2. **State Determination**

Since the initial state of a FSM is known and the input sequence is given and hence output sequence can be determined. So it is always possible to discover some unknown state, in which the FSM resides at a particular instance.

3. **Impossibility of multiplication**

A FSM cannot remember long sequences and for multiplication operation there is a need to remember two full long sequences, one for multiplier and other for multiplicand. Also the model should be capable of remembering partial sums that are obtained at intermediate stages of multiplication. Therefore no FSM can multiply two given arbitrarily long numbers.

4. **Impossibility of Palindrome Recognition**

No FSM can recognize a palindrome since it cannot remember long sentences, so it cannot remember all the symbols it read until the half way of the input sequence in order to match them with the symbols in the other half of the input sequence in the reverse order.

5. **Impossibility to check well-formedness of parenthesis**

As FSM has no capability to remember all the earlier inputs to it, cannot compare with the remaining to check well formedness. It is impossible task for any FSM.

1.2 DFA AND NFA

1.2.1 Deterministic Finite Automata (DFA)

❑ **Definition :** DFA consist of finite set of state, one state is called start state and there can be one or more final states. In DFA from each state on each i/p symbol there is exactly one transition.

DFA is represented using five tuple representation and it is defined as follows.

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

Where,

Q = finite set of states

Σ = i/p alphabet

δ = transition function $\delta, : Q \times \Sigma \rightarrow Q$

q_0 = Start state q_0 E Q

F = Finite set of final states $F \subseteq Q$

e.g. :

$Q = \{q_A, q_B, q_C\}$

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

$q_0 = q_A$

$F = \{q_C\}$

δ :-

$Q \backslash \Sigma$	0	1
$\rightarrow q_A$	q_A	q_B
q_B	q_C	q_B
q_C^*	q_A	q_B

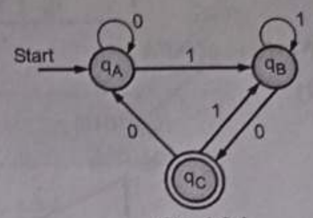


Fig. 1.2.1

Working of DFA

(1) $(q_A, 110)$
$\vdash (q_B, 10)$
$\vdash (q_B, 0)$
$\vdash (q_C, \in)$
<u>Accept</u>

(2) $(q_A, 1001)$
$\vdash (q_B, 001)$
$\vdash (q_C, 01)$
$\vdash (q_A, 1)$
$\vdash (q_B, \in)$
<u>Reject</u>

1.2.2 Non-Deterministic Finite Automata (NFA)

❑ **Definition :** NFA consists of finite set of states. One state is called start state and there can be one or more final states.

In NFA, from each sate, on each i/p symbol there can be 0, 1 or more transitions.

NFA is represented using five tupple representation and it is defined as follows :

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

Where,

Q = finite set of states

Σ = i/p alphabet

δ = transition function $\delta, : Q \times \Sigma \rightarrow 2^Q$

q_0 = start state q_0 E Q

F = finite set of final states $F \subseteq Q$

e.g.

$Q = \{q_A, q_B, q_C\}$

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

$q_0 = q_A$

$F = \{q_C\}$

δ :

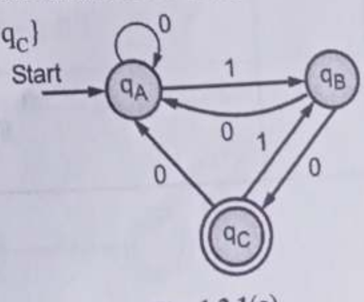


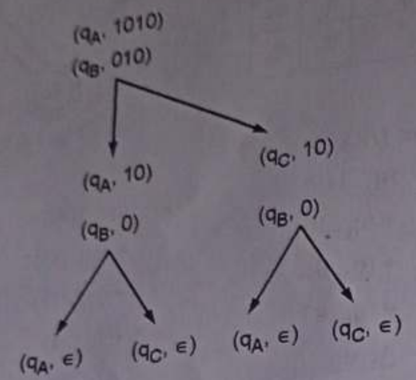
Fig. 1.2.1(a)

δ :-

$Q \setminus \Sigma$	0	1
$\rightarrow q_A$	q_A	q_B
q_B	$\{q_A, q_C\}$	{ }
q_C^*	q_A	q_B

Working of NFA

(1)



(2) $(q_A, 110)$

$(q_B, 10)$

Reject

Note : If any of leaf notes contains a final state then you accept the string otherwise you reject it.

NOTES

1.2.3 Difference between NFA and DFA

MU - Q. 1(b), May 19, 5 Marks

UQ. Differentiate DFA and NFA.

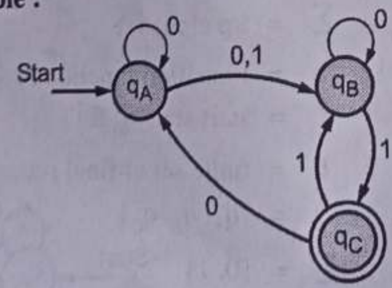
<table>
<tr><th>Sr. No.</th><th>NFA</th><th>DFA</th></tr>
<tr><td>1.</td><td>In NFA from each state on each i/p symbol there can be 0, 1 or more transitions.</td><td>In DFA from each state on each input symbol there is exactly one transition.</td></tr>
<tr><td>2.</td><td>In NFA the transition function is defined as $\delta : Q \times \Sigma \rightarrow 2^Q$</td><td>In DFA the transition function is defined as $\delta : Q \times \Sigma \rightarrow Q$</td></tr>
<tr><td>3.</td><td>In NFA we can have epsilon (∈) transition.</td><td>In DFA we cannot have epsilon (∈) transition.</td></tr>
<tr><td>4.</td><td>The implementation of NFA is difficult because of its non-deterministic nature.</td><td>The implementation of DFA is simple.</td></tr>
<tr><td>5.</td><td>Example :


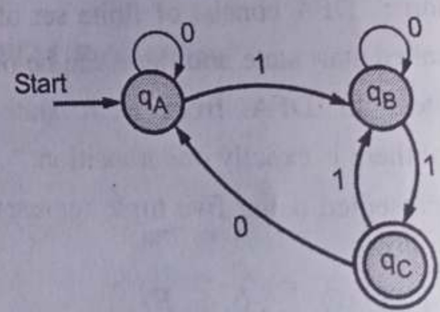
Fig. 1.2.2</td><td>Example :


Fig. 1.2.3</td></tr>
</table>

1.3 TYPES OF PROBLEMS ON NFA AND DFA

1. RE to NFA
2. NFA to DFA
3. DFA to minimised DFA (By classical method)
4. NFA with epsilon (∈) to NFA without Epsilon (∈)
5. NFA without Epsilon (∈) to DFA

6a. Transition Diagram (TD) to R.E. (By Arden theorem)

6b. Transition diagram to RE (By state elimination method)

7. FSM to DFA
8. DFA to minimised DFA (By box method)

1.3.1 Conversion of RE to NFA

Guidelines

- Divide the given R.E. into smaller sub expression and create NFA for each using Rule 1, 2 and 3
- Combine the above NFA's using Rule 4.

Rule 1 : NFA for r = φ i.e.{ }

Fig. 1.3.1

Rule 2 : NFA for r = ∈ i.e. {∈}

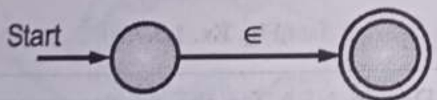


Fig. 1.3.2

Rule 3 : NFA for r = a i.e.{ a }

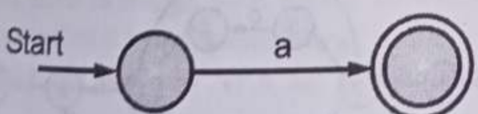


Fig. 1.3.3

Rule 4 : (i) NFA for r = (R) / (S) i.e. L(R) ∪ L(S)

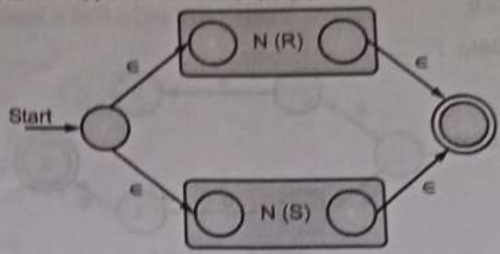


Fig. 1.3.4

(ii) NFA for r = (R) · (S) i.e. L(R) · L(S)

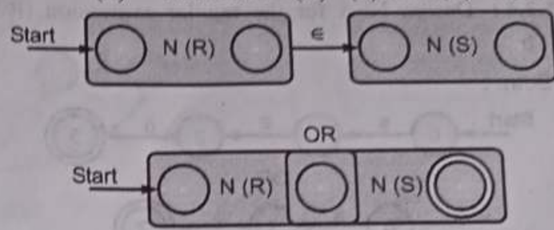


Fig. 1.3.5

(iii) NFA for r = $(R)^*$ i.e. $L(R)^*$

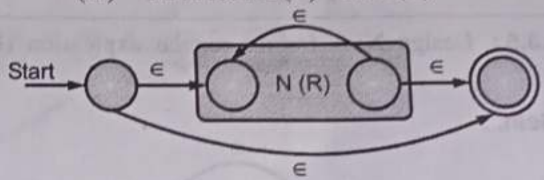


Fig. 1.3.6

1.3.1(a) Solved Examples on Conversion of R.E. to NFA

Ex. 1.3.1 : Design NFA for the regular expression (R·E.) r = a

☑ **Soln. :**

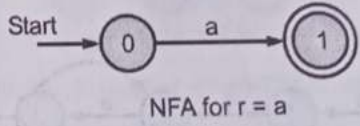


Fig. Ex.1.3.1

Ex. 1.3.2 : Design NFA for the regular expression (R·E.) r = b

☑ **Soln. :**

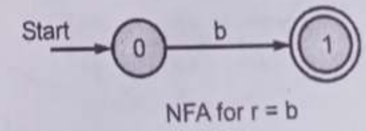
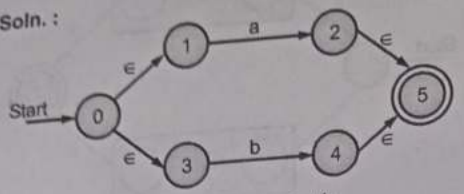


Fig. Ex.1.3.2

Ex. 1.3.3 : Design NFA for the regular expression (R·E.) $r = a + b$

☑ **Soln. :**

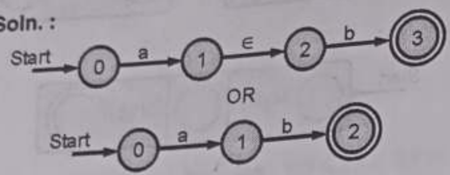


NFA for r = a + b

Fig. Ex.1.3.3

Ex. 1.3.4 : Design NFA for the regular expression (R·E.) $r = a \cdot b$

☑ **Soln. :**

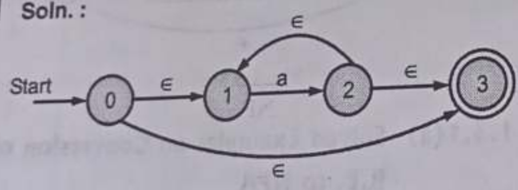


NFA for r = a . b

Fig. Ex.1.3.4

Ex. 1.3.5 : Design NFA for the regular expression (R·E.) $r = a^*$

☑ **Soln. :**

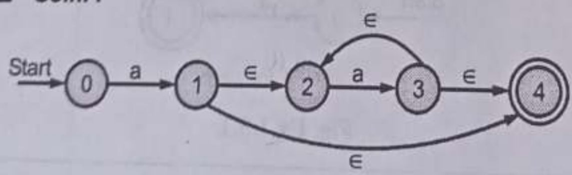


NFA for r = a*

Fig. Ex.1.3.5

Ex. 1.3.6 : $r = a^+$ i.e. equivalent to $\cdot\, a \cdot a^*$

☑ **Soln. :**



NFA for r = a⁺

Fig. Ex. 1.3.6

Ex. 1.3.7 : Design NFA for the regular expression (R·E.) $r = (a + b)^*$

☑ **Soln. :**

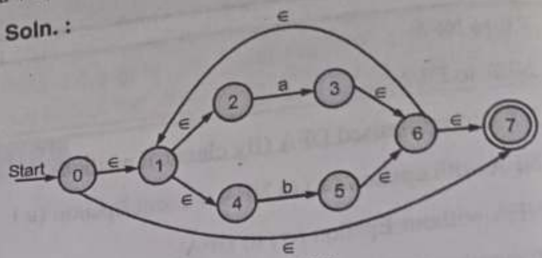
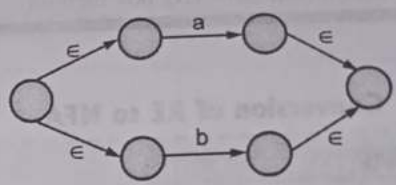


Fig. Ex.1.3.7

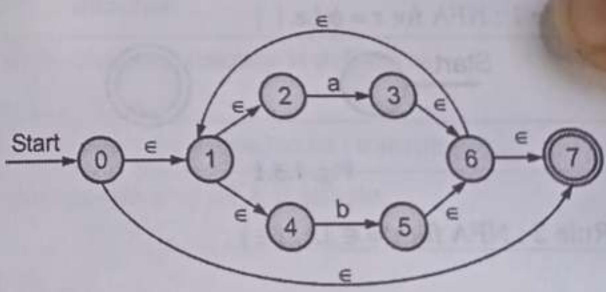
Note

- Here we need to design NFA for $r = (a + b)^*$ and we have created it in the following way :
- So first we created NFA for $(a + b)$ in the following way.



(1A53)**Fig. Ex. 1.3.7(a)**

- Now we will apply closure on $(a + b)$ create NFA for $r = (a + b)^*$



(1A54)**Fig. Ex. 1.3.7(b)**

Ex. 1.3.8 : Design NFA for the regular expression (R·E.) $r = 0(0 + 1)^*\,10$

☑ **Soln. :**

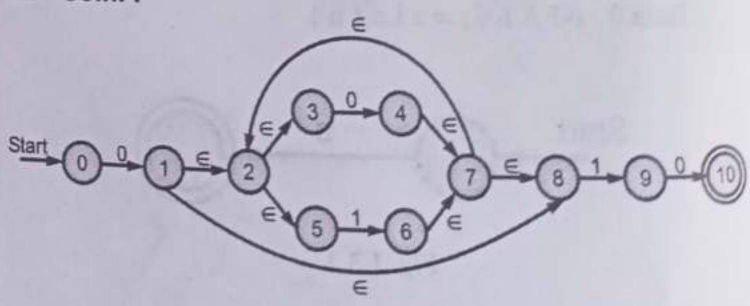
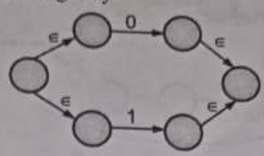


Fig. Ex.1.3.8

Note

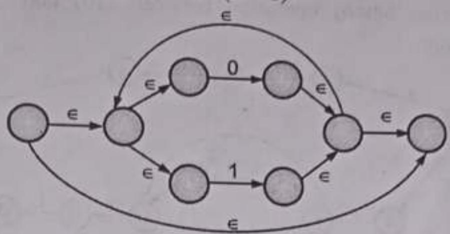
Here we need to design NFA for $r = 0\,(0 + 1)^*\,10$ and we have created it in the following way :

- So first we create NFA for $(0 + 1)$ in the following way



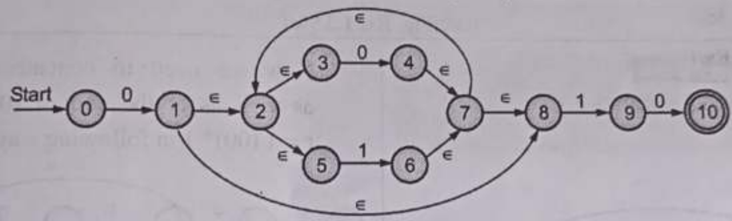
(1A55)Fig. Ex. 1.3.8(a)

Now we apply closure on $(0 + 1)$ and create NFA for $r = (0 + 1)^*$



(1A56)Fig. Ex. 1.3.8(b)

Therefore now we concatenate 0 at the start and 10 at the end of $(0 + 1)^*$ and we create NFA for $r = 0\,(0 + 1)^*\,10$



(1A57)Fig. Ex. 1.3.8(c)

Ex. 1.3.9 : Design NFA for the regular expression (R.E.) $r = 10 + (0 + 10)^*\,1$

☑ **Soln. :**

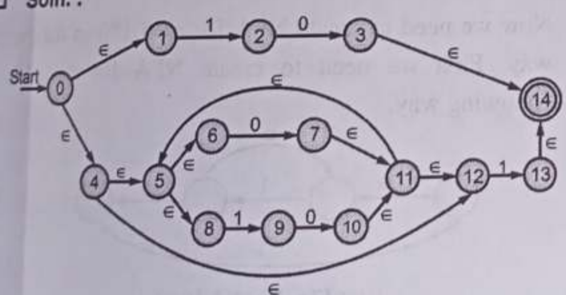
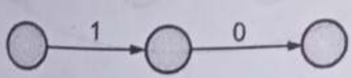


Fig. Ex.1.3.9

Note :

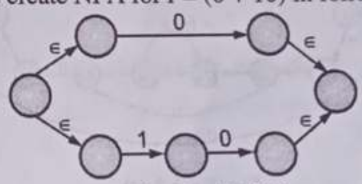
Here we need to design NFA for $r = 10 + (0 + 10)^*\,1$ and we have created it in following way.

- First we need to create NFA for $r = 10$ in following way.



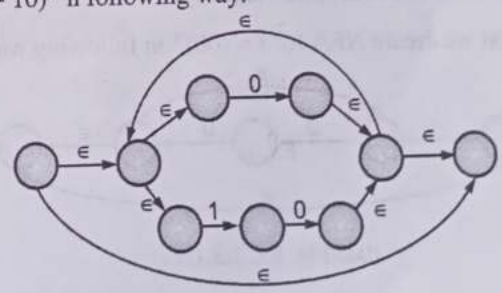
(1A58)Fig. Ex. 1.3.9(a)

- Now we need to create NFA for $r = (0 + 10)^*\,1$ in following way.
- First we create NFA for $r = (0 + 10)$ in following way.



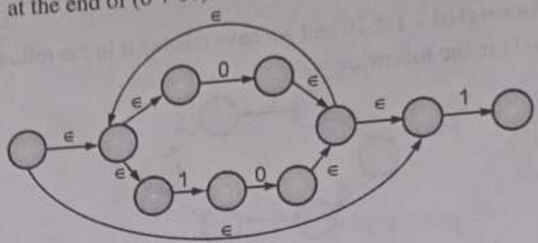
(1A59)Fig. Ex. 1.3.9(b)

Now we apply closure on $(0 + 10)$ and create NFA for $r = (0 + 10)^*$ n following way.



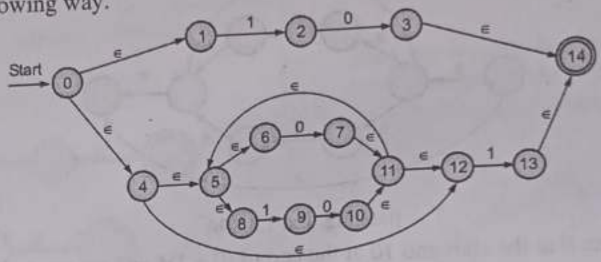
(1A60)Fig. Ex. 1.3.9(c)

Now we need to concatenate 1 at the end of $(0 + 10)^*$ 1 and create NFA for $r = (0 + 10)^* \cdot 1$ in following way.



(1A61)Fig. Ex. 1.3.9(d)

Now we need to perform OR (i.e. union) operation between (10) and $(0 + 10)^* \cdot 1$ and create NFA for $r = 10 + (0 + 10)^* \cdot 1$ in the following way.



(1A62)Fig. Ex. 1.3.9(e)

UEx. 1.3.10 **MU – Dec. 11, 5 Marks**

Convert the following RE to NFA with ∈

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

☑ **Soln. :**

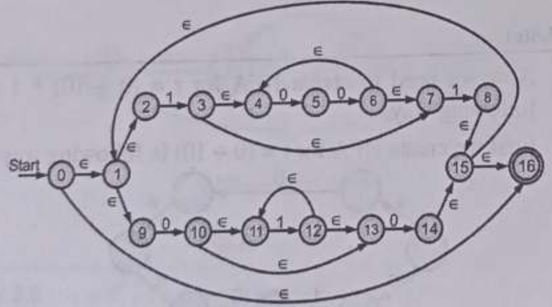
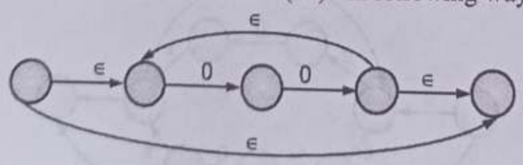


Fig. Ex.1.3.10

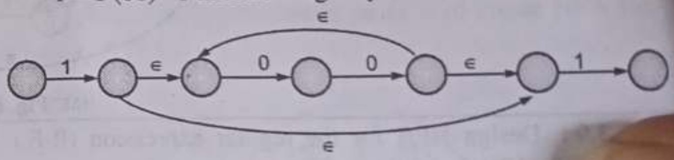
Note : Here we need to design NFA for $r = (1 (00)^* 1 + 01^* 0)^*$ and we create it in following way.

– First we create NFA for $r = (00)^*$ in following way.



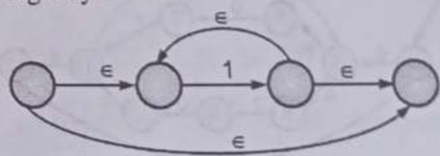
(1A63)Fig. Ex. 1.3.10(a)

– Now we need to concatenate 1 both at the start as well as at the end of $(00)^*$ and create NFA for $r = 1 (00)^* 1$ in following way :



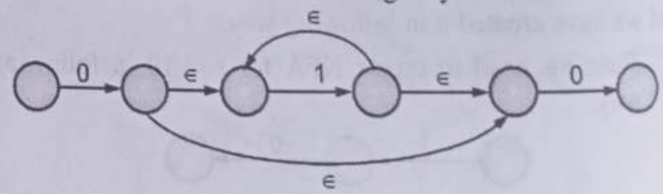
(1A64)Fig. Ex. 1.3.10(b)

– Now we need to create NFA for $r = 01^*0$ in following way. First we need to create NFA for $r = 1^*$ in following way.



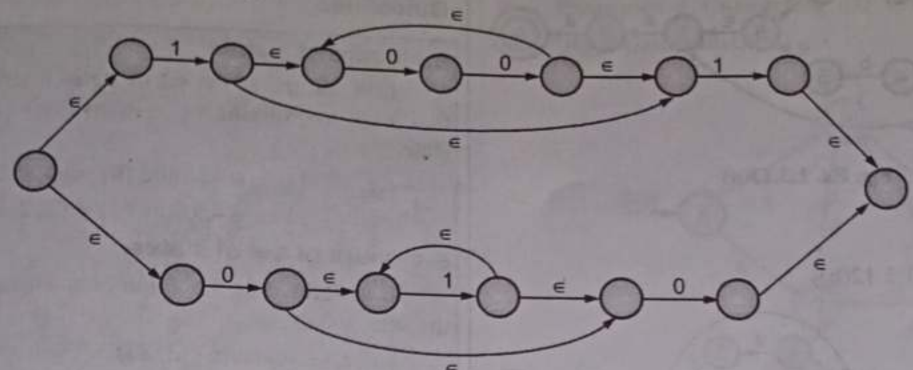
(1A65)Fig. Ex. 1.3.10(c)

Now we need to concatenate 0 both at the start as well as at the end of 1^* in following way.



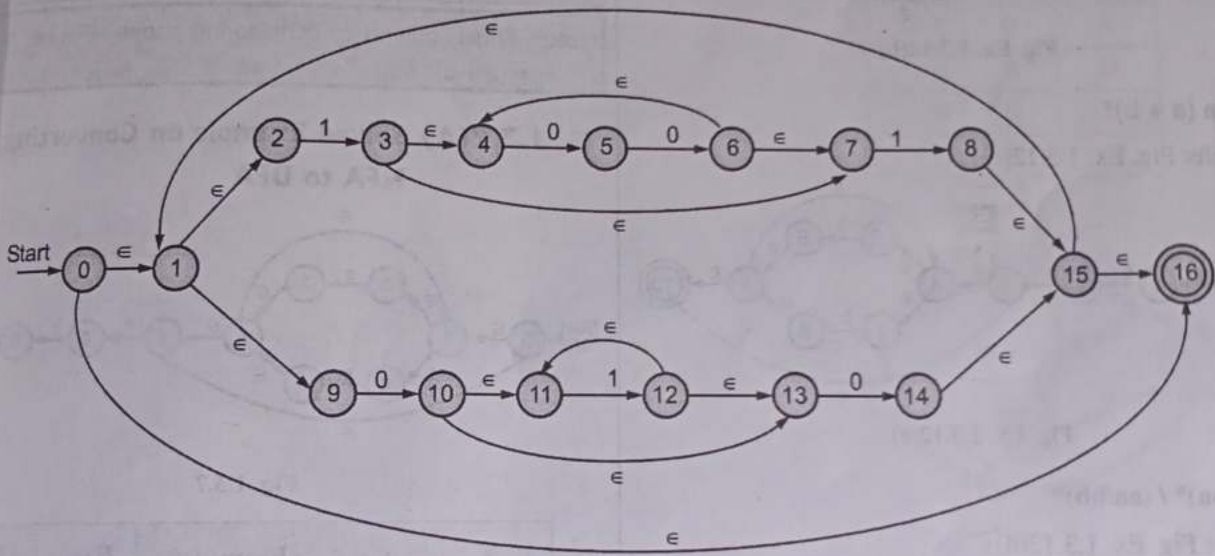
(1A66)Fig. Ex. 1.3.10(d)

- Now we need to perform OR (i.e union) operation between r = 1 (00)* and r = 01* 0 and create NFA for r = 1 (00)* 1 + 01* 0 in following way.



(1A67)Fig. Ex. 1.3.10(e)

- Now we need to apply closure on (1 (00)* 1 + 01* 0) and design NFA for r = (1 (00)* 1 + 01* 0)* in following way.



(1A68)Fig. Ex. 1.3.10(f)

Ex. 1.3.11 : Design NFA for the regular expression (R.E.) r = (a + b)*abb

☑ **Soln. :**

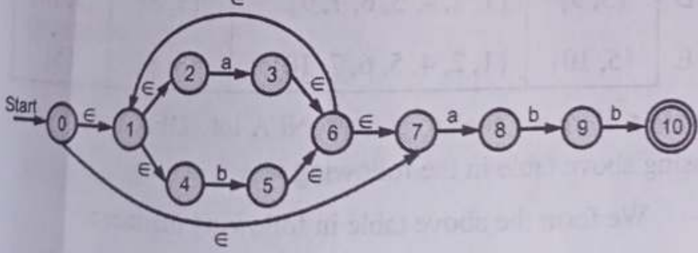


Fig. Ex.1.3.11

Ex. 1.3.12 : Construct NFA for given regular expression :

(i) (a + b)*ab, (ii) aa (a + b)*b,

(iii) aba (a + b)*, (iv) (ab/ba)*/(aa/bb)*

☑ **Soln. :**

(i) (a + b) * ab

(Refer Fig. Ex. 1.3.12(a))

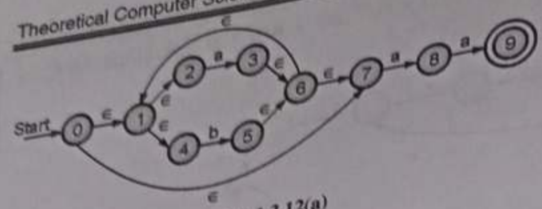


Fig. Ex. 1.3.12(a)

(ii) aa (a + b)* . b

(Refer Fig. Ex. 1.3.12(b))

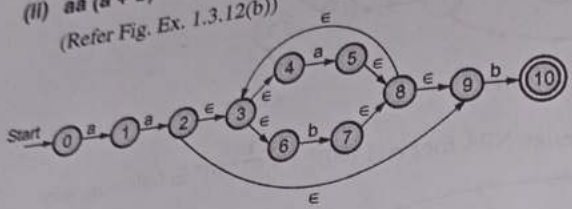


Fig. Ex. 1.3.12(b)

(iii) aba (a + b)*

(Refer Fig. Ex. 1.3.12(c))

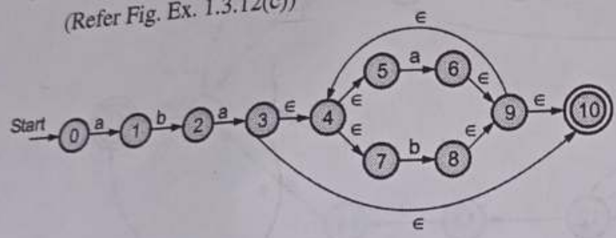


Fig. Ex. 1.3.12(c)

(iv) (ab/ba)* / (aa/bb)*

(Refer Fig. Ex. 1.3.12(d))

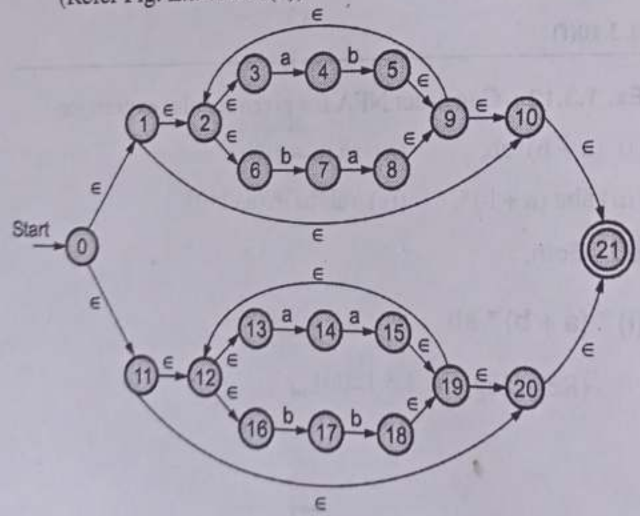


Fig. Ex. 1.3.12(d)

1.3.2 Conversion of NFA to DFA

Guidelines

∈ closure of a state

It is defined as the set of states that are reachable from that state by walking on ∈-transitions only including that state.

e.g. (1) ∈-closure (0) = {0, 1, 2, 4, 7}

(2) ∈-closure (5) = {1, 2, 4, 5, 6, 7}

∈ closure of set of states

It is defined as the union of ∈-closures of each state of the set.

e.g. : ∈-closure ({3, 8})

= ∈-closure (3) ∪ ∈ - closure (8)

= {1, 2, 3, 4, 6, 7} ∪ {8}

= {1, 2, 3, 4, 6, 7, 8}

Note : This is solved by considering above NFA i.e. NFA for r = (a + b)* abb

1.3.2(A) Solved Example on Converting NFA to DFA

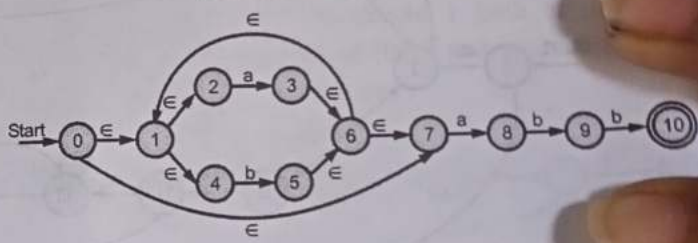


Fig. 1.3.7

	X	y = ∈ - closure (x)	δ (y, a)	δ (y, b)
A	{0}	{0, 1, 2, 4, 7}	{3, 8}	{5}
B	{3, 8}	{1, 2, 3, 4, 6, 7, 8}	{3, 8}	{5, 9}
C	{5}	{1, 2, 4, 5, 6, 7}	{3, 8}	{5}
D	{5, 9}	{1, 2, 4, 5, 6, 7, 9}	{3, 8}	{5, 10}
E	{5, 10}	{1, 2, 4, 5, 6, 7, 10}	{3, 8}	{5}

Note : Here we need to convert NFA into DFA and we do it using above table in the following way.

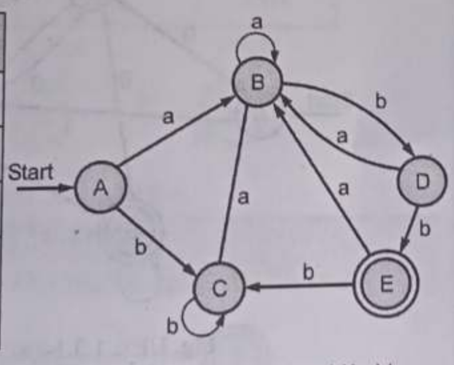
- We form the above table in following manner :
- We start with start state i.e. in this case its state 0 and we perform ∈-closure on it and the states we get on performing ∈-closure form the part of y i.e. in this case y = {0, 1, 2, 4, 7}.

- Now we check that is there transition on input 'a' from the states that are mentioned in y and in this case, there is a transition from states 2 and 7 on input 'a' and they go to states 3 and 8 respectively. Therefore in this case $\delta(y, a) \rightarrow \{3, 8\}$.
- Similarly we check is there a transition on input 'b' from the states that are mentioned in y and in this case, there is a transition from state '4' on input 'b' and it goes to state 5.
- Therefore in this case $\delta(y, b) \rightarrow \{5\}$. Now the states $\{3, 8\}$ and $\{5\}$ make part of X and we perform ϵ-closure on it and the above process goes on continuing till we stop getting new states.
- Now we give unique names to the states that are part of column X and we form a new transition table (δ) as shown below :

Note : If the final state is the part of y, then we make that y's state as final state as shown in the below table. In this case, 10 is the final state and it is part of y of state E. Therefore in the below table we have made state E as final.

δ :

$Q \setminus \Sigma$	a	b
$\rightarrow$A	B	C
B	B	D
C	B	C
D	B	E
E^*	B	C



DFA for r = (a + b)* abb

Fig. 1.3.8

1.3.3 DFA to minimised DFA (By Classical Method)

Guidelines

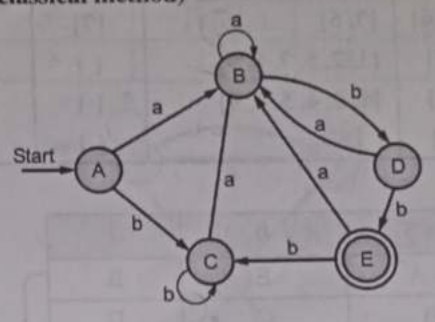
States can be merged if

(All states have same transition) and (All are final OR all are non-final)

Module 1

1.3.3(A) Solved Examples

Examples : Converting DFA to minimised DFA (By classical method)

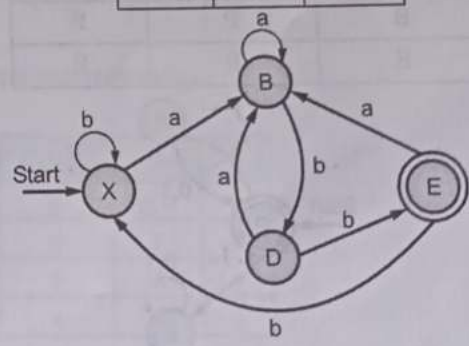


DFA for r = (a + b)* abb

Fig. 1.3.9

δ :

$Q \setminus \Sigma$	a	b
$\rightarrow$X	B	X
B	B	D
D	B	E
E^*	B	X



Min DFA for r = (a + b)* abb

Fig. 1.3.10

1.3.3(B) Solved Examples on Type 1, 2, 3

Ex. 1.3.13 : Construct FA for r = (11 + 10)*

☑ **Soln. :**

- **Step 1 :** R.E. to NFA :

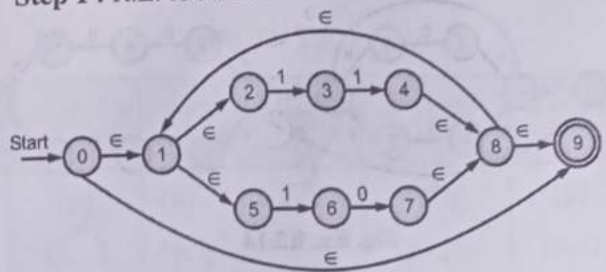


Fig. Ex. 1.3.13

▸ **Step 2 :** NFA to DFA

	X	y = ∈ - closure (x)	δ (y, 0)	δ (y, 1)
A	{0}	{0, 1, 2, 5, 9}	{ }	{3, 6}
B	{3, 6}	{3, 6}	{7}	{4}
C	{7}	{1, 2, 5, 7, 8, 9}	{ }	{3, 6}
D	{4}	{1, 2, 4, 5, 8, 9}	{ }	{3, 6}
E	{ }	{ }	{ }	{ }

δ :

Q \ Σ	0	1
→ A*	E	B
B	C	D
C*	E	B
D*	E	B
E	E	E

▸ **Step 3 :** DFA to min DFA :

Q \ Σ	0	1
→ P*	E	B
B	P	P
E	E	E

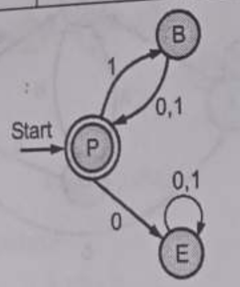


Fig. Ex. 1.3.13(a)

UEx. 1.3.14 **MU – Dec. 09, 10 Marks**

Construct NFA from $r = (0 + 1)^* (00 + 11)$ and convert into min DFA.

☑ **Soln. :**

▸ **Step 1 :** R.E. to NFA

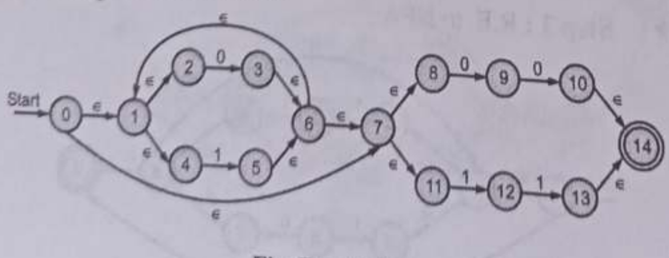


Fig. Ex. 1.3.14

▸ **Step 2 :** NFA to DFA

	x	y = ∈ - closure (x)	δ (y, 0)	δ (y, 1)
A	{0}	{0, 1, 2, 4, 7, 8, 11}	{3, 9}	{5, 12}
B	{3, 9}	{3, 6, 7, 8, 11, 1, 2, 9, 4}	{3, 9, 10}	{5, 12}
C	{5, 12}	{1, 2, 4, 5, 6, 7, 8, 11, 12}	{3, 9}	{5, 12,13}
D	{3, 9, 10}	{1, 2, 4, 3, 6, 7, 8, 11, 9, 10, 14}	{3, 9, 10}	{5, 12}
E	{5, 12, 13}	{ 1, 2, 4, 5, 6, 7, 8, 1, 12, 13, 14}	{3, 9}	{5, 12, 13}

δ :

Q \ Σ	0	1
→ A	B	C
B	D	C
C	B	E
D*	D	C
E*	B	E

▸ **Step 3 :** DFA to min DFA

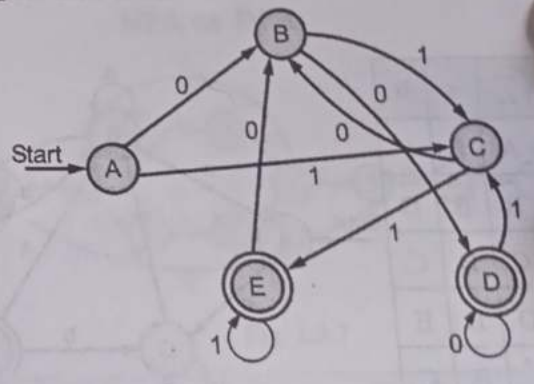


Fig. UEx. 1.3.14(a)

1.3.4 NFA with Epsilon (∈) to NFA withou Epsilon (∈)

Guidelines

(1) The number of states remain the same.

(2) Start state remains the same.

(3) Final states remain the same.

(4) If the ∈ closure of the start state contains a final s then make the start state also a final state.

Module 1

1.3.4(A) Solved Examples

- **Examples :** Converting NFA with Epsilon (∈) to NFA without Epsilon (∈)

Ex. 1.3.15

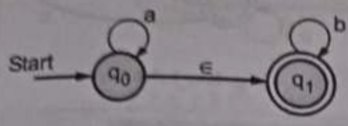


Fig. Ex. 1.3.15

☑ **Soln. :**

x	y = ∈ closure (x)	P = δ (y, a)	∈-closure (p)	q = δ (y, b)	∈-closure (q)
q_0	$\{q_0, q_1\}$	$\{q_0\}$	$\{q_0, q_1\}$	$\{q_1\}$	$\{q_1\}$
q_1	$\{q_1\}$	{ }	{ }	$\{q_1\}$	$\{q_1\}$

δ :

Q \ Σ	a	b
$\rightarrow q_0^*$	$\{q_0, q_1\}$	$\{q_1\}$
q_1^*	{ }	$\{q_1\}$

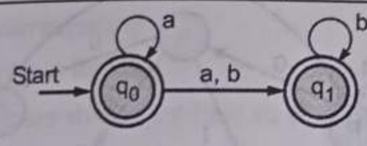


Fig. Ex. 1.3.15(a)

UEx. 1.3.16 **MU - Dec.10, 5 Marks**

Convert the following NFA with ∈ to NFA without ∈.

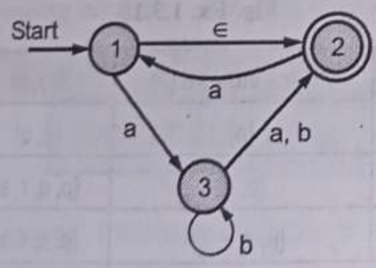


Fig. Ex. 1.3.16

☑ **Soln. :**

x	y = ∈ closure (x)	P = δ (y, a)	∈-closure (p)	q = δ (y, b)	∈-closure (q)
1	{1, 2}	{1, 3}	{1, 2, 3}	{ }	{ }
2	{2}	{1}	{1, 2}	{ }	{ }
3	{3}	{2}	{2}	{2, 3}	{2, 3}

δ :

Q \ Σ	a	b
→ 1*	{1, 2, 3}	{ }
2^*	{1, 2}	{ }
3	{2}	{2, 3}

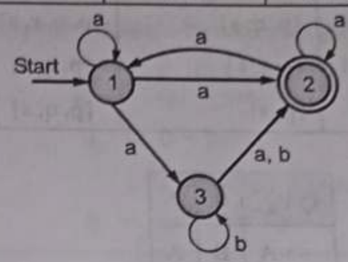


Fig. Ex. 1.3.16(a)

1.3.5 NFA without Epsilon (∈) to DFA

1.3.5(A) Solved Examples

Ex. 1.3.17 : Convert NFA to DFA :

$$\left(\frac{(p, q, r, s)}{Q}, \frac{(0, 1)}{\Sigma}, \frac{\delta}{\delta}, \frac{P}{S}, \frac{(S)}{F}\right)$$

(Give tupple representation)

δ :

Q \ Σ	0	1
p	p, q	p
q	r	r
r	s	–
s	s	s

☑ **Soln. :**

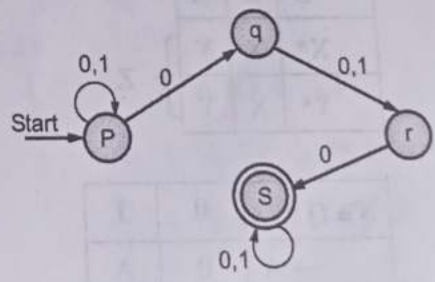


Fig. Ex. 1.3.17

	x	y = ∈ - closure (x)	δ (y, 0)	δ (y, 1)
A	{p}	{p}	{p, q}	{p}
B	{p, q}	{p, q}	{p, q, r}	{p, r}
C	{p, q, r}	{p, q, r}	{p, q, r, s}	{p, r}