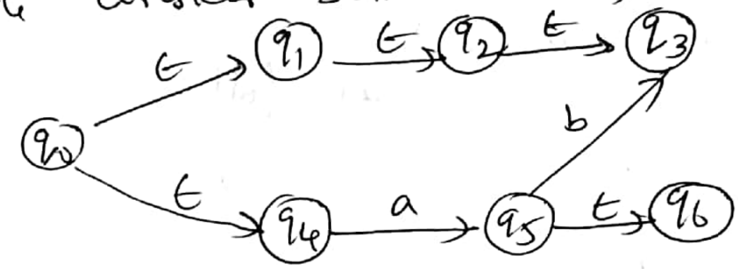
Equivalence of Non-Deterministic & Deterministic FSM's :-

* Given the NDFSM that accepts certain language L. It is possible to convert it into DFSM that also accepts the same language L.
* It is also possible to convert DFSM into its equivalent NDFSM.
* ϵ-NDFSM can also be converted into DFSM

Epsilon closures :- Epsilon closure of a state q denoted by ECLOSE(q) is a set of states reached from state q by following a path whose label is epsilon (ϵ) only.

For example consider some FSM,



ECLOSE(q_0) = {q_0, q_1, q_2, q_3, q_4}

ECLOSE(q_1) = {q_1, q_2, q_3}

ECLOSE(q_2) = {q_2, q_3}

ECLOSE(q_3) = {q_3}

ECLOSE(q_4) = {q_4}

ECLOSE(q_5) = {q_5, q_6}

Formally, we define ϵ-closure(q) recursively as follows:

Basis: state q is in ECLOSE(q).

Induction: If state p is in ECLOSE(q) and if there is a transition from state p to state r labelled with ϵ, then r is in ECLOSE(q).

That is $q \xrightarrow{\epsilon} p \xrightarrow{\epsilon} r \xrightarrow{a} s$

$$\text{ECLOSE}(q) = \{q, p, r\}$$

Theorem: If there is an NDFSM or ϵ-NDFSM for L, there is a DFSM for L.

Proof: Proof is by Construction of DFSM D.

Let $N = (Q_N, \Sigma_N, \delta_N, q_{0N}, F_N)$ be ~~an~~ a given NDF or ϵ-NDF.

We need to Construct equivalent DFSM $D = (Q_D, \Sigma_D, \delta_D, q_{0D}, F_D)$. The Components of D are Constructed as follows:

(1). Q_D is a set of subsets of Q_N. If Q_N has n states then Q_D has 2^n states. among these 2^n states, only those states that are reachable from start state of DFSM are the states of DFSM. We Compute ϵ-closure of those states that can be reached from start state q_0 of DFSM.

(2). q_{0D} = ECLOSE(q_{0N}) start state of NDFSM is obtained by Computing ϵ-CLOSE of start state of NDFSM.

3. $\delta_D(S, a)$ is computed for all symbols a in Σ and all sets S in Q_D by

(a) Let $S = \{p_1, p_2, \ldots p_k\}$
(subset)

(b) Compute $\bigcup_{i=1}^{k} \delta_N(p_i, a)$. Let this be $\{r_1, r_2, \ldots r_m\}$

(c) Then $\delta_D(S, a) = \bigcup_{j=1}^{m} ECLOSE(r_j)$.

4. F_D is those sets of subsets that contains atleast one Final (accepting) state of N.

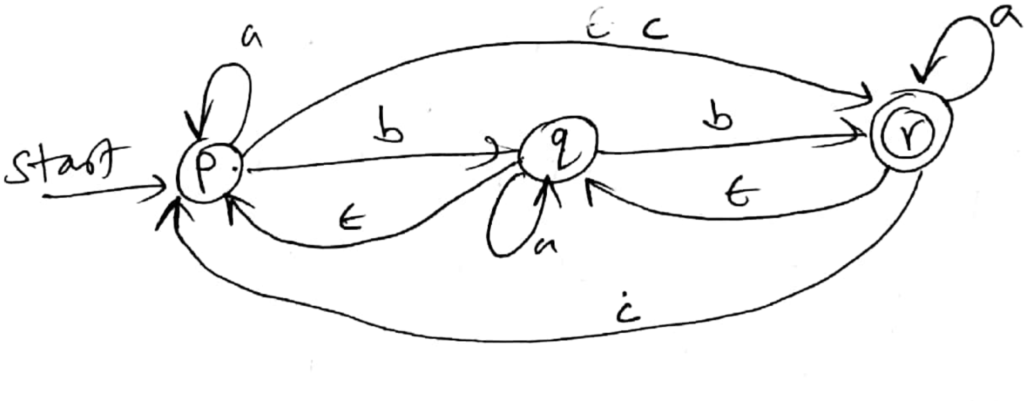
That is: $F_D = \{ S \mid S$ is in Q_D and $S \cap F_N \neq \phi \}$

Problems:

(1) Convert the following ϵ-NDFSM into DFSM.

δ_N	ϵ	a	b	c
→ p	ϕ	{p}	{q}	{r}
q	{p}	{q}	{r}	ϕ
* r	{q}	{r}	ϕ	{p}

Step (1): Construct Transistion Diagram of NDFSM.



Scanned by CamScanner

Step(ii) Compute ECLOSE of each state

$$\boxed{\begin{aligned} ECLOSE(p) &= \{p\} \\ ECLOSE\{q\} &= \{q, p\} = \{p, q\} \\ ECLOSE\{r\} &= \{r, q, p\} = \{p, q, r\} \end{aligned}}$$

Now we need to construct equivalent DFA $D = (Q_D, \Sigma_D, \delta_D, q_{0D}, F_D)$.

(1) $\Sigma_D = ?$ We know $\Sigma_N = \Sigma_D$
and $\Sigma_N = \{a, b, c\}$
(alphabet of NDFA)

$$\boxed{\therefore \Sigma_D = \{a, b, c\}}$$
alphabet of DFA

(2) $q_{0D} = ?$ $q_{0D} = ECLOSE(\underbrace{\{q_{0N}\}})$

(start state of DFSM)

$\downarrow$ start state of ϵ-NDFA

$q_{0D} = ECLOSE(\{p\})$

$$\boxed{q_{0D} = \{p\}}$$

(3) $\delta_D = ?$ Begin with start state $q_{0D} = \{p\}$.

δ_D	a	b	c
$\rightarrow \{p\}$	$\{p\}$	$\{p, q\}$	$\{p, q, r\}$
$\{p, q\}$	$\{p, q\}$	$\{p, q, r\}$	$\{p, q, r\}$
$* \{p, q, r\}$	$\{p, q, r\}$	$\{p, q, r\}$	$\{p, q, r\}$

Fig:- Transition table of DFSM

$$\delta_D(\{p\}, a) = \delta_N(p, a) = \{p\}$$
$$\text{ECLOSE}(\{p\})$$

$$\underline{\delta_D(\{p\}, a) = \{p\}}$$

$$\delta_D(\{p\}, b) = \delta_N(p, b) = \{q\}$$
$$\text{ECLOSE}(\{q\}) =$$

$$\underline{\delta_D(\{p\}, b) = \{p, q\}}$$

$$\delta_D(\{p, q\}, a) = \delta_N(p, a) \cup \delta_N(q, a)$$
$$\{p\} \cup \{q\}$$
$$= \{p, q\}$$
$$\text{ECLOSE}(p) \cup \text{ECLOSE}(q)$$
$$\{p\} \cup \{p, q\}$$
$$=$$
$$\underline{\delta_D(\{p, q\}, a) = \{p, q\}}$$

$$\delta_D(\{p, q\}, b) = \delta_N(p, b) \cup \delta_N(q, b)$$
$$= \{q\} \cup \{r\}$$
$$= \text{ECLOSE}(\{q\}) \cup \text{ECLOSE}(r)$$
$$= \{p, q\} \cup \{p, q, r\}$$
$$\underline{\delta_D(\{p, q\}, b) = \{p, q, r\}}$$

(4) $Q_D = \{\{p\}, \{p, q\}, \{p, q, r\}\}$ set of states of DFA.

(5) $F_D = ?$

We know $F_N = \{r\}$

& $Q_D = \{ \{p\}, \{p, q\}, \{p, q, r\} \}$

F_N contains those states of Q_D that contains r.

$\therefore F_N = \{p, q, r\}$

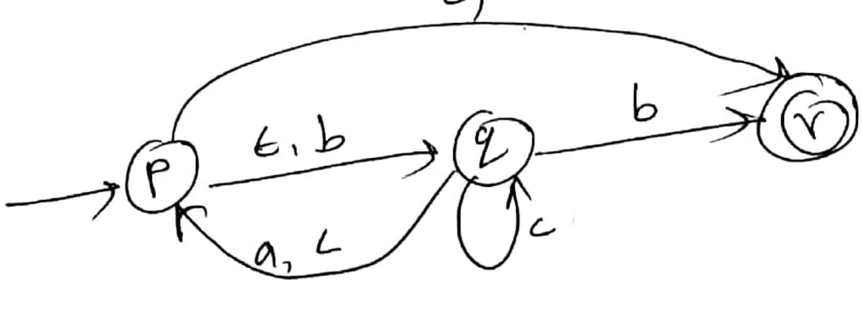
Transistion Diagram representation of DFSM is:

Problem (2)

Convert the following ε-NDFSM into equivalent DFSM.

δ_N	ε	a	b	c
→ p	{q, r}	φ	{q}	{r}
q	φ	{p}	{r}	{p, q}
* r	φ	φ	φ	φ

Step ① Construct Transistion Diagram of ε-NDFSM



Step (ii) Compute ϵ-closure of each state.

$$\boxed{\begin{array}{l} ECLOSE(p) = \{p, q, r\} \\ ECLOSE(q) = \{q\} \\ ECLOSE(r) = \{r\} \end{array}}$$

Now Construct DFSM $D = (Q_D, \Sigma_D, \delta_D, q_{0D}, F_D)$

(i) $\Sigma_D = ?$ We know $\Sigma_N = \Sigma_D$.

& $\Sigma_N = \{a, b, c\}$

$\boxed{\therefore \Sigma_D = \{a, b, c\}}$

(ii) $q_{0D} = ?$ (start state of DFSM)

$q_{0D} = ECLOSE(\{q_{0N}\})$

We know $q_{0N} = p$. (start state of ϵ-NDFSM)

$q_{0D} = ECLOSE(\{p\})$

$\boxed{q_{0D} = \{p, q, r\}}$

(iii) $\delta_D = ?$ Construct δ_D Using δ_N (Use Transistion table of ϵ-NDFSM)

δ_D	a	b	c
$\rightarrow \{p, q, r\}$	$\{p, q, r\}$	$\{q, r\}$	$\{p, q, r\}$
$* \{q, r\}$	$\{p, q, r\}$	$\{r\}$	$\{p, q, r\}$
$* \{r\}$	ϕ	ϕ	ϕ

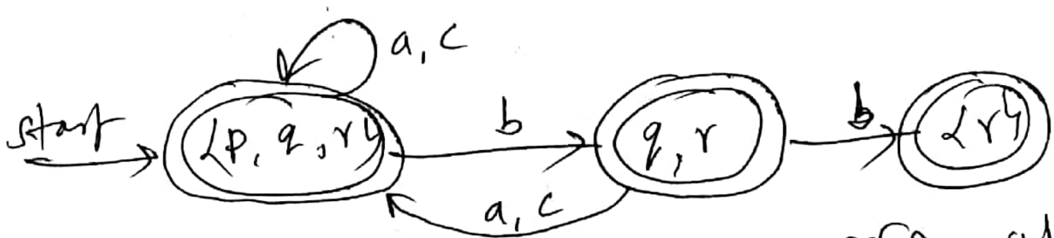


Fig: Transistion Diagram of DFA after eliminating ϵ- transistion (epsilon)

From FSMs to operational systems

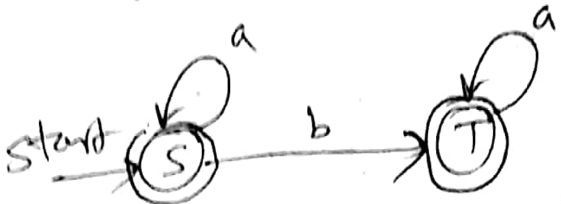
FSMs for real problems can be converted into operational systems in the following ways.

1. An FSM can be translated into a circuit design and implemented directly in hardware. For example parity checking FSM can be implemented as a hardware.
2. An FSM can be interpreted as a general purpose interpreter.
3. An FSM can be used as a specification for some critical aspect of the behaviour of a complex system.

Simulators for FSM:- Once FSM is designed for a given problem, we can simulate its execution.

For example, Consider the following deterministic FSM that accepts the language

$L = \{ w \in \{a, b\}^* : w \text{ contains no more than one } b \}$.



FSM M can be viewed as a specification for the following program,

```
Until accept or reject do:
S:      s = get-next-symbol
        if s = end-of-file then accept
        Else if s=a then goto S
        Else if s=b then goto T
        Else if
```

T : s = get-next-symbol
If s = end-of-file then accept
Else if s = a then goto T
Else if s = b then reject.

End.

Given an FSM with states n, This approach will create a program of length ; $2 + (|n| \cdot |\Sigma| + 2)$

Minimizing FSM :- (Important)

It refers to eliminating redundant states and thereby reducing the states of the FSM.

The main advantage of minimizing the states of DFSM is to reduce the length of the program.

Our goal is to convert the DFSM to a program that must be efficient in terms of time & space.

Table filling algorithm is used to Identify distinguishable and equivalent (in-distinguishable) pair of states

Algorithm: input: Pair of states (p, q)

output: Set of Equivalent (indistinguishable) and distinguishable pair of states.

method:

1. If p is Final (accepting) state and q is Non-Final state and Vice Versa

then pair of states (p, q) is distinguishable

2. For any input symbol a in Σ),
   if $\delta(p, a) = r$ and $\delta(q, a) = s$ and
state r is Final state and state s is Non-Final state and Vice Versa, then pair of states (p, q) are distinguishable

3. Other wise (p, q) are equivalent (in-distinguishable) pair of states.

Problem:

(1) Consider the following DFSM.

δ	0	1
$\to q_1$	q_2	q_3
q_2	q_3	q_5
$* q_3$	q_4	q_3
q_4	q_3	q_5
$* q_5$	q_2	q_5

(i) Draw a table of Distinguishable & In-distinguishable pair of states.

(ii) Construct minimum state equivalent DFSM.

Solution:

step ①: Draw a table of $(n-1) \times (n-1)$ states where n is the number of states of DFSM.

Here $n = 5$ $n-1 = 4$

q_2	X			
* q_3	X	X		
q_4	X	✓	X	
* q_5	X	X	✓	X
	q_1	q_2	q_3 *	q_4

X – Distinguishable

✓ – Equivalent

Equivalent (In-distinguishable) pair of states:

(q_2, q_4), (q_3, q_5).

Final states: q_3, q_5

Non-Final states: q_1, q_2, q_4

Consider pair of states (q_1, q_5). q_5 is Final state, Since q_1 is non-Final state & q_5 is Final state, pair (q_1, q_5) is distinguishable.

In the table mark (q_1, q_5) as X (Distinguishable)

Consider the pair (q_1, q_3). Since q_1 is Non-Final & q_5 is Final state, pair (q_1, q_3) is distinguishable.

Similarly mark other states by applying step ① of table filling algorithm.

Consider a pair of states (q_1, q_4). There both are Non-Final states.

apply step ② of table filling algorithm.

$\delta(q_1, 0) = q_2 \rightarrow$ Non-Final state

& $\delta(q_4, 0) = q_3 \rightarrow$ Final state

Therefore pair (q_1, q_4) is distinguishable

mark (q_1, q_4) as X.

Consider pair of states (q_1, q_2) Here both are Non-Final states

apply step(2) of table filling algorithms,

$\delta(q_1, 0) = q_2$ - Non-accepting state

$\delta(q_2, 0) = q_3$ - accepting state

$\therefore (q_1, q_2)$ is distinguishable

Important

Consider pair of states (q_2, q_4)

$\delta(q_2, 0) = q_3$ which is Final state

$\delta(q_4, 0) = q_3$ —— " ——

Now apply $\delta(q_2, 1) = q_5$ which is final state

$\delta(q_4, 1) = q_5$ —— " ——

Therefore (q_2, q_4) is Indistinguishable pair (equivalent) of states. mark as ✓.

Now consider pair of states (q_3, q_5)

$\delta(q_3, 0) = q_4$ which is Non-Final state

$\delta(q_5, 0) = q_2$ which is Non Final state.

Now apply $\delta(q_3, 1) = q_3$ - Final state

$\delta(q_5, 1) = q_5$ - Final state

Therefore (q_3, q_5) is equivalent (Indistinguishable)

Step (iii): Now Partition the state set Q into blocks of mutually equivalent states.

We know Set $Q = \{q_1, q_2, q_3, q_4, q_5\}$

and in-distinguishable pair of states: (q_2, q_4), (q_3, q_5).

Set Q can be Partitioned into

$\{\{q_1\}, \{q_2, q_4\}, \{q_3, q_5\}\}$

Step (iv): Minimized DFSM can be obtained as follows

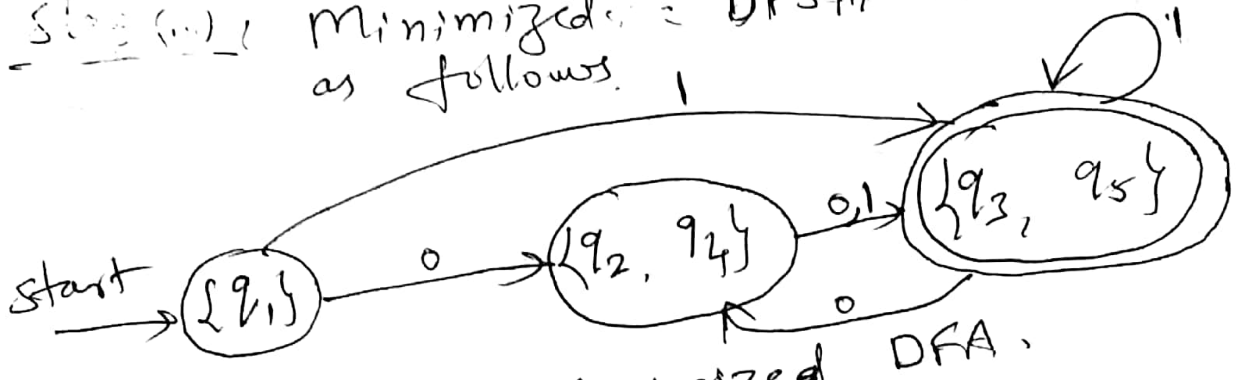


Fig: Minimized DFA.

Note: Use transistion table (or diagram) of given FSM to construct Minimized DFSM.

$\delta(q_1, 0) = q_2$ which is in block (q_2, q_4).

i.e. draw an arc from q_1 to (q_2, q_4) labelled with 0.

$\delta(q_1, 1) = q_3$ which is in block $\{q_3, q_5\}$.

Draw an arc from $\{q_1\}$ to $\{q_3\ q_5\}$ labelled with 1

Now Consider the block {q_2 q_4}

$\delta(q_2, 0) = q_3$ which is in block {q_3, q_5}

draw an arc from q_3 to (q_3, q_5) labelled with 0.

repeat this procedure for all the states of DFSM to obtain Minimized DFSM.

Problem (2): Minimize the following DFSM.

δ	0	1
→ A	B	E
B	C	F
* C	D	H
D	E	H
E	F	I
* F	G	B
G	H	B
H	I	C
* I	A	E

Here n (no. of states) = 9

n−1 = 8

Step ①: Draw a table of n−1 × n−1 states

	A	B	C	D	E	F	G	H
B	✗							
C	✗	✗						
D	✓	✗	✗					
E	✗	✓	✗	✗				
F	✗	✗	✓	✗	✗			
G	✓	✗	✗	✓	✗	✗		
H	✗	✓	✗	✗	✓	✗	✗	
I	✗	✗	✓	✗	✗	✓	✗	✗

✗ – Distinguishable

✓ – equivalent (in-distinguishable)

Equivalent (in-distinguishable) pair of states } (A, D), (A, G), (B, E), (B, H), (C, F), (C, I), (D, G), (E, H), (F, I)

Now re-write the above pair of states as:

(A, D), (A, G), (D, G) = (A, D, G)

(B, E), (B, H), (E, H) = (B, E, H)

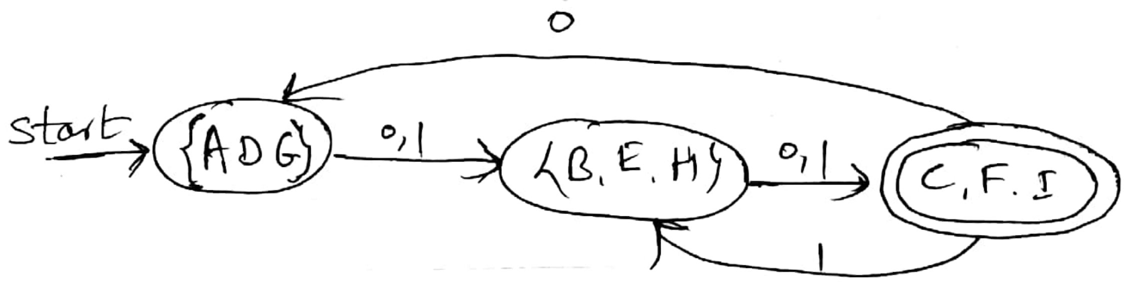
(C, I), (C, F) (F, I) = (C, F, I)

Step (II) Partition state set into Block of mutually equivalent states.

Set Q = {A, B, C, D, E, F, G, H, I}

{{A, D, G}, {B, E, H}, {C, F, I}}

Now Construct Minimum state DFSM



Differences between DFSM & NDFSM

Deterministic Finite State Machine (DFSM)	Non-Deterministic Finite state Machine (NDFSM)
1). It Can determine exactly the Single Next State when it Knows present state and input Symbol.	1). It may or may not determine single next State given present State & input symbol
2) Next state is always a Single state. $\delta(q, a) = p$ — present state, input symbol, Next state	2) Next state may be a set of one or more states. that is $\delta(q, a) = \{p\}$ or $\delta(q, a) = \{p, q\}$
3) It has the ability to stay in exactly one state at any time	4) It has the ability to stay in one or more states
4) It always has exactly one transistion (arc) out of each state for the same input symbol.	4) It may have one or more arcs out of any state labelled with same input symbol
5) It cannot guess on input	5) It may guess on input.
6) It has more number of transistions (arcs)	6) It has less no. of Transistions
7) Difficult to Construct due to restrictions	7) Easy to Construct due to flexibility