r_{12}^0 = 0$

r_{21}^0 = No input from q_1 to q_0 and $i \neq j$

So we add ϕ over there.

r_{22}^0 = No input from q_1 to q_1, since i = j.

We will add $\in$.

Let us build the table when k = 1

	k = 1	
	Computation	**Regular Expression**
r_{11}^1	$r_{ij}^k = r_{ik}^{k-1}\left(r_{kk}^{k-1}\right)^* r_{kj}^{k-1} + r_{ij}^{k-1}$ $i=1, j=1, k=1$ $r_{11}^1 = r_{11}^0\left(r_{11}^0\right)^*\left(r_{11}^0\right) + r_{11}^0$ $= \in (\in)^* (\in) + \in$ $r_{11}^1 = \in$	$\in$
r_{12}^1	$i=1, j=2, k=1$ $r_{12}^1 = r_{11}^0\left(r_{11}^0\right)^*\left(r_{12}^0\right) + r_{12}^0$ $r_{12}^1 = \in (\in)^* (0) + 0$ $= \in .0 + 0$ $= 0 + 0$ $= 0$	0
r_{21}^1	$i=2, j=1, k=1$ $r_{21}^1 = r_{21}^0\left(r_{11}^0\right)^*\left(r_{11}^0\right) + \left(r_{21}^0\right)$ $= \phi(\in)^* \in + \phi$ $= \phi + \phi \quad \therefore \phi \in = \phi$ $= \phi$	ϕ
r_{22}^1	$i=2, j=2, k=1$ $r_{22}^1 = r_{21}^0\left(r_{11}^0\right)^*\left(r_{12}^0\right) + r_{22}^0$ $= \phi.(\in)^* (0) + \in$ $= \phi + \in$ $= \in$	$\in$

Now let us compute for final state, which denotes the regular expression.

r_{12}^2 will be computed, because there are total 2 states and final state is q_1 whose start state is q_0.

$$r_{12}^2 = (r_{12}^1)(r_{22}^1)^*(r_{22}^1)+(r_{12}^1)$$
$$= 0(\in)^*(\in) + 0$$
$$= 0 + 0$$

$r_{12}^2 = 0$ which is a final regular expression.

3.6.1 Arden's Method for Converting DFA to RE

As we have seen the Arden's theorem is useful for checking the equivalence of two regular expressions, we will also see its use in conversion of DFA to RE.

Following algorithm is used to build the r. e. from given DFA.

1. Let q_0 be the initial state.
2. There are $q_1, q_2, q_3, q_4, \ldots q_n$ number of states. The final state may be some q_j where $j \leq n$.
3. Let α_{ij} represents the transition from q_j *to* q_i.
4. Calculate q_i such that

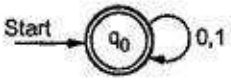
$$q_i = \alpha_{ji} \cdot q_j$$

If q_i is a start state

$$q_i = \alpha_{ji} \cdot q_j + \in$$

5. Similarly compute the final state which ultimately gives the regular expression r.

Example 1 : Construct RE for the given DFA.



Solution :

Since there is only one state in the finite automata let us solve for q_0 only.

$$q_0 = q_0 0 + q_0 1 + \in$$
$$q_0 = q_0 (0 + 1) + \in$$

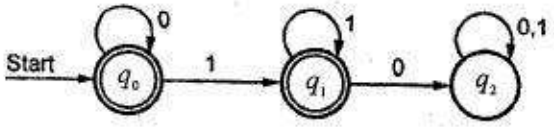
$= \in .(0 + 1)^*$ $\because R = Q + RP$

$q_0 = (0+1)^*$

Since q_0 is a final state, q_0 represents the final r. e. as

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

Example 2 : Construct RE for the given DFA.



Solution : Let us build the regular expression for each state.

$q_0 = q_0 0 + \in$

$q_1 = q_0 1 + q_1 1$

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

Since final states are q_0 and q_1, we are interested in solving q_0 and q_1 only.

Let us see q_0 first

$q_0 = \in + q_0 0$

Which is R = Q + R P equivalent so we can write

$q_0 = \in .(0)^*$

$q_0 = 0^*$ $\because \in .R = R$

Substituting this value into q_1, we will get

$q_1 = 0^* 1 + q_1 1$

$q_1 = 0^* 1 (1)^*$ $\because R = Q + RP \Rightarrow QP^*$

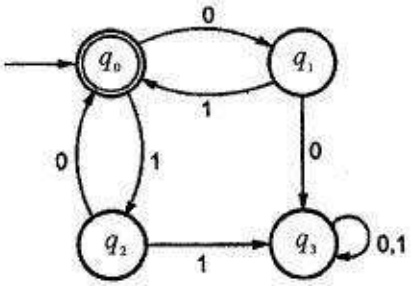
The regular expression is given by

$r = q_0 + q_1$

$= 0^* + 0^* 1 . 1^*$

$r = 0^* + 0^* 1^+$ $\because 1 . 1^* = 1^+$

Example 3 : Construct RE for the DFA given in below figure.



Solution : Let us see the equations

$$q_0 = q_1 1 + q_2 0 + \in$$
$$q_1 = q_0 0$$
$$q_2 = q_0 1$$
$$q_3 = q_1 0 + q_2 1 + q_3 (0+1)$$

Let us solve q_0 first,

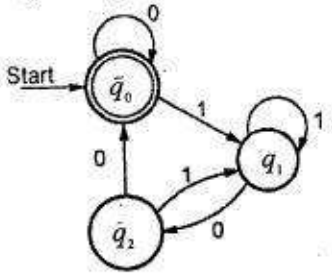
$$q_0 = q_1 1 + q_2 0 + \in$$
$$q_0 = q_0 01 + q_0 10 + \in$$
$$q_0 = q_0 (01+10) + \in \qquad \because R = Q + RP$$
$$q_0 = \in (01+10)^* \qquad \Rightarrow QP^* \text{ where}$$
$$q_0 = (01+10)^* \qquad R = q_0, Q = \in, P = (01+10)$$

Thus the regular expression will be

$$r = (01+10)^*$$

Since q_0 is a final state, we are interested in q_0 only.

Example 4 : Find out the regular expression from given DFA.



Solution : Let us solve the DFA by writing the regular expression, for each state .

$$q_0 = q_0 0 + q_2 0 + \in \qquad \because \text{Initial state}$$
$$q_1 = q_1 1 + q_2 1 + q_0 1$$
$$q_2 = q_1 0$$

For getting the r. e. we have to solve q_0 the final state.

$$q_1 = q_1 1 + q_1 01 + q_0 1$$
$$q_1 = q_1 (1 + 01) + q_0 1$$

We will compare R = Q + R P with above equation, so $R = q_1, Q = q_0 1,\ P = (1 + 01)$ which ultimately gets reduced to QP*.

$$q_1 = q_0 1(1 + 01)^*$$

Substituting this value to q_0

$$q_0 = q_0 0 + q_2 0 + \in$$
$$= q_0 0 + q_1 00 + \in$$
$$= q_0 0 + q_0 (1(1 + 01)^*)00 + \in$$
$$q_0 = q_0 (0 + 1(1 + 01)^* 00) + \in$$

Again $R = Q + RP$

Where $R = q_0$

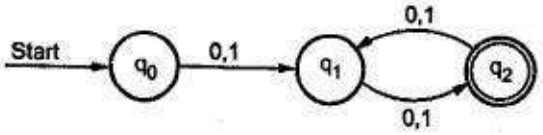
$Q = \in$

$P = 0 + 1(1 + 01)^* 00$

Hence $q_0 = \in .[0 + p(1 + 01)^* .00]^*$

$q_0 = [0 + 1(1 + 01)^* .00]^* \because \in .R = R$

Example 5 : Construct the regular expression for following DFA.



Solution : We can get the regular expression from state q_2. Let us see the equation of each state.

$$q_0 = \in$$
$$q_1 = q_0 1 + q_0 0 + q_2 1 + q_2 0$$
$$q_2 = q_1 1 + q_1 0$$

Putting value of q_0 in q_1

$$q_1 = \in .1 + \in .0 + q_2(0+1)$$
$$q_1 = (1+0) + q_2(0+1)$$

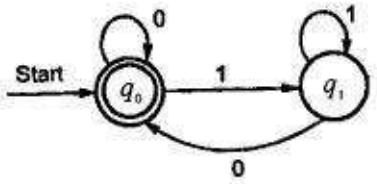
Now solve q_2

$$q_2 = (1+0)\, q_1$$
$$= ((1+0))\,[(1+0) + q_2(1+0)]$$
$$q_2 = (1+0).\,(1+0) + q_2(1+0)\,(1+0)$$

Here $R = q_2$, $Q = (1+0)\ (1+0)$, $P = (1+0)\ (1+0)$

$q_2 = (1+0)\ (1+0)\,[(1+0)\ (1+0)]^*$ is a regular expression.

Example 6 : Give the regular expression of following DFA.



For given DFA we can write the equation

$$q_0 = q_0 0 + q_1 0 + \in \quad \text{.... (1)}$$
$$q_1 = q_0 1 + q_1 1 \quad \text{... .(2)}$$

By theorem R = Q + RP we get R = QP*

$$R = q_1$$
$$Q = q_0 1$$
$$P = 1$$
$$\therefore \quad q_1 = q_0 11^*$$

As we know $R^+ = RR^*$ we can also write

$q_1 = q_0 1^+$

Let us put value of q_1 in equation (1)

$$q_0 = q_0 0 + q_0 1^+ 0 + \epsilon$$
$$q_0 = q_0 (0 + 1^+ 0) + \epsilon$$

Again we will apply R = Q + RP gives QP*

$R = q_0$

$Q = \epsilon$

$P = 0 + 1^+ 0$

$q_0 = \epsilon . (0 + 1^+ 0)^*$

$q_0 = (0 + 1^+ 0)^*$ $\quad \because R\epsilon = \epsilon R = R$

In the given DFA, q_0 is a final state the equation computed for state q_0 will be regular expression. Hence r. e. for above DFA is

$r.e. = (0 + 1^+ 0)^*$

3.7 REGULAR AND NON - REGULAR LANGUAGES

The languages accepted by finite automata are described by regular expressions. So to prove a language is accepted by finite automata it is sufficient to prove the regular expression of that language is accepted by finite automata.

The languages which are accepted by some finite automata are called regular languages. Here it means that the FA accepts only the words of this language and does not accept any word outside it.

1. Some of the words of the language are not accepted by FA.

(or)

2. All the words of the language are accepted in addition to that some extra strings are also accepted.

All languages are either regular or non regular, none of the languages are both.

By looking at some of the languages we can say whether they are regular or not.

i) The languages whose words need some sort of comparison can never be regular.

Example : $L = \{a^n b^n, n \geq 0\}$

Here the number of a's must be equal to number of b's for each 'a' we check the existence of b which cannot be done using FA.

ii) The languages whose words are in arithmetic progression and need no comparisons will be regular.

Example : 1. $L = \{a^{2n}, n \geq 1\}$

The words of this language are , *aa*, *aaaa*, *aaaaaa*, ..., a^{2n} which are in A.P with period 2. Hence it is a regular language.

2. $L = \{ a^p, p \text{ is prime} \}$

The words of this language are $\{ a, aa, aaa, aaaaaaa, ... a^p \}$. We can see these words are not in A.P. Hence it is not regular.

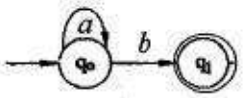
In this section, we will discuss how to prove that certain language is not regular (non - regular) language. Pumping Lemma is a useful tool to prove that a certain language is not regular language.

Since, the number of states in a FA is finite, say it is n (for some fixed value of n), and then it can recognize all the words of length less than n without any loop. Suppose, a regular language L has infinite number of words and the length of these words may or may not be equal to n. So, how can a FA recognize the L ? A FA can recognize L having some loop(s) and whenever the length of a given word is greater than or equal to n. So, we conclude that the loop in FA makes it able to accept those strings, which have length greater than or equal to its total number of states.

When a string z has bigger length (greater than number of states in FA) then we break this string into three parts, say u, v (v should not be null string), and w. Let FA has loop for v, and $z = uvw \in L$ is accepted by FA.

So, $z = uv^i w$ for i = 0,1, ... is also accepted by FA having some loop for v. This is the main concept, used in Pumping Lemma.

Now, consider a regular language L = a*b and corresponding FA shown in below figure.



We see the list of accepted strings given below :

b , ab, aab, aaab, ...

Let $u = \in, v = a$ (v should not be $\in$), and w = b, then a^i i.e. $z = uv^i w$ for some i = 0, 1, ... is accepted by FA. Now, we have good base to discuss the Pumping Lemma.

3.8 Pumping Lemma for Regular Sets

Pumping Lemma is useful because

1. It gives a method for pumping (generating) many substrings from a given string. In other words, we say, it provides means to break a given long input string into several substrings.

2. It gives necessary condition(s) to prove a set of strings is not regular.

Theorem :

Let $M = (Q, \Sigma, \delta, q_0, F)$ be a DFA having n states. M recognizes the language L. A long string $z \in L$ such that $|z| \geq n$ and $z = uvw$, where $v \neq \in$, then $uv^i w \in L$ for $i \geq 0$.

Proof :

M recognizes L and L is a regular set. If $z \in L$ such that $w = uvw$. Here v is optional in z and $|z| \geq n$, where n is the number of states in DFA.

Consider following DFA shown in below figure.

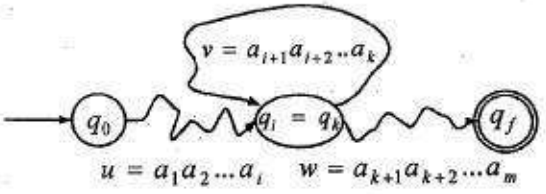


FIGURE : DFA for $uv^i w$

Let $z = a_1a_2a_3...a_ia_{i+1}...a_ka_{k+1}..a_m$

Where $u = a_1a_2a_3..a_i, v = a_{i+1}...a_k$ and $w = a_{k+1}...a_m$

The length of z is m and $m \geq n$. It means, when $m \geq n$, it indicates that there is some loop in transition diagram of M. Let v is the string obtained from the edges involved in looping as shown in above figure.

Case 1 : When $z = uv^0w = uw$ for $i = 0$, it means, uw is accepted and $uw \in L$.

Case 2 : $z = uv^iw$ for $i \geq 1$, it means that control of DFA M goes i - times into the loop with label v and uv^iw is accepted by M.

So, for all values of $i \geq 0$, $z = uv^iw$ is accepted by M.

Hence, the statement of the theorem is proved.

Application of Pumping Lemma

Pumping Lemma is used to prove certain sets are not regular sets. This is done as follows :

Step 1 : We assume that given set is regular and accepted by DFA M having n states.

Step 2 : Choose a string z such that $|z| \geq n$ and use Pumping Lemma to write $z = uv^iw$ for $i \geq 0, v \notin \in$, and $|uw| \leq n$.

Step 3 : Find a suitable integer i such that $uv^iw \notin L$ and this contradicts our assumption made in step 1 and hence L is not regular.

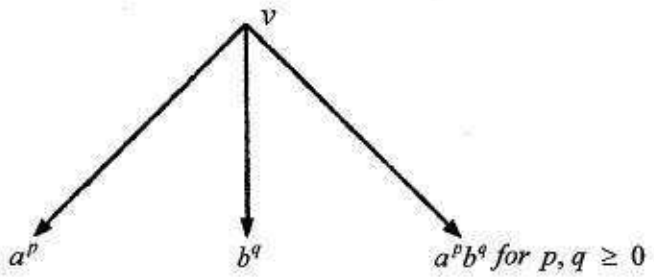
Example 1 : Prove that $L = \{a^nb^n : n \geq 1\}$ is not regular.

Solution : In given language the number of $a's$ is equal to the number of $b's$. This is the one clue to find the contradiction.

Step 1 : Let L is regular and accepted by DFA M with n states.

Step 2 : String $z \in L$ such that $|z| \geq n$ and $z = uv^iw \in L$ for $i \geq 0, v \notin \in$, and $|uw| \leq n$.

Step 3 : Selecting substring v



Let $z = uv^i w$ for $i = 0$

Case 1 : When $v = a^p$, then

$z = a^{n-p} b^n$

Number of $a's = n - p$, and number of $b's = n$

Number of $a's$ = Number of $b's$ if and only if $p = 0$ and number of $a's$ and $b's$ is not equal when $p > 0$.

So, for $p > 0$, $z = uv^i w \notin L$

Case 2 : When $v = a^q$, then

$z = a^n b^{n-q}$

Number of $a's = n$, and number of $b's = n - q$

Number of $a's$ = Number of $b's$ if and only if $q = 0$ and number of $a's$ and $b's$ is not equal when $q > 0$.

So, for $q > 0$, $z = uv^i w \notin L$.

Case 3 : When $v = a^p b^q$, then $z = a^{n-p} b^{n-q}$

Number of $a's = n - p$, and number of $b's = n - q$.

Number of $a's$ = Number of $b's$ if and only if $q = p$.

So, for $p \neq q$, $z = uv^i w \notin L$

Since, we get contradiction in all the cases, therefore L is not regular.

Example 2 : Show that $L = \{a^n b^n | n \geq 0\}$ is not regular.

Solution :

Step 1 : Let L is regular and n be the number of states in FA. Consider the string $z = a^n b^n$.

Step 2 : Note that |z|=2n and is greater than n. So, we can split z into uvw such that $|uv| \leq n$ and $|v| \geq 1$ as shown below.

$$z = \underbrace{aaaaaa}_{u} \underbrace{a}_{v} \underbrace{bbbbbbb}_{w}$$

where $|u| = n-1$ and $|v| = 1$ so that $|uv| = |u| + |v| = n-1+1 = n$ and |w| = n. According to pumping lemma, $uv^i w \in L$ for i = 0, 1, 2, ...

Step 3 : If i is 0 i. e., v does not appear and so the number of a's will be less than the number of b's and so the string w does not contain some number of a's followed by same number of b's (equal to that of a's)

Similarly, if i = 2, 3,..., then number of a's will be more than the number of b's and so number of a's followed by equal number of b's does not exist. But, according to pumping lemma, n number of a's should be followed by n number of b's which is a contradiction to the assumption that the language is regular. So, the language L is not regular.

Example 3: Prove that $L = \{a^{i^2} : i \geq 1\}$ is not regular

Solution :

Method - I (Using Pumping Lemma for regular sets)

In L, all words have their lengths in perfect square and this is the clue for proving non - regular.

Step 1 : Let L be regular and accepted by DFA M with n states.

Step 2 : String $z \in L$ such that $|z| \geq n$ and $z = uv^i w \in L$ for $i \geq 0, y \notin \in$, let $|z| = n^2 \geq n$, and $|uw| \leq n$, (n is the number of states).

Step 3 : Since, length of v can not exceed n (the number of states), it means, $|v| \leq n$.

Let i = 2, so $z = uv^2 w \in L$, and

$|z| = |uvw| + |v| = n^2 + |v|$

So, $n^2 \leq |z| \leq n^2 + n$ (Since, $|v| \leq n$)

Or, $n^2 \leq |z| \leq n^2 + n + (n+1)$ (Adding n + 1 to make perfect square)

Or, $n^2 \le |z| \le (n+1)^2$

It means, the length of z is between n^2 and $(n+1)^2$, and is not a perfect square. Therefore, L is not regular.

Method - II

For example,

$z = a^{i^2}$

Let i = 2
z = aaaa
z = uvw
Assume uvw = aaaa
Take u = a
v = aa
w = a

By pumping lemma, even if we pump v i.e. increase v then language should show the length as perfect square .

uvw
= uv.vw
= aaaaaa
= length of a is not a perfect square

Thus the behaviour of the language is not regular, as after pumping something onto it does not show the same property (being square for this example.)

Example 4 : Show that $L = \{ww^R | w \in (0+1)^*\}$ is not regular.

Solution :

Step 1 : Let L is regular and n be the number of states in FA. Consider the string

$$z = \overbrace{\underbrace{1.....1}_{n}\underbrace{0.....0}_{n}}^{w}\overbrace{\underbrace{0.....0}_{n}\underbrace{1.....1}_{n}}^{w^R}$$

where n is the number of states of FA, w = 1 ... 10 .... 0 and reverse of w i. e., w^R = 0....01.....1.

Step 2 : Split the string z into uvw such that $|uv| \leq n$ and $|v| \geq 1$ as shown below.

$$z = \underbrace{1 \ldots 1}_{u}\ \underbrace{0 \ldots 0}_{v}\ \underbrace{0 \ldots 0}_{w}\ \underbrace{0 1 \ldots 1}$$

where $|u| = n-1$ and $|v| = 1$ so that $|uv| = |u| + |v| = n-1+1 = n$ which is true. According to pumping lemma, $uv^i w \in L$ for i = 0, 1, 2, ...

Step 3 : If i is 0 i. e., v does not appear and so the number of 1's on the left of z will be less than the number of 1's on the right of z and so the string is not in the form ww^R. So, $uv^i w \notin L$ when i = 0. This is a contradiction to the assumption that the language is regular. So, ww^R is not regular.

Example 5 : Show that $L = \{ 0^{2n} \mid n \geq 1 \}$ is regular.

Solution : This is a language length of string is always even.

i.e. n = 1 ; z = 00

n = 2 ; z = 00 00 and so on.

Let z = uvw

$z = 0^{2n}$

$|z| = 2^n = uv^i w$

If we add 2n to this string length.

$|z| = 4n = uv.vw$

= even length of string.

Thus even after pumping 2n to the string we get the even length. So the language L is regular language.

Example 6: Prove that L= { ww | w in (a + b)* } is not regular.

Solution : Prove the result by the method of contradiction.

Step 1 : Suppose L is regular, let 'n' be the number of states in the automaton M accepting 'L'.

Step 2 : Let us consider $ww = a^n b^n a^n b$ in L. $|ww| = 2(n+1) > n$ apply pumping lemma we write ww = xyz with $|y| \neq 0$, $|xy| \leq n$.

Step 3 : To find i such that $xy^i z \notin L$ for getting a contradiction. The string 'y' can be in only one of the following forms.

Case 1 : y has no b's i. e., $y = a^k$ for some $k \geq 1$.

Case 2 : y has only one b.

We may note that y cannot have two b's. If so $|y| \geq n+2$.

But $|y| \leq |xy| \leq n$.

In case 1, we can take i = 0 .

Then $xy^0z = xz$ is of the form $a^m ba^n b$. Where $m = n - k < n (or\ a^n b\ a^n b)$ $x\ z$ can not be written in the form uu with $u \in \{a,b\}^*$ and so $xz \notin L$.

In case 2 also. We can take i = 0.

Then $xy^0z = xz$ has only one b.

So $xz \notin L$ as any element in L should have even number of a's and even number of b's.

Thus in both cases we get contradiction.

$\therefore$ L is not regular.

Example 7 : Show that $L = \{a^p \mid p\ is\ a\ prime\ number\}$ is not regular.

Method - I :

Step 1 : Let L is regular and get a contradiction. Let n be the number of states in the FA accepting L.

Step 2 : Let p be a prime number greater than n. Let $z = a^p$. By pumping lemma, z can be written as $z = uvw$, with $|uv| \leq n$ and $|v| > 0$. u, v, w are simply strings of a's. So, $v = a^m$ for some $m \geq 1$ (and $\leq$ n).

Step 3 : Let i = p + 1. Then $|uv^i w| = |uvw| + |v^{i-1}| = p + (i-1)m = p + pm$. By pumping lemma, $uv^i w \in L$. But $|uv^i w| = p + pm = p(1+m)$ and $p(1+m)$ is not a prime. So $uv^i w \notin L$. This is a contradiction. Thus L is not regular.

Method - II : Let us assume L is a regular and P is a prime number.

$z = a^p$

$|z| = uvw$ i = 1

Now consider $z = uv^i w$ where i = 2

$= uv.vw$

Adding 1 to P we get,

$P < |uvvw|$

$P < P + 1$

But P + 1 is not a prime number. Hence what we have assumed becomes contradictory. Thus L behaves as it is not a regular language.

Example 8 : Show that the language $L = \{a^i\ b^{2i} | i > 0\}$ is not regular.

Solution : The set of strings accepted by language L is,

$L = \{abb, aabbbb, aaabbbbbb, aaaabbbbbbbb...\}$

Applying Pumping lemma for any of the strings above.

Take the string abb.

It is of the form *uvw*.

Where, $|uv| \leq i, |v| \geq 1$

To find i such that $uv^i w \notin L$

Take i = 2 here, then

$uv^2w = a(bb)b$

$= abbb$

Hence $uv^2w = abbb \notin L$

Since abbb is not present in the strings of L.

$\therefore$ L is not regular.

Example 9 : Show that $L = \{0^n | n$ is a perfect square $\}$ is not regular.

Solution :

Step 1 : Let L is regular by Pumping lemma. Let n be number of states of FA accepting L.

Step 2 : Let $z = 0^n$ then $|z| = n \geq 2$.

Therefore, we can write z = uvw ; Where $|uv| \leq n, |v| \geq 1$.

Take any string of the language L = { 00, 0000, 000000 .... }

Take 0000 as string, here u = 0, v = 0, w = 00 to find i such that $uv^i w \notin L$.

Take i = 2 here, then

$uv^i w = 0(0)^2 00$

$= 00000$

This string 00000 is not present in strings of language L. So $uv^i w \notin L$.

$\therefore$ It is a contradiction.

3.9 PROPERTIES OF REGULAR SETS

Regular sets are closed under following properties.

1. Union
2. Concatenation

3. Kleene Closure
4. Complementation
5. Transpose
6. Intersection

1. Union : If R_1 and R_2 are two regular sets, then union of these denoted by $R_1 + R_2$ or $R_1 \cup R_2$ is also a regular set.

Proof : Let R_1 and R_2 be recognized by NFA N_1 and N_2 respectively as shown in Figure1(a) and Figure1(b).

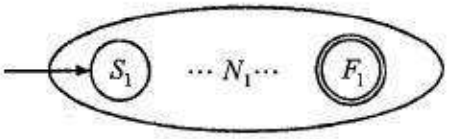


FIGURE 1(a) NFA for regular set R_1

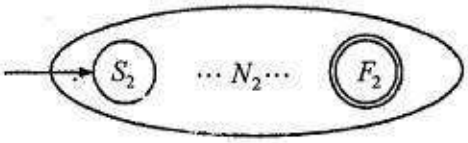


FIGURE 1(b) NFA for regular set R_2

We construct a new NFA N based on union of N_1 and N_2 as shown in Figure 1 (c)

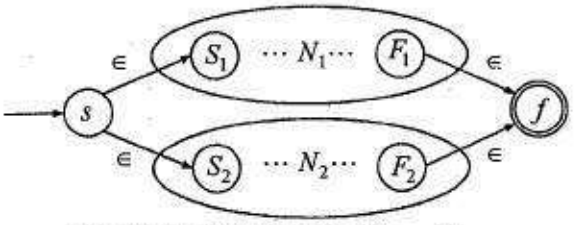


FIGURE 1(c) NFA for $N_1 + N_2$

Now,

$$L(N) = \in L(N_1) \in + \in L(N_2) \in$$
$$= \in R_1 \in + \in R_2 \in$$
$$= R_1 + R_2$$

Since, N is FA, hence $L(N)$ is a regular set (language). Therefore, $R_1 + R_2$ is a regular set.

2. Concatenation : If R_1 and R_2 are two regular sets, then concatenation of these denoted by R_1R_2 is also a regular set.

Proof : Let R_1 and R_2 be recognized by NFA N_1 and N_2 respectively as shown in Figure 2(a) and Figure 2(b).

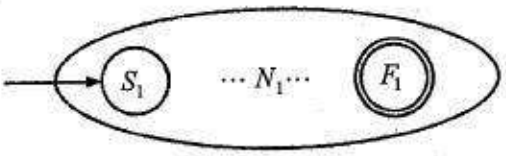


FIGURE 2(a) NFA for regular set R_1

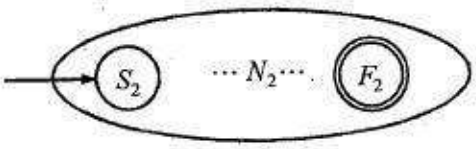


FIGURE 2(b) NFA for regular set R_2

We construct a new NFA N based on concatenation of N_1 and N_2 as shown in Figure2(c).

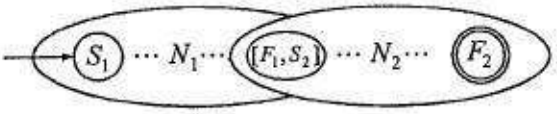


FIGURE 2(c) NFA for regular set R_1R_2

Now,

$L(N)$ = Regular set accepted by N_1 followed by regular set accepted by $N_2 = R_1R_2$

Since, $L(N)$ is a regular set, hence R_1R_2 is also a regular set.

3. Kleene Closure : If R is a regular set, then Kleene closure of this denoted by R^* is also a regular set.

Proof : Let R is accepted by NFA N shown in Figure 3(a).

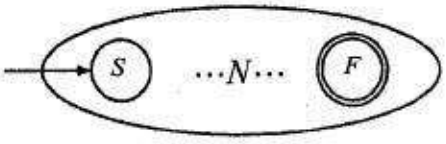


FIGURE 3(a) NFA for regular set R

We construct a new NFA based on NFA N as shown in Figure 3(b).

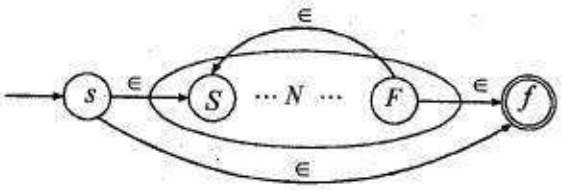


FIGURE 3(b) NFA for regular expression for R^*

Now,

$$L(N) = \{\in, R, RR, RRR, ...\}$$

$$= L^*$$

Since, $L(N)$ is a regular set, therefore R^* is a regular set.

4. **Complement :** If R is a regular set on some alphabet Σ, then complement of R is denoted by $\Sigma^* - R$ or $\overline{R}$ is also a regular set.

Proof : Let R be accepted by NFA $N = (Q, \Sigma, \delta, s, F)$. It means, $L(N) = R$. N is shown in Figure 4(a).

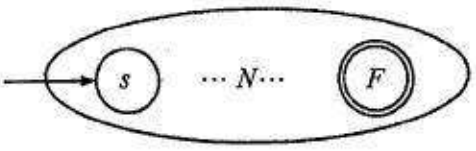


FIGURE 4(a) NFA for regular set R

We construct a new NFA N' based on N as follows :

(a) Change all final states to non-final states.
(b) Change all non-final states to final states.
N' is shown in Figure 4(b)

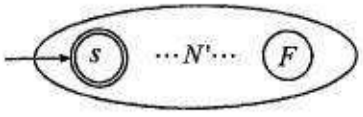


FIGURE 4 (b) NFA

Now,

$L(N')$ = {All the words which are not accepted by NFA N}

= { All the rejected words by NFA N}

$= \Sigma^* - R$

Since, $L(N')$ is a regular set, therefore $(\Sigma^* - R)$ is a regular set.

5. Transpose : If R is a regular set, then the transpose denoted by R^T, is also a regular set.

Proof : Let R be accepted by NFA $N = (Q, \Sigma, \delta, s, F)$ as shown in Figure 5(a).

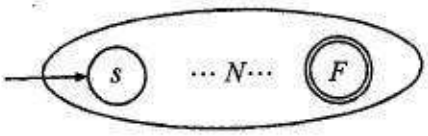


FIGURE 5 (a) NFA N for regular set R

If w is a word in R, then transpose (reverse) is denoted by w^T.

Let $w = a_1 a_2 ... a_n$

Then $w^T = a_n a_{n-1} .... a_1$

We construct a new N' based on N using following rules :

(a) Change the all final states into non-final states and merge all these into one state and make it initial state.

(b) Change initial state to final state.

(c) Reverse the direction of all edges.

N' is shown in Figure5 (b)

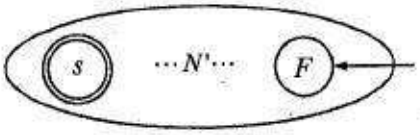


FIGURE 5(b) NFA N' for regular set R^T

Let $w = a_1 a_2 \ldots a_n$ be a word in R, then it is recognized by N and $w^T = a_n a_{n-1} \ldots a_1$ is recognized by N' as shown in Figure5 (b)

In general, we say that if a word w in R is accepted by N, and then N' accepts w^T.

Since, $L(N')$ is a regular set containing all w^T; it means, $L(N') = R^T$.

Thus, R^T is a regular set.

6. **Intersection :** if R_1 and R_2 are two regular sets over Σ, then intersection of these denoted by $R_1 \cap R_2$ is also a regular set.

Proof : By De Morgan's law for two sets A and B over R,

$A \cap B = R^* - ((R^* - A) \cup (R^* - B))$

So, $R_1 \cap R_2 = \Sigma^* - ((\Sigma^* - R_1) \cup (\Sigma^* - R_2))$

Let $R_3 = (\Sigma^* - R_1)$ and $R_4 = (\Sigma^* - R_2)$

So, R_3 and R_4 are regular sets as these are complement of R_1 and R_2.

Let $R_5 = R_3 \cup R_4$

So, R_5 is a regular set because it is the union of two regular sets R_3 and R_4.

Let $R_6 = \Sigma^* - R_5$

So, R_6 is a regular set because it is the complement of regular set R_5.

Therefore, intersection of two regular sets is also regular set.

REVIEW QUESTIONS

Q1. What is regular set ? Explain with an example.

Answer :

For Answer refer to Topic : 3.1, Page No : 3.1.

Q2. What is regular expression ? Explain with an example.

Answer :

For Answer refer to Topic : 3.2 , Page No : 3.2.

Q3. Obtain a regular expression to accept a language consisting of strings of a's and b's of even length.

Answer :

For Answer refer to example - 1 , Page No : 3.4.

Q4. Obtain a regular expression to accept a language consisting of strings of a's and b's of odd length.

Answer :

For Answer refer to example - 2 , Page No : 3.4.

Q5. Obtain a regular expression such that $L(r) = \{W \mid W \in \{0,1\}^*$ with at least three consecutive 0's $\}$.

Answer :

For Answer refer to example - 3 , Page No : 3.4.

Q6. Obtain a regular expression to accept strings of a's and b's ending with 'b' and has no substring aa.

Answer :

For Answer refer to example - 4 , Page No : 3.5.

Q7. Obtain a regular expression to accept strings of 0's and 1's having no two consecutive zeros.

Answer :

For Answer refer to example - 5 , Page No : 3.5.

Q8. Obtain a regular expression to accept strings of a's and b's of length ≤ 10.

Answer :

For Answer refer to example - 6 , Page No : 3.5.

Q9. Obtain a regular expression to accept strings of a's and b's starting with 'a' and ending with 'b'.

Answer :

For Answer refer to example - 7 , Page No : 3.6.

Q10. Explain equivalence of two REs using Arden's theorem.

Answer :

For Answer refer to Topic : 3.3.1, Page No : 3.7.

Q11. Prove $(1+00*1)+(1+00*1)(0+10*1)*(0+10*1)=0*1(0+10*1)*$

Answer :

For Answer refer to example - 1 , Page No : 3.9.

Q12. Show that $(0*1*)*=(0+1)*$

Answer :

For Answer refer to example - 2 , Page No : 3.9.

Q13. If r be a regular expression then there exists a NFA with ϵ - moves, which accepts L(R).

Answer :

For Answer refer to Topic : 3.5 , Page No : 3.10.

Q14. Construct NFA for the regular expression a + ba *.

Answer :

For Answer refer to example - 1 , Page No : 3.13.

Q15. Construct NFA with ϵ moves for the regular expression $(0+1)*$.

Answer :

For Answer refer to example - 2 , Page No : 3.15.

Q16. Construct NFA for the language having odd number of one's over the set $\Sigma = \{1\}$.

Answer :

For Answer refer to example - 3 , Page No : 3.16.

Q17. Construct NFA for the r. e. $(01+2^*)0$.

Answer :

For Answer refer to example - 4 , Page No : 3.17.

Q18. Obtain an NFA which accepts strings of a's and b's starting with the string ab.

Answer :

For Answer refer to example - 5 , Page No : 3.18.

Q19. Obtain an NFA for the regular expression $a^* + b^* + c^*$

Answer :

For Answer refer to example - 6 , Page No : 3.19.

Q20. Obtain an NFA for the regular expression $(a+b)^* aa(a+b)^*$

Answer :

For Answer refer to example - 7 , Page No : 3.20.

Q21. Construction of DFA equivalent to a regular expression $(0+1)^*(00+11)(0+1)^*$ and also find the reduced DFA.

Answer :

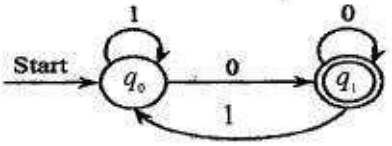
For Answer refer to example - 8 , Page No : 3.22.

Q22. If L is accepted by a DFA, then L is denoted by a regular expression.

Answer :

For Answer refer to Theorem , Page No : 3.24.

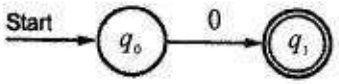
Q23. Write equivalent regular expression for the following deterministic finite automaton.



Answer :

For Answer refer to example - 1 , Page No : 3.26.

Q24. Construct the regular expression for the finite automata given in below figure.



Answer :

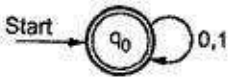
For Answer refer to example - 2 , Page No : 3.28.

Q25. Explain Arden's method for converting DFA to RE.

Answer :

For Answer refer to Topic : 3.6.1 , Page No : 3.30.

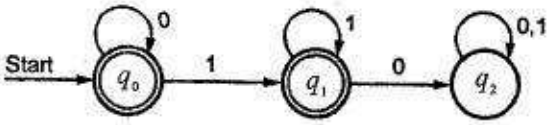
Q26. Construct RE for the given DFA.



Answer :

For Answer refer to example - 1 , Page No : 3.30.

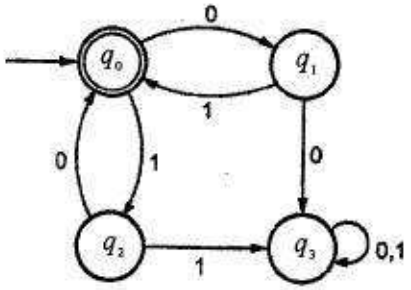
Q27. Construct RE for the given DFA.



Answer :

For Answer refer to example - 2 , Page No : 3.31.

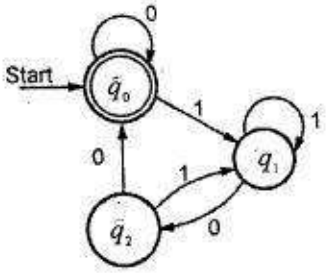
Q28. Construct RE for the DFA given in below figure.



Answer :

For Answer refer to example - 3 , Page No : 3.32.

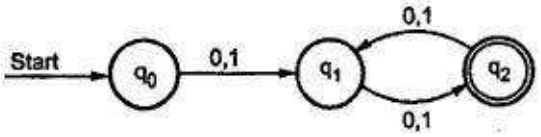
Q29. Find out the regular expression from given DFA.



Answer :

For Answer refer to example - 4 , Page No : 3.32.

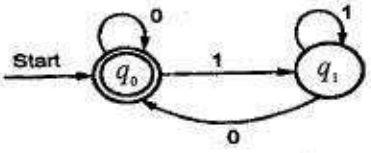
Q30. Construct the regular expression for following DFA.



Answer :

For Answer refer to example - 5 , Page No : 3.33.

Q31. Give the regular expression of following DFA.



Answer :

For Answer refer to example - 6 , Page No : 3.34.

Q32. State and prove Pumping Lemma for regular sets.

Answer :

For Answer refer to Theorem , Page No : 3.37.

Q33. Prove that $L = \{a^n b^n : n \geq 1\}$ is not regular.

Answer :

For Answer refer to example - 1 , Page No : 3.38.

Q34. Show that $L=\{a^nb^n|n\geq 0\}$ is not regular.

Answer :

For Answer refer to example - 2 , Page No : 3.40.

Q35. Prove that $L=\{a^{i^2} : i\geq 1\}$ is not regular .

Answer :

For Answer refer to example - 3 , Page No : 3.40.

Q36. Show that $L=\{ww^R|w\in(0+1)^*\}$ is not regular.

Answer :

For Answer refer to example - 4 , Page No : 3.41.

Q37. Show that $L=\{\ 0^{2^n}|\ n\geq 1\ \}$ is regular.

Answer :

For Answer refer to example - 5 , Page No : 3.42.

Q38. Prove that L= { ww | w in (a + b)* } is not regular.

Answer :

For Answer refer to example - 6 , Page No : 3.42.

Q39. Show that $L=\{a^p \mid p \text{ is a prime number}\}$ is not regular.

Answer :

For Answer refer to example - 7 , Page No : 3.43.

Q40. Show that the language $L=\{a^ib^{2i}\mid i>0\}$ is not regular.

Answer :

For Answer refer to example - 8 , Page No : 3.44.

Q41. Show that $L=\{0^n|n$ is a perfect square $\}$ is not regular.

Answer :

For Answer refer to example - 9 , Page No : 3.44.

Q42. List and prove various closure properties of regular sets.

Answer :

For Answer refer to Topic : 3.9 , Page No : 3.44.

OBJECTIVE TYPE QUESTIONS

1. Find the regular expression for the set of all strings over $\{a,b\}$ in which there are atleast two occurrences of b between any two occurrences of a.

   (a) $b*(aa+bb)*a*$ (b) $(aa)*ba(bb)*$

   (c) $b*+(b+abb)*ab*$ (d) None of the above.

2. $(1+00*1)+(1+00*1)(0+10*1)*(0+10*1)=?$

   (a) $(0+10*1)*0*1$ (b) $(1+00*1)(0+10*1)*$

   (c) $0*1(0+10*1)*$ d) None of the above.

3. The empty string is the string with:

   (a) zero occurrence of symbol (b) non zero occurrence of symbol

   (c) no occurrence of symbols (d) None of the above

4. Which of the following regular expressions over {0,1} denotes the set of all string not containing 100 as a substring?

   (a) $0*1010*$ (b) $0*(1*0)*$

   (c) $0*1*01*$ (d) $0*(10+1)*$

5. Find the regular expression for the set of all strings having atmost one pair of 0's or atmost one pair of 1's

   (a) $(1+00)*+(1+01)*(1+10)*+(1+11)*+(0+10)*11(0+10)*$

   (b) $(1+01)*+(1+00)*(1+10)*+(1+10)*+(1+10)*11(0+10)*$

   (c) $(1+01)*+(1+01)*00(1+01)*+(0+10)*+(0+10)*11(0+10)*$

   (d) None of the above.

6. $\in+1*(011)*(1*(011)*)*=?$

   (a) $1*(011)*$ (b) $(1+011)*$

   (c) $1*01*(1+011)*$ (d) None of the above.

7. Find the regular expression for the set of all strings of the form vw where a's occur in pairs in v and b's occur in pairs in w.

   (a) $((aa)*b)((bb)*a)$ (b) $(aabaa)*(bb+a)*$

   (c) $(aa+b)*(bb+a)*$ (d) None of the above.

8. The intersection of $(a+b)^*a$ and $b(a+b)^*$ is given by

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

(c) $(a+b)^*ab(a+b)^*$ d) $b(a+b)^*a$

9. Which one is false

(a) $(r_1+r_2)^* = (r_1^* r_2^*)^*$ (b) $(r^*)^* = r^*$

(c) $r_1^*(r_1+r_2)^* = (r_1+r_2)^*$ (d) none.

10. The set of regular languages over a given alphabet set is not closed under

(a) Intersection (b) union

(c) Complement (d) none

11. Which of the following pairs are equivalent

(a) (a^*+b) and $(a+b)^*$ (b) $(ab)^*a$ and $a(ba)^*$

(c) $(a+b)^*$ and (a^*+b^*) (d) None

12. The language of all words with at least 2 a's can be described as

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

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

(c) $(a+b)^*ab^*(a+b)^*$

(d) all

13. Which of the following pairs are not equivalent

(a) x^+ and x^*x^+ (b) $(ab)^*$ and a^*b^*

(c) $x(xx)^*$ and $(xx)^*x$ (d) $1(01)^*$ and $(10)^*1$

14. Let L may be language. Define even(w) as the string obtained by extracting from w the letters in even numbered positions i.e., if $w = a_1a_2a_3a_4.....$, then even(w) $= a_2a_4.....$ Corresponding to this, we can define a language : even(L) = { even(w) : $w \in L$} then given L is regular, even(L) is

(a) is not context free

(b) context free

(c) must be regular

(d) may not be regular

15. Which is the correct regular expression for the language : "Set of strings over alphabet $\{a,b,c\}$ containing at least 1 'a' and at least 1 'b'

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

(b) $(a+b+c)^* - (a^* + b^* + c^*)$

(c) $(a+b+c)^*[a(a+b+c)^*b + b(a+b+c)^*a](a+b+c)^*$

(d) none of these.

16. Which is the correct order of precedence of regular expression operators in increasing order?

(a) *,( ),+,... (b) ( ),.,*,+ (c) *,( ),....,+ (d) ( ),*,....,+

17. Which of the following is accepted by $L(aa^* + aba^*b^*)$

(a) abab (b) aaab (c) abba (d) None.

18. Let r_1 *and* r_2 are regular expression and let $\cup$ stands for equivalence in the sense of the language generated, then

(a) $r_1(r_1 + r_2)^* \equiv (r_1 + r_2)^*$ (b) $(r_1^2 + r_2)^* \equiv (r_1^* r_2^*)^*$

(c) $(r_1^*)^* \equiv r_1$ (d) None of these

19. Regular expression for the language, $L = \{w \in \{0,1\}^* : w$ has no pair of consecutive zeros$\}$ is

(a) $r = (1+01)^*(0+1^*)$ (b) $r = (1^*011^*)^*(0+A)+1^*(0+A)$

(c) $r = (1+01)^*(0+A)$ (d) all of these

20. For $L(r) = \{a, bb, aa, abb, ba, bbb, ...\}, r$ is given by

(a) $r = (a+b)^*(a+bb)$ (b) $r = (aa+b)(a+b)^*$

(c) $r = (a+bb)^*$ (d) $r = a(a+bb)^*$

21. A language $L = \{awa : w \in \{a,b\}^*\}$ is

(a) context sensitive (b) regular

(c) context free (d) none of these

22. The value of the relation $A + RR^*$ is

(a) R^* (b) ϕ (c) $\in$ (d) R

23. The value of the relation $(R^*)^*$ is

(a) $\in$ (b) R (c) R^* (d) None of the above

24. The value of the relation ϕ^* is

(a) ϕ (b) Σ (c) $\in$ (d) None of the above

25. The value of the relation A^* is

(a) ϕ (b) Σ (c) $\in$ (d) None of the above

26. The value of the relation $R \in$ is

(a) ϕ (b) R (c) $\in$ (d) None of the above

27. The value of the relation $\in R$ is

(a) ϕ (b) R (c) $\in$ (d) None of the above

28. The value of the relation $R\phi$ is

(a) ϕ (b) R (c) $\in$ (d) None of the above

29. The value of the relation ϕR is

(a) ϕ (b) R (c) $\in$ (d) None of the above

30. The value of the relation $\phi + R$ is

(a) ϕ (b) R (c) $\in$ (d) None of the above

31. Which of the following identities for regular expression does not hold good?

(a) $(R+S)^* = R^* + S^*$ (b) $(R^*S^*)^* = (R+S)^*$

(c) $(\in + R^*) = R^*$ (d) $(R^*)^* = R^*$

32. ϕ^* (Kleene's closure of ϕ) (ϕ is the empty language over Σ) is equivalent to

(a) Σ (b) ϕ (c) $\in$ (d) none of these.

33. Give English description of the language of the regular expression : $(1+\in)(00^*1)^*0^*$

(a) alternating 1's and 0's (b) 0's only in pairs

(c) no pair of consecutive 1's (d) set of all strings of 0's and 1's containing

34. Let L be the language $\{\in,0,10\}$ over $\{0,1\}$. Determine the set $L \cup \overline{L}$.

(a) $\{0,1\}^*$ (b) ϕ

(c) same as the given set (d) None of the above

35. The regular expression representing the set of all strings over $\{x, y\}$ ending with xx beginning with y

(a) $x(x+y)^* yy$ (b) $y(x+y)^* xx$ (c) $yy(x+y)^* x$ (d) $xx(x+y)^* y$

36. How many strings of length t are in Σ^* if Σ is an alphabet of cardinality r.

(a) r + t (b) tr (c) rt (d) None of the above

37. Which of the following doesn't hold?

(a) $\in +1^*(011)^*(1^*(011)^*)^* = \in +1^*(0111^*)$

(b) $(111^*)^* = (11+111)^*$

(c) $(1+0)^* = 1^*(01^*)^*$

(d) $(1+00^*1)+(1+00^*1)(0+10^*1)^*(0+10^*1) = 0^*1(0+10^*1)^*$

38. A solution to the equation $R = Q + RP$ is

(a) $R = PQ^*$ (b) $P = RQ^*$ (c) $Q = RP^*$ (d) $R = QP^*$

39. The value of the relation $(P^* + Q^*)^*$ is

(a) $(P^*Q^*)^*$ (b) Σ^* (c) P^*Q^* (d) None of the above

40. The value of the relation $(P+Q)^*$ is

(a) $P^* + Q^*$ (b) $(P^*Q^*)^*$ (c) P^*Q^* (d) Σ^*

41. The value of the relation $R + R$ is

(a) R^* (b) ϕ (c) $\in$ (d) R

42. The value of the relation $RR^* + \in$ is

(a) R^* (b) ϕ (c) $\in$ (d) R

43. In English language the set represented by a^*b+b^*a is:

(a) Strings of a's followed by one b and strings of b's followed by one a.

(b) String containing single a and single b

(c) Strings of a's followed by one b or strings of b's followed by one a.

(d) String containing single a or single b

44. The regular expression representing the set of all strings over $\{a,b\}$ with three consecutive b's

(a) $(a+b)^*bbb(a+b)$ (b) $(a+b)^*bb(a+b)^*$

(c) $(a+b)bbb(a+b)^*$ (d) $(a+b)^*bbb(a+b)^*$

45. For the following conditions find all the strings over the alphabet {a, } that satisfy the condition, (i) no symbol is repeated in the string and (ii) the length of string is 3.

(a) No such string is possible (b) All possible strings of length 3

(c) Only single string ab (d) None of the above.

46. Find the true statement

(a) If R is regular expression then so is R^*

(b) If R and S are regular expression then so is $R \cup S$

(c) If R and S are regular expression then so is R.S

(d) All of the above.

47. Find the true statement

(a) ϕ represents empty word, $\in$ represents empty language.

(b) $\in$ represents empty word, ϕ represents empty language

(c) $\in, \phi$ represents empty word

(d) $\in, \phi$ represents empty language

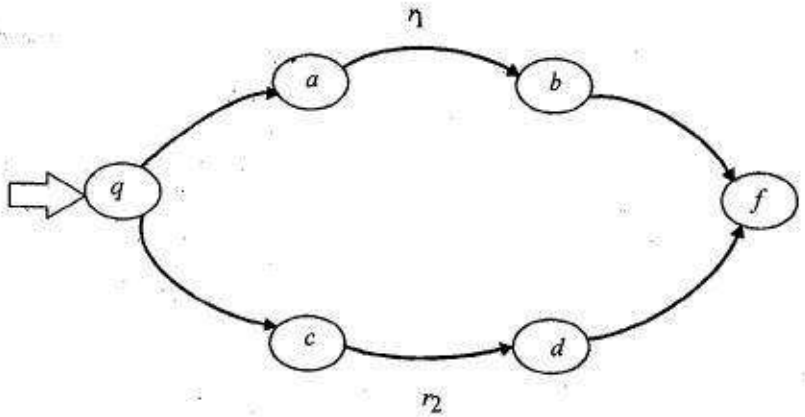
48. Regular expression for the set $\{a^2, a^5, a^8, ...\}$ is :

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

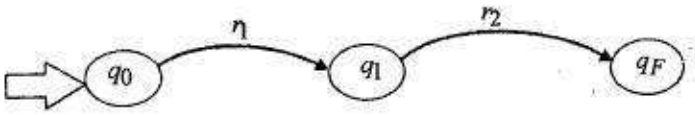
(c) $aa(aaaa)^*$ (d) $aa(aaa)^*$

49. Let r_1 *and* r_2 are regular expression which of the following represent $r_1 + r_2$

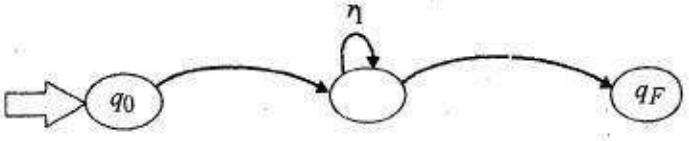
(a)



(b)



(c)



(d) none.

50. $01^* + 0$ is represented by

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

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

(c) $((01)^* + 0)$

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

51. Which of the following identities doesn't hold?

(a) $R^*.R^* = R^*$

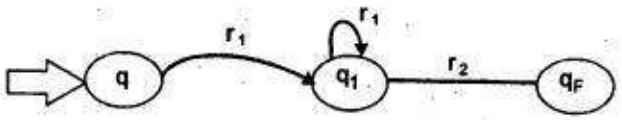
b) $(R \cup S)^* = (R^*.S^*)^*$

(c) $(R^* \cup S^*)^* = (R.S)^*$

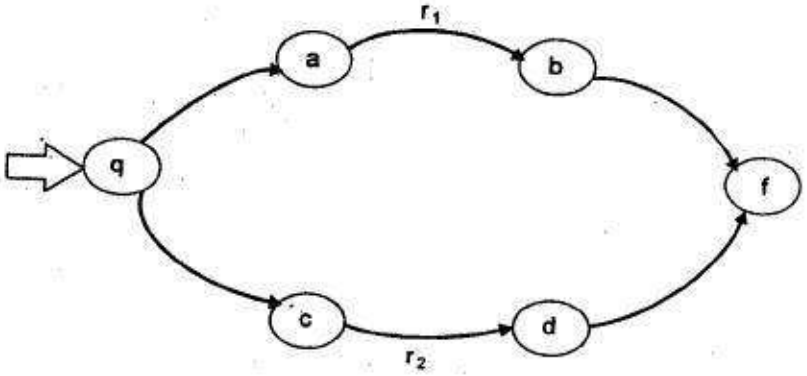
d) $(R \cup S)^* = (R^* \cup S^*)^*$

52. Let r_1 *and* r_2 are regular expression which of the following represent r_1

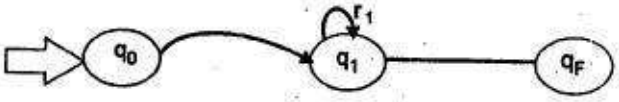
(a)



(b)



(c)



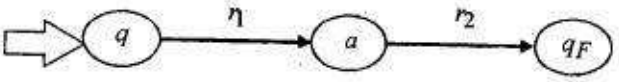
(d) none

53. Which of the following is false

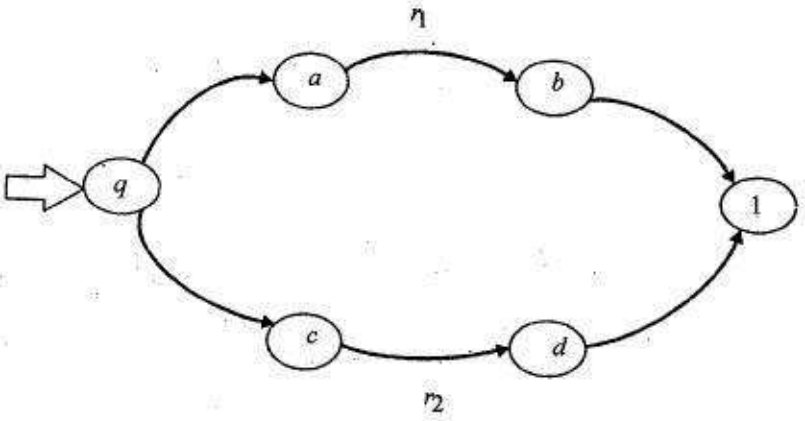
(a) $RR^* = R^*R$ (b) $R + R = R$ (c) $(R^*)^* = R^*$ (d) none.

54. Let r_1 *and* r_2 are regular expression which of the following represent $r_1.r_2$.

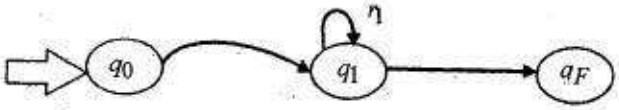
(a)



(b)



(c)



(d) none

55. Which of the following are correct

(a) If L^* is regular then L is regular

(b) If $L_1 \cup L_2$ is regular and L_1 is regular, then L_2 is regular,

(c) If L_1L_2 is regular and L_1 is regular, then L_2 is regular.

(d) all

56. Which of the following is set of strings of the form vw where a's occur in pairs in v and b's occur in pairs in w.

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

(c) $a^*b + b^*a$ (d) $a(a + b)^*ab$

57. The set of all strings which are either strings of a's followed by one b or strings of b's followed by one a.

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

(c) $a^*b + b^*a$ (d) $a(a + b)^*ab$

58. Select which of following represent a set of all strings with a and ending with ab.

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

(c) $a^*b + b^*a$ (d) $a(a + b)^*b$

59. $(a^*ab+ba)^*a^*$ is equivalent to

(a) $(a+b+ab)^*$ (b) $(aba+bab)^*$

(c) $(a+ab+ba)^*$ (d) $(ab+ba+aba)$

60. The two regular expressions are equivalent i.e., $\in +(a+b)^*b(a+b)^* = [a^*b(a^*ba^*b)^*a^*]^*$

(a) True (b) False.

61. The set all strings of 0's and 1's such that every pair of adjacent 0's appears before any pair of adjacent 1's

(a) $(10+0)^*(epsilon+1)(01+01)^*(epsilon+0)$

(b) $(10+0)^*(epsilon+1)(01+1)^*(epsilon+0)$

(c) $(10+0)^*(epsilon+1)^*(epsilon+0)$

(d) $(100)^*(epsilon+1)(01+1)^*(epsilon+0)$

62. Write the regular expression for the following:

"The set of the strings over alphabet $\{a,b,c\}$ containing at least one a and at least one b"

(a) $ca^*(a+c)^*b(a+b+c)^*c^*b(b+c)^*a(a+b+c)^*$

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

(c) $ca^*(a+c)^*b(a+b+c)^*c^*b(b+c)^*a(a+b+c)^*$

(d) $c^*a(a+c)^*b(a+b+c)^*+c^*b(b+c)^*a(a+b+c)^*$

63. The reversal of the language $L=\{001,10,111\}$ *is* :

(a) {111,01,110} (b) {100,01,111}

(c) {111,10,001} (d) none

64. $(L^*)^*$ equal to :

(c) (L^{**}) (a) L^* (b) L^{**} (d) none

65. The language generated by the regular expression $(aa)^*(bb)^*b$ is

(b) $a^{2n}b^{2n+1}$ (a) $(ab)^{2n}b$ (c) none of these.

66. $(1^*011^*)^*(0+\in)$ is equivalent to

(b) $(0+1)(1+10)^*$ (a) $(1+01)^*(0+\in)$ (c) none of these

67. Which of the following identity doesn't hold?

(a) $\phi + R = R + \phi = R$ (b) $\phi R = R\phi = \phi$

(c) $\in + R = R + \in = R$ (d) $\in R = R \in = R$

68. The language of all words that have at least one a and at least one b is

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

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

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

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

69. Let a and b be two regular expressions then $(a^* \cup b^*)^*$ is equivalent to

(a) $a \cup b$ (b) $(b \cup a)^*$ (c) $(b^* \cup a^*)^*$ (d) $(a \cup b)^*$

70. If e_1 *and* e_2 are regular expressions denoting the languages L_1 and L_2 respectively, then which is false?

(a) $(e_1)^*$ is a regular expression denoting L_1^*

(b) ϕ is not a regular expression

(c) $(e_1)(e_2)$ is a regular expression denoting L_1L_2

(d) $(e_1)(e_2)$ is a regular expression denoting $L_1 \cup L_2$

71. What is the regular expression defining the language of all words with an odd number of b's is

(a) $a^*b(a^*ba^*b)^*a^*$

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

(c) $a^*(a^*b)^*a^*$

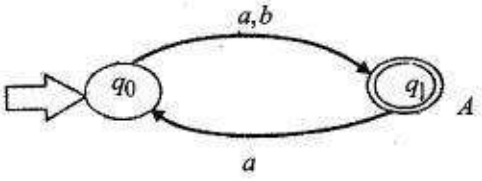
(d) None of these.

72. $(1+0)^*$ represents
(a) Set of strings over 1 and 0.
(b) Set of strings starting with 1 and ending with 0
(c) Set of strings with equal number of 1's and 0's
(d) Set of strings with even number of 1's and 0's

73. Which one is TRUE
(a) $\{1010\} belongs\ to\ (10)^*$
(b) $(10)^* = 1^* + 0^*$
(c) $(10)^* = (1^*0^*)^*$
(d) $(10)^* = 1^*0^*$

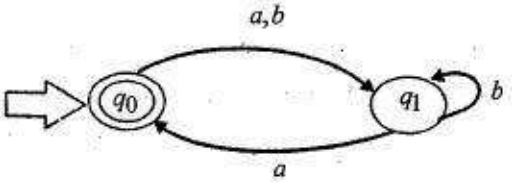
74. The $R.E. = (10+01+11+00)^*$ represents
(a) set of strings with at least one 0 and at least one 1
(b) set of strings with even length
(c) set of strings with equal 0 and 1's
(d) All strings over 0 and 1

75. Regular expression generated by the following automaton is given as



(a) $(a+b)(ab+aa)^*$
(b) $(a+b)(ab+aa)^*a$
(c) $\in +(a+b)(ab+aa)^*a$
(d) $\in +(a+b)(ab+aa)^*$

76. Regular expression generated by the following automaton is given as:



(a) $(a+b)(b+ab+aa)^*a$
(b) $\in +(a+b)(b+ab+aa)a$
(c) $(a+b)(b+ab+aa)^*$
(d) $\in +(a+b)(b+ab+aa)^*a$

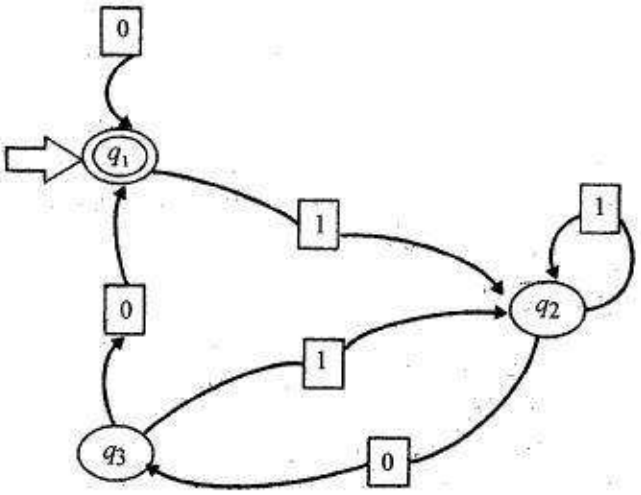
77. Regular expression corresponding to the finite automaton drawn below is given by

(d) $(0+0(1+10)^*00)^*$

(b) $(0+0(1+01)^*00)^*$

(c) $(0+1(0+10)^*00)^*$

(a) $(0+1(1+01)^*00)^*$



78. The regular expression $(1+00^*1)+(1+00^*1)(0+0+10^*1)^*(0+10^*1)$ is equivalent to

(a) $(1+00^*1)(0^*(10^*1)^*)^*$ (b) $0^*1(0+10^*1)^*$

(c) $(1+00^*1)(0+10^*1)^*$ (d) All of the above.

79. Which of the following is regular?

(a) String of odd number of zeroes.

(b) Strings of 0's, whose length is a prime number

(c) String of all palindromes made up of 0's and 1's

(d) String of 0's whose length is a perfect square

80. The recognizing capability of NDFA and DFA

(a) must be same (b) may be different

(c) must be different (d) none of the above.

81. The intersection of the two regular languages below:

$L_1 = (a+b)^*a$ and $L_2 = b(a+b)^*$ is given by

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

(c) $(a+b)^*ab(a+b)^*$ (d) $b(a+b)^*a$

82. Which of the following regular expression over {0,1} denotes the set of all string not containing 100 as a substring?

(a) $0^*(10+1)^*$ (b) 0^*101^* (c) 0^*1010^* (d) $0^*(1^*0)^*$

83. The set all strings over $\{a,b\}$ in which there are atleast two occurrences of b between the two occurrences of a.

(a) $a(bbabb)^* a$ (b) $b^*(b+ab)^* b^*$

(c) $b^* + (b+abb)^* ab^*$ (d) None of the above.

84. Set of all strings over $\{0,1\}$ having atmost one pair of 0's or atmost one pair of 1's.

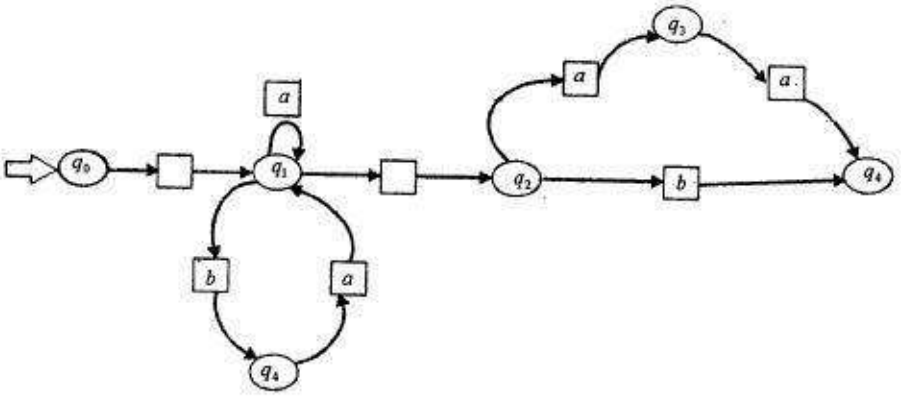
(a) $(1^* + (01)^*)^* + (1^*)(01)^* 00(1^*(01)^*)^* + (0^*)(10)^* + ((0^*) + (10)^*)^* 11(0+10)^*$

(b) $(1+01)^* + (1+01)^* 00(1+01)^* + (0+10)^* + (0+10)^* 11(0+10)^*$

(c) $(1+01)^* + (0+10)^* 00(0+10)^* + (0+10)^* + (1+01)^* 11(1+01)^*$

(d) None of the above.

85. Regular expression corresponding to the FA given below is



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

(c) $(a^* b + b^* a)^*$ (d) None of the above.

86. Which of the following closure properties hold for regular sets?

(i) If L is regular, then L^T is also regular.

(ii) If L is regular set over Σ, then $\Sigma^* - L$ is also regular over $\cap$.

(iii) If X and Y are regular sets over Σ, then X intersection Y is also regular over Σ

(a) Only (i), (ii) and (iii). (b) Only (i) and (iii)

(c) Only (ii) (d) Only (i)

87. If L is an infinite regular language, then there exist some positive integer m such that any string $w \in L$ whose length is m or greater can be decomposed into three part, xyz, where

(a) $w = xy^i z$ is also in L for all i = 0,1,2,3,. (b) $|xy|$ *is less than or equal to m.*

(c) $|y| > 0$.. (d) All of the above.

88. If L is the language $L(01^*2)$, what is h(L):

(a) aba^*ab (b) $aab(ba)^*$ (c) aab^*ba (d) $a(ab)^*ba$

89. The inverse homomorphism of a regular language is :

(a) not regular (b) regular (c) none

A homomorphism is a function from some alphabet Σ_1 to strings in another alphabet Σ_2. If $x = a_1a_2,....a_k \in \Sigma_1^*$, *then* $h(x) = h(a_1)h(a_2)....h(a_k)$, *and if* $L \subseteq \Sigma^*$, then $h(L) = \{h(x) / x \in L\}$.

90. Suppose h is the homomorphism from the alphabet {0,1,2} to the alphabet {a,b} defined by: $h(0) = a$; $h(1) = ab$, *and* $h(2) = ba$ .What is $h(0120)$?

(a) ababa (b) abbbb (c) aaabb (d) aabba

91. If L is regular, then {x: reverse (x) in L} is also regular

(a) May or may not be (b) Yes

(c) No (d) None of the above.

92. Finite state machines..... can recognize palindromes

(a) may not (b) may (c) can't (d) can

93. Pick the correct statement. The logic of Pumping lemma is a good example of

(a) Iteration (b) Recursion

(c) The divide and conquer technique (d) The Pigeon hole principle

94. Which of the following is not regular

(a) String of zero whose length is prime

(b) String of zero whose length is perfect square

(c) Set of palindromes over 0 and 1

(d) All

95. Pumping lemma can be used

(a) Whether two languages are equivalent

(b) To check whether a language is regular

(c) To check whether a language is irregular

(d) None.

96. Let L be a regular language defined are Σ^*. Then

(a) index (R_L) may be zero (b) index (R_L) is finite

(c) index (R_L) may be infinite (d) None.

97. Let $\Sigma = \{0,1\}$ and R be the relation defined on Σ^* by $(x, y) \in R$ iff $|x| - |y| = odd$ Then R is

(a) not a right congruence (b) an equivalence relation

(c) a right congruences (d) none.

98. Let $\Sigma = \{a\}$ and let I be the identify relation on Σ^*. Let $L = \{\in\} \cup \{a\} \cup \{aa\}$. Then index (I) is

(a) 3 (b) finite (c) infinite (d) None.

99. Is there a finite automation which accepts all palindromes over {a,b}?

(a) No,but it cannot be proved. (b) No, it can be proved.

(c) Yes, but it cannot be proved (d) Yes,it can be proved

100. Which of the following sets is regular?

(a) $\{a^{2n} \mid n \geq 1\}$ (b) $\{ww \mid w \in \{a,b\}^*\}$

(c) $\{a^p \mid p \text{ is a prime}\}$ (d) $\{a^{i^2} \mid i \geq 1\}$

101. Which of the following languages cannot be produced by a regular grammar?

(i) $\{a^n b^n : n \geq 0\}$

(ii) $\{a^m b^k : k > n \geq 0\}$

(iii) $\{ww^R : w \in \{a,b\}^*\}$

(a) (ii) and (iii) (b) (i)

(c) (i) and (ii) (d) All of the above.

ANSWER KEY

1(c)	2 (b)	3 (a)	4 (d)	5 (c)	6 (a)	7 (c)	8 (d)	9 (a)
10 (d)	11 (c)	12 (b)	13 (b)	14 (c)	15 (c)	16(b)	17(a)	18(b)
19.(d)	20.(a)	21.(a)	22.(a)	23.(c)				
24.(c)	25.(c)	26.(b)	27.(b)	28.(a)	29.(a)	30.(b)	31.(a)	
32(c)	33.(a)	34.(b)	35.(b)	36.(c)	37.(a)	38.(d)	39.(a)	
40.(b)	41.(a)	42.(a)	43.(c)	44.(d)	45.(a)	46.(d)	47.(b)	
48.(d)	49.(a)	50.(d)	51.(c)	52.(d)	53.(d)	54.(a)	55.(c)	
56.(b)	57.(c)	58.(d)	59.(c)	60.(a)	61.(b)	62.(d)	63.(b)	
64.(b)	65.(a)	66.(b)	67.(c)	68.(a&b)	69.(d)	70.(b)		
71.(a)	72.(a)	73.(a)	74.(b)	75.(a)	76.(d)	77.(d)	78.(b)	
79.(a)	80.(a)	81.(d)	82.(a)	83.(c)	84.(b)	85.(b)	86.(a)	
87.(d)	88.(d)	89.(b)	90.(d)	91.(b)	92.(c)	93.(d)	94.(d)	
95.(c)	96.(b)	97.(a)	98.(c)	99.(b)	100.(a)	101.(d)		

4

REGULAR GRAMMARS

After going through this chapter, you should be able to understand :

- Regular Grammar
- Equivalence between Regular Grammar and FA
- Interconversion

4.1 REGULAR GRAMMAR

Definition : The grammar G = (V, T, P, S) is said to be regular grammar iff the grammar is right linear or left linear.

A grammar G is said to be right linear if all the productions are of the form

$A \rightarrow wB$ and / or $A \rightarrow w$ where $A, B \in V$ and $w \in T^*$.

A grammar G is said to be left linear if all the productions are of the form

$A \rightarrow Bw$ and / or $A \rightarrow w$ where $A, B \in V$ and $w \in T^*$.

Example 1 : The grammar

$S \rightarrow aaB \mid bbA \mid \in$

$A \rightarrow aA \mid b$

$B \rightarrow bB \mid a \mid \in$

is a right linear grammar. Note that $\in$ and string of terminals can appear on RHS of any production and if non - terminal is present on R. H. S of any production, only one non - terminal should be present and it has to be the right most symbol on R. H. S.

Example 2 :

The grammar

$S \rightarrow Baa \mid Abb \mid \in$

$A \rightarrow Aa \mid b$

$B \rightarrow Bb \mid a \mid \in$

is a left linear grammar. Note that $\in$ and string of terminals can appear on RHS of any production and if non - terminal is present on L. H. S of any production, only one non - terminal should be present and it has to be the left most symbol on L. H. S.

Example 3 :

Consider the grammar

S → aA

A → aB | b

B → Ab | a

In this grammar, each production is either left linear or right linear. But, the grammar is not either left linear or right linear. Such type of grammar is called linear grammar. So, a grammar which has at most one non terminal on the right side of any production without restriction on the position of this non - terminal (note the non - terminal can be leftmost or right most) is called linear grammar.

Note that the language generated from the regular grammar is called regular language. So, there should be some relation between the regular grammar and the FA, since, the language accepted by FA is also regular language. So, we can construct a finite automaton given a regular grammar.

4.2 FA FROM REGULAR GRAMMAR

Theorem : Let G = (V, T, P, S) be a right linear grammar. Then there exists a language L(G) which is accepted by a FA. i. e., the language generated from the regular grammar is regular language.

Proof : Let $V = (q_0, q_1, ...)$ be the variables and the start state $S = q_0$ Let the productions in the grammar be

$$q_0 \rightarrow x_1 q_1$$

$$q_1 \rightarrow x_2 q_2$$

$$q_3 \rightarrow x_3 q_3$$

.

.

.

$$q_n \rightarrow x_n q_n$$

Assume that the language L(G) generated from these productions is w. Corresponding to each production in the grammar we can have a equivalent transitions in the FA to accept the string w. After accepting the string w, the FA will be in the final state. The procedure to obtain FA from these productions is given below :

Step 1 : q_0 which is the start symbol in the grammar is the start state of FA.

Step 2 : For each production of the form

$$q_i \rightarrow wq_j$$

the corresponding transition defined will be

$$\delta^*(q_i, w) = q_j;$$

Step 3 : For each production of the form $q_i \rightarrow w$

the corresponding transition defined will be $\delta^*(q_i, w) = q_f$, where q_f is the final state,

As the string $w \in L(G)$ is also accepted by FA, by applying the transitions obtained from step1 through step3, the language is regular. So, the theorem is proved.

Example 1 : Construct a DFA to accept the language generated by the following grammar

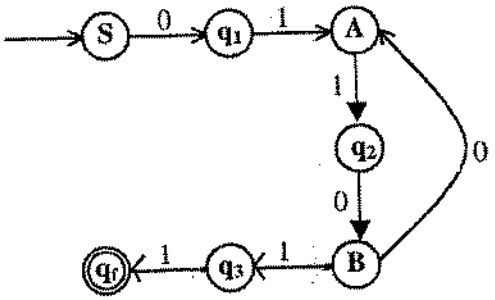
$$S \rightarrow 01A$$
$$A \rightarrow 10B$$
$$B \rightarrow 0A \mid 11$$

Solution :

Note that for each production of the form A → wB, the corresponding transition will be $\delta(A, w) = B$. Also, for each production $A \rightarrow w$, we can introduce the transition $\delta(A, w) = q_f$ where q_f is the final state. The transitions obtained from grammar G is shown using the following table :

Productions			Transitions
S	→	01A	$\delta(S, 01) = A$
A	→	10B	$\delta(A, 10) = B$
B	→	0A	$\delta(B, 0) = A$
B	→	11	$\delta(B, 11) = q_f$

The FA corresponding to the transitions obtained is shown below :



So, the DFA $M = (Q, \Sigma, \delta, q_0, A)$ where

$Q = \{ S, A, B, q_f, q_1, q_2, q_3 \}$, $\Sigma = \{0,1\}$

$q_0 = S$, $A = \{q_f\}$

δ is as obtained from the above table.

The additional vertices introduced are q_1, q_2, q_3.

Example 2 : Construct a DFA to accept the language generated by the following grammar .

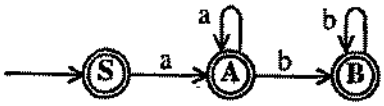
S → **aA** | ∈
A → **aA| bB** | ∈
B → **bB** | ∈

Solution :

Note that for each production of the form $A \rightarrow wB$, the corresponding transition will be $\delta(A,w) = B$. Also, for each production $A \rightarrow w$, we can introduce the transition $\delta(A,w) = q_f$ where q_f is the final state. The transitions obtained from grammar G is shown using the following table :

Productions			Transitions
S	→	aA	$\delta(S,a) = A$
S	→	∈	S is the final state
A	→	aA	$\delta(A,a) = A$
A	→	bB	$\delta(A,b) = B$
A	→	∈	A is the final state
B	→	bB	$\delta(B,b) = B$
B	→	∈	B is the final state.

Note : For each transition of the form $A \rightarrow \in$, make A as the final state.
The FA corresponding to the transitions obtained is shown below :



So, the DFA $M = (Q, \Sigma, \delta, q_0, A)$ where

$Q = \{S, A, B\}$, $\Sigma = \{a, b\}$

$q_0 = S$, $A = \{S, A, B\}$

δ is as obtained from the above table .

4.3 REGULAR GRAMMAR FROM FA

Theorem : Let $M = (Q, \Sigma, \delta, q_0, A)$ be a finite automaton. If L is the regular language accepted by FA, then there exists a right linear grammar G = (V, T, P, S) so that L = L(G).

Proof : Let $M = (Q, \Sigma, \delta, q_0, A)$ be a finite automata accepting L where

$Q = \{q_0, q_1, \ldots q_n\}$

$\Sigma = \{a_1, a_2, \ldots a_m\}$

A regular grammar G = (V, T, P, S) can be constructed where

$V = \{q_0, q_1, \ldots q_n\}$

$T = \Sigma$

$S = q_0$

The productions P from the transitions can be obtained as shown below :

Step 1 : For each transition of the form $\delta(q_i, a) = q_j$

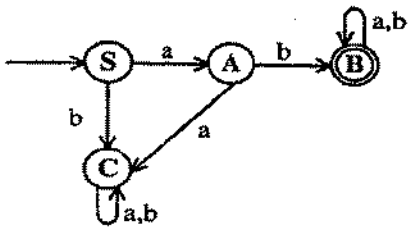
the corresponding production defined will be $q_i \rightarrow aq_j$

Step 2 : If $q \in A$ i. e., if q is the final state in FA, then introduce the production

$q \rightarrow \in$

As these productions are obtained from the transitions defined for FA, the language accepted by FA is also accepted by the grammar.

Example 1 : Construct a regular grammar from the following FA.



Solution : Note that for each transition of the form $\delta(A, a) = B$, introduce the production $A \rightarrow aB$. If $q \in A$ i. e., if q is the final state, introduce the production $A \rightarrow \in$. The productions obtained from the transitions defined for FA is shown using the following table :

Transitions	Productions
$\delta(S, a) = A$	S → aA
$\delta(S, b) = C$	S → bC
$\delta(A, a) = C$	A → aC
$\delta(A, b) = B$	A → bB
$\delta(B, a) = B$	B → aB
$\delta(B, b) = B$	B → bB
$\delta(C, a) = C$	C → aC
$\delta(C, b) = C$	C → bC

It is very important to note that B is the final state. So, we have to introduce the production $B \rightarrow \in$. The grammar G corresponding to the productions obtained is shown below :

Grammar G = (V, T, P, S) where

V = { S, A, B, C }

T = { a, b }

P = {

$S \rightarrow aA \mid bC$

$A \rightarrow aC \mid bB$

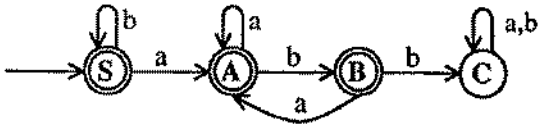
$B \rightarrow aB \mid bB \mid \in$

$C \rightarrow aC \mid bC$

}

S is the start symbol

Example 2 : Construct a regular grammar for the following FA.



Solution : Note that for each transition of the form $\delta(A, a) = B$, introduce the production $A \rightarrow aB$. If $q \in A$ i.e., if q is the final state, introduce the production $A \rightarrow \in$. The productions obtained from the transitions defined for FA is shown using the following table :

Transitions	Productions
$\delta(S, a) = A$	S → aA
$\delta(S,b)=S$	S → bS
$\delta(A,a)=A$	A → aA
$\delta(A, b) = B$	A → bB
$\delta(B,a)=A$	B → aA
$\delta(B,b)=C$	B → bC
$\delta(C, a) = C$	C → aC
$\delta(C, b) = C$	C → bC

It is very important to note that S, A and B are final states. So, we have to introduce the productions $S \rightarrow \in$, $A \rightarrow \in$ and $B \rightarrow \in$. The grammar G corresponding to the productions obtained is shown below :

Grammar G = (V, T, P, S) where

V = { S, A, B, C }

T = { a, b }

P = {
S → aA | bS | ∈
A → aA | bB | ∈
B → aA | bC | ∈
C → aC | bC
}

S is the start symbol

Note : The FA in this problem accepts strings of a's and b's except those containing the substring abb. So, from the grammar G we can obtain a regular language which consists of strings of a's and b's without the substring abb.