(*iii*) The NFA without null move for the $RE 10 + (0 + 11)0 * 1$ is shown in Fig. 5.30.

The equivalent DFA for the previous NFA is given in Fig. 5.31.

There are four states of the DFA for the RE $10 + (0 + 11)0^*1$. So, for the grammar of the RE, there will be four non-terminals. Let us take them as A (for Q_0), B (for Q_1), C (for Q_2), and D (for Q_3).

(*iii*) The NFA without null move for the $RE 10 + (0 + 11)0 * 1$ is shown in Fig. 5.30.

The equivalent DFA for the previous NFA is given in Fig. 5.31.

There are four states of the DFA for the RE 10 + (0 + 11)0*1. So, for the grammar of the RE, there will be four non-terminals. Let us take them as A (for Q_0), B (for Q_1), C (for Q_2), and D (for Q_3).

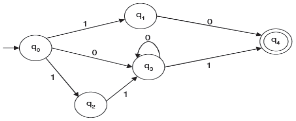


Fig. 5.30

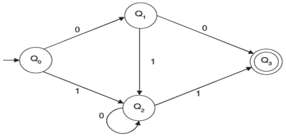


Fig. 5.31

Now, we have to construct the production rules of the grammar.

For the transitional function $\delta(Q_0, 0) \rightarrow Q_1$, the production rule will be $A \rightarrow 0B$.

Similar to $\delta(Q_0, 1) \rightarrow Q_2$, the production rule will be $A \rightarrow 1C$.

$\delta(Q_1, 0) \rightarrow Q_3$, the production rule will be $B \rightarrow 0D$ and $B \rightarrow 0$ (as Q_3 is the final state)

$\delta(Q_1, 1) \rightarrow Q_2$, the production rule will be $B \rightarrow 1C$

$\delta(Q_2, 0) \rightarrow Q_2$, the production rule will be $C \rightarrow 0C$

$\delta(Q_2, 1) \rightarrow Q_3$, the production rule will be $C \rightarrow 1D$ and $C \rightarrow 1$ (As Q_3 is the final state).

The start symbol will be A as Q_0 is the beginning state.

The grammar G for the $RE 10 + (0 + 11)0 * 1$ is V_N, Σ, P, S where

$$VN = \{A, B, C, D\}$$
$$\Sigma = \{0, 1\}$$

P : $\{A \to 0B/1C, B \to 0D/1C/0, C \to 0C/1D/1\}$

S : $\{A\}$.

(N.B: From *NFA* without $\epsilon - move$, the regular grammar can be constructed. We can do this by the construction of the equivalent *DFA*.)

P : $\{A \rightarrow 0B/1C, B \rightarrow 0D/1C/0, C \rightarrow 0C/1D/1\}$

S : $\{A\}$.

(N.B: From *NFA* without $\epsilon - move$, the regular grammar can be constructed. We can do this by the construction of the equivalent *DFA*.)

5.10 Constructing FA from Regular Grammar

P : $\{A \rightarrow 0B/1C, B \rightarrow 0D/1C/0, C \rightarrow 0C/1D/1\}$
S : $\{A\}$.
(N.B: From *NFA* without $\epsilon - move$, the regular grammar can be constructed. We can do this by the construction of the equivalent *DFA*.)

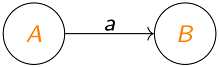
5.10 Constructing FA from Regular Grammar

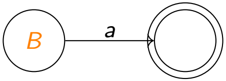
FA can directly be constructed from regular grammar. This can be considered as a reverse process of constructing the FA from an RE. The following section describes the process of constructing the FA from an RE.

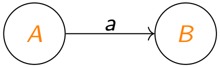
Step I:

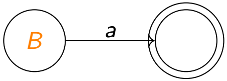
Step I:

- If the grammar does not produce any null string, then the number of states of the *FA* is equal to the number of non-terminals of the regular grammar $+1$. Each state of the FA represents each non-terminal and the extra state is the final state of the FA. If it produces a null string, then the number of states is the same as the number of non-terminals.
- The initial state of the FA is the start symbol of the regular grammar.
- If the language generated by the regular grammar contains a null string, then the initial state is also the final state of the constructing *FA*.

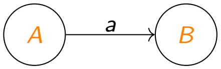

A
a
B

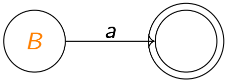

B
a





Step II:





Step II:

- For a production in the form $A \rightarrow aB$, make a δ function $\delta(A, a)$ $\rightarrow B$.

  There is an arc from state A to state B with label a.
- For a production in the form $A \rightarrow a$, make a δ function $\delta(A, a) \rightarrow$ final state.
- For a production $A \rightarrow \epsilon$, make a δ function $\delta(A, \epsilon) \rightarrow A$, and A is the final state.

  Consider the following examples.

Example 5.23 Convert the following regular grammar into FA:

$S \to aA/bB/a/b$

$A \to aS/bB/b$

$B \to aA/bS.$

Example 5.23 Convert the following regular grammar into FA:

$S \rightarrow aA/bB/a/b$

$A \rightarrow aS/bB/b$

$B \rightarrow aA/bS$.

Solution: In the grammar, there are three non-terminals, namely S, A, and B. Therefore, the number of states of the FA is four. Let us name the final state as C.

Example 5.23 Convert the following regular grammar into FA:

$S \to aA/bB/a/b$

$A \to aS/bB/b$

$B \to aA/bS$.

Solution: In the grammar, there are three non-terminals, namely S, A, and B. Therefore, the number of states of the FA is four. Let us name the final state as C.

$(i) For the production S \to aA/bB, the transitional diagram is given in Fig.5.32(a)$

$(ii) For the production S \to$

$a/b, the transitional diagram including the previous one becomes as Fig.5.32(b)$.

Example 5.23 Convert the following regular grammar into FA:

$S \rightarrow aA/bB/a/b$

$A \rightarrow aS/bB/b$

$B \rightarrow aA/bS.$

Solution: In the grammar, there are three non-terminals, namely S, A, and B. Therefore, the number of states of the FA is four. Let us name the final state as C.

$(i) For the production S \rightarrow aA/bB, the transitional diagram is given in Fig.5.32(a)$

$(ii) For the production S \rightarrow a/b, the transitional diagram including the previous one becomes as Fig.5.32(b).$

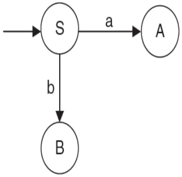


Fig. 5.32(a)

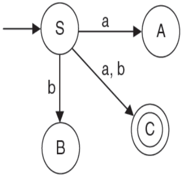


Fig. 5.32(b)

Regular Expression | 259

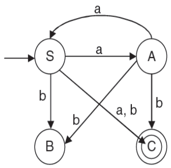


Fig. 5.32(c)

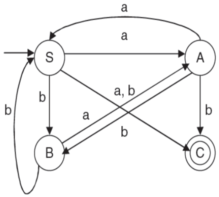


Fig. 5.32(d)

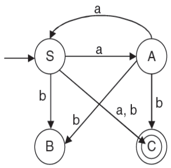


Fig. 5.32(c)

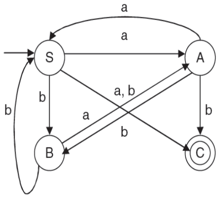


Fig. 5.32(d)

(iii) For the production $A \to aS/bB/b$, the transitional diagram including the previous one looks like Fig. 5.32(c).

For the production $B \to aA/bS$, the transitional diagram including the previous one looks like Fig. 5.32(d)

This is the FA for the given regular grammar.

Example 5.24 Convert the following regular grammar into FA:

$S \rightarrow aA/bS$

$A \rightarrow bB/a$

$B \rightarrow aS/b.$

Example 5.24 `Convert the following regular grammar into FA:`

$S \rightarrow aA/bS$

$A \rightarrow bB/a$

$B \rightarrow aS/b.$

Solution: In the grammar, there are three non-terminals, namely S, A, and B. So, the number of states of the FA will be four. Let us name the final state as C.

Example 5.24 `Convert the following regular grammar into FA:`

$S \rightarrow aA/bS$

$A \rightarrow bB/a$

$B \rightarrow aS/b.$

Solution: In the grammar, there are three non-terminals, namely S, A, and B. So, the number of states of the FA will be four. Let us name the final state as C.

- For the production $S \rightarrow aA/bS$, the transitional diagram becomes
- For the production $A \rightarrow bB/a$, the transitional diagram including the previous one is
- For the production $B \rightarrow aS/b$, the transitional diagram including the previous one looks like

  This is the FA for the given regular grammar. The construction process is described in Figs. 5.33(a) to 5.33(c).

Example 5.24 Convert the following regular grammar into FA:

$S \rightarrow aA/bS$

$A \rightarrow bB/a$

$B \rightarrow aS/b.$

Solution: In the grammar, there are three non-terminals, namely S, A, and B. So, the number of states of the FA will be four. Let us name the final state as C.

- For the production $S \rightarrow aA/bS$, the transitional diagram becomes
- For the production $A \rightarrow bB/a$, the transitional diagram including the previous one is
- For the production $B \rightarrow aS/b$, the transitional diagram including the previous one looks like

  This is the FA for the given regular grammar. The construction process is described in Figs. 5.33(a) to 5.33(c).

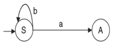


Fig. 5.33(a)

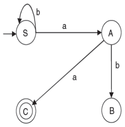


Fig. 5.33(b)

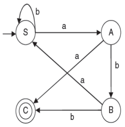


Fig. 5.33(c)

5.11 Pumping Lemma for Regular Expression

There is a necessary condition for an input string to belong to a regular set. This necessary condition is the pumping lemma. Pumping means generating. This lemma is called a pumping lemma because it gives a method of generating many input strings from a given string.

As the pumping lemma is a necessary condition for a string to belong to a regular set, it is used to prove that certain sets are not regular.If any set fulfills all the conditions of Pumping Lemma it can not be said confirm that the set is regular.

But the reverse is true, i.e., if any set breaks the conditions of the pumping lemma, it can be said that the set is not regular. So, the pumping lemma is used to prove that certain sets are not regular.

Theorem 5.2*: Statement of the pumping lemma: Let*$M = \{Q, \Sigma, \delta, q_0, F\}$ be an FA with n number of states. Let L be a regular set accepted by M. Let w be a string that belongs to the set L and $\mid w \mid \geq m$. If $m \geq n$, i.e., the length of the string is greater than or equal to the number of states, then there exists x, y, z such that $w = xyz$, where $\mid xy \mid \leq n, \mid y \mid > 0$ and $xyiz \in L$ for each $i \geq 0$.

(It needs some clarification, after that we shall go to the proof of it.

Let us take the following FA in Fig. 5.34. Here, from q0, by getting 'a' it goes to q1 and from q1 by getting 'b' it goes to q_2, which is the final state. The FA consists of three states, q_0, q_1, and q_2. The string that is accepted by the FA is ba. The length of the string is 2 which is less than the number of states of the FA.

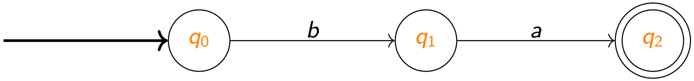

$$\begin{tikzcd} \arrow[r] & q_0 \arrow[r, "b"] & q_1 \arrow[r, "a"] & q_2 \end{tikzcd}$$

Fig. 5.34

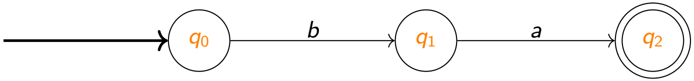


Fig. 5.34

Here, from a state by getting a single input, it goes to a single distinct state. But, if the length of the string is equal or greater than the number of states of the FA, then from a state by getting a single input it is not possible to get all single distinct states. There must be a repetition of at least a single state. This can be described by the following diagram in Fig. 5.35.

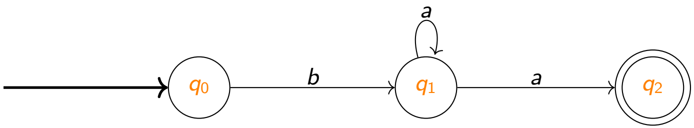


Fig. 5.35

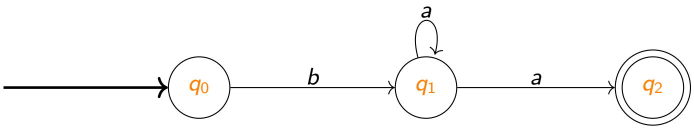


Fig. 5.35

The RE accepted by the automata is ba*b. The expression can be divided into three parts: x, y, z where y is the looping portion, x is the portion before looping, and z is the portion after the looping portion.)

Proof: Let w be a string of length m where m is greater than n, the number of states of the FA accepting the string w as given in Fig. 5.36.