Example 4.13. Construct the ϵ-NFA for the regular expression $(0 + 1)^* 1 (0 + 1)$.

Solution. Here automation for 0 and 1 is basic automation. First we will make automation for $(0 + 1)$ as in Fig. 4.9 (a).

Now we will convert it into automation for $(0 + 1)^*$ as shown in Fig. 4.9(b).

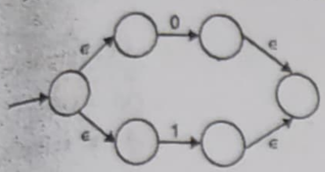

Fig. 4.9(a). Automation for $(0 + 1)$.

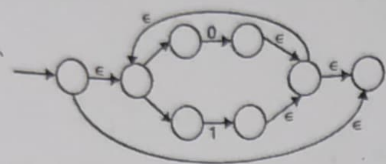

Fig. 4.9(b). Automation for $(0 + 1)^*$.

Now we will construct automation for $1 . (0 + 1)$ as shown in Fig. 4.9(c).

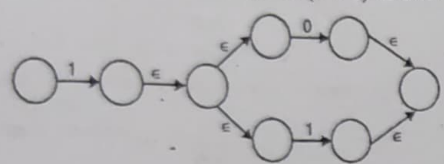

Fig. 4.9 (c). Automation for $1 . (0 + 1)$.

Now we have to connect ϵ-NFA of Fig. 4.9(b) and ϵ-NFA of Fig. 4.9 by a ϵ-transition to find out the ϵ-NFA for given regular expression as shown in Fig. 4.9(d).

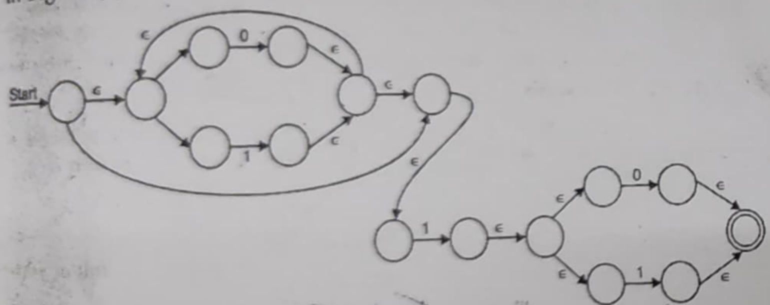

Fig. 4.9 (d). Automation for $(0 + 1)^* 1(0 + 1)$.

4.4. Construction of Regular Expression from DFA

Now it is clear that regular expression is mathematical expression for a given regular language. We have already know that for every regular language there exist a deterministic finite automate. We can conclude that regular expression, regular language for that regular expression and deterministic finite automata for that regular language are similar things in different representation. So we can construct regular expression for every deterministic automata.

4.4.1. Ardens's Theorem

Let P and Q be two regular expression over alphabet Σ. If P does not contain null string ϵ, then

$$R = Q + RP$$

has a unique solution that is $R = QP^*$

It can be understand as : $R = Q + RP$

Put the value of R in R.H.S.

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

When we put the value of R again and again we got the following equation.

$$R = Q + QP + QP^2 + QP^3 \ldots$$
$$R = Q(1 + P + P^2 + P^3 \ldots$$
$$R = Q(\epsilon + P + P^2 + P^3 + \ldots$$
$$R = QP^*$$

(By the definition of closure operation for regular expression.)

4.4.2. Use of Arden's Theorem to find Regular Expression of a Deterministic Finite Automata

There are certain assumptions which are made regarding the transition system :

(i) The transition diagram should not have ϵ-transitions.

(ii) It must have only a single initial state.

(iii) It vertices are $q_1 \ldots q_n$.

(iv) q_i is final state.

(v) w_{ij} denotes the regular expression representing the set of labels of edges from q_i to q_j. We can get the following set of equation in $q_1 \ldots q_n$.

$$q_1 = q_1 w_{11} + q_2 w_{21} + \ldots + q_n w_{n1} + \epsilon \quad \ldots(1)$$

(since q_1 is the initial state)

$$q_2 = q_1 w_{12} + q_2 w_{22} + \ldots q_n w_{n2} \quad \ldots(2)$$
$$\vdots \qquad \vdots$$
$$q_n = q_1 w_{1n} + q_2 w_{2n} + \ldots + q_n w_{nn} \quad \ldots(n)$$

We solve these equation for q_i in terms of w_{ij}'s and it will be required regular expression. One thing should be noted that we add ϵ (null string) in the equation starts with starting state q_1 and we solve equation to find out q_i (final state) in terms of w_{ij}'s, it is one of the regular expression for given deterministic finite automata.

Example 4.14. Find the regular expression for transition diagram given in Fig. 4.11.

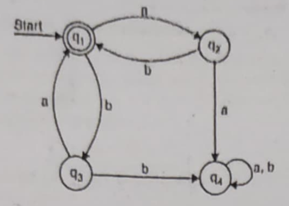

Fig. 4.11.

Solution. Now let us from the equations :

$$q_1 = q_2b + q_3a + \epsilon.$$
$$q_2 = q_1a$$
$$q_3 = q_1b$$
$$q_4 = q_2a + q_3b + q_4a + q_4b$$

Put q_2 and q_3 in q_1 as

$$q_1 = q_1ab + q_1ba + \epsilon$$
$$q_1 = \epsilon + q_1(ab + ba)$$
$$q_1 = \epsilon(ab + ba)^*$$

So required regular expression is $(ab + ba)^*$.

Example 4.15. Construct a regular expression corresponding to the state diagram given in Fig. 4.12.

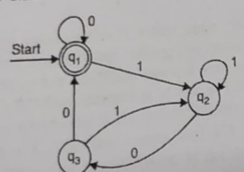

Fig. 4.12.

Solution. Let us form the equations :

$$q_1 = q_10 + q_30 + \epsilon$$
$$q_2 = q_11 + q_21 + q_31$$
$$q_3 = q_20$$
$$q_2 = q_11 + q_21 + (q_20)\,1 = q_11 + q_2(1 + 01)$$
$$q_2 = q_1\,1\,(1 + 01)^*$$

So

$$q_1 = q_10 + q_30 + \epsilon = q_10 + q_200 + \epsilon = q_10 + (q_11\,(1+01)^*)00 + \epsilon$$
$$q_1 = q_1(0 + 1(1 + 01)^*00) + \epsilon$$
$$q_1 = \epsilon(0 + 1(1 + 01)^*00)^*$$
$$q_1 = (0 + 1(1 + 01)^*\,00)^*$$

Scanned with CamScanner

So regular expression is $(0 + 1(1 + 01)^*\,00)^*$.

- **Example 4.16. Find the regular expression corresponding to Fig. 4.13.**

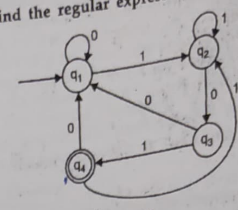

Fig. 4.13.

Solution. Now let us from the equations

$$q_1 = q_10 + q_30 + q_40 + \epsilon \quad \text{— } 1$$
$$q_2 = q_11 + q_21 + q_41 \quad \text{— } 2$$
$$q_3 = q_20 \quad \text{— } 3$$
$$q_4 = q_31 \quad \text{— } 4$$

4 ⇒ Now $q_4 = q_31 = q_201$

2 ⇒ Using q_2 equation, we get $q_2 = q_11 + q_21 + q_2011 = q_11 + q_2(1 + 011)$

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

Put q_3 and q_4 in q_1 equation

$$q_1 = q_10 + q_200 + q_2010 + \epsilon$$
$$= q_10 + q_2(00 + 010) + \epsilon$$
$$= q_10 + q_11(1 + 011)^*\,(00 + 010) + \epsilon$$
$$q_1 = \epsilon(0 + 1(1 + 011)^*\,(00 + 010))^*$$

Put q_2 in q_4 equation

$$q_4 = q_11(1 + 011)^*\,01 = q_1(1(1 + 011)^*01)$$
$$q_4 = (0 + 1(1 + 011)^*\,(00 + 010))^*\,(1(1 + 011)^*01)$$

So required regular expression is

$$(0 + 1(1 + 011)^*\,(00 + 010))^*\,(1(1 + 011)^*01)$$

4.5 Algebraic Laws for Regular Expressions

In this section, we will see a collection of algebraic laws that bring the issue of when two regular laws are equivalent. Instead of examining specific regular expressions, we shall consider pairs of regular expressions with variables as arguments.

> **TIPS**
>
> Two expressions with variables are equivalent if whatever languages we substitute for the variables, the result of the two expressions are the same language.

Like arithmetic expressions, the regular expressions have a number of laws that work for them. Many of these are similar to the laws for arithmetic, if we think of union as addition and concatenation as multiplication. However, there are a few places where the analogy breaks down, and there are also some laws that apply to regular expressions but have no analog for arithmetic.

Scanned with CamScanner