4/8/18

Theory of Computation

* Complexity Theory - deals with the level of difficulty
* Computability Theory - problems solvable or not,
* Automata Theory - deals with properties and mathematical models

⇒ for solvable problems - we can find an algorithms.

unsolvable problems - can not "

→ You take an unsolvable problem, prove that it is unsolvable, then it becomes solvable problem.

Finite state automata

Formal Languages

* symbol - Any letter or digit ;
* Alphabet - A finite set of symbols

eg:- $\Sigma = \{a, b, c\}$

* string - ~~Gro~~ sequence of symbols from the alphabet.
* Language - set of strings over a fixed alphabet.

String example:- aa , aab, bba

language example:- {aa, aab, bba}

⇒ Empty string or null string with length zero represented as ε (epsilon).

⇒ ~~Cleen star~~

⇒ KLEENE star (Σ^*)

- set of strings including the empty string

eg:- $\Sigma = \{0, 1\}$

$\Sigma^* = \{\varepsilon, 00, 11, 1\}$

⇒ KLEENE PLUS (Σ^+) or positive closure

set of all strings in the alphabet excluding the null string.

$\Sigma = \{a, b\}$, $\Sigma^+ = \{a, b, ab, ba\}$

$\Sigma^0 = \{\varepsilon\}$

$\Sigma^1 = \{a, b\}$

$\Sigma^2 = \{aa, ab, ba, bb\}$

$\cdot$ —

7/8/17
Monday

⇒ Language 'L' :- set of strings over an alphabet(Σ) which is a subset of Σ^*

⇒ Grammar : rule of language

⇒ $\boxed{L(G) = \{ w \in \Sigma^* \mid S \xRightarrow[G]{*} w \}}$

⇒ Parsing a sentence - finding the components of a sentence

⇒ Start symbol - from where you starts deriving a sentence.

⇒ Terminal or non-terminal.

ex:- $S \rightarrow NP\,VP$ (This is a production rule)

$NP \rightarrow$ noun

$VP \rightarrow$ verb adverb

noun → Geevan / Kavya / Subrtta

verb → run, ran

adverb → quickly, easily

Derivation

S ⇒ (NP VP) → start symbol

⇒ noun VP

⇒ Geevan VP

⇒ Geevan verb adverb

⇒ Geevan ran adverb

⇒ Geevan ran quickly

Parsing

NP, VP, noun - non-terminal

Geevan, ran - terminal

'$\rightarrow$' : Production '$*$' : multiple times.

'$\Rightarrow$' : derivation

Formal Definition

* Grammar can be defined as a 4-tuple

$G = (V, \Sigma, P, S)$ OR (N, T, P, S)

V - finite non empty set of non-terminal

Σ - Finite, non-empty set of terminals

P - finite set of production rule

S - is a start symbol

$V \cap \Sigma = \phi$

$\Rightarrow$ α - a string that has both terminals & non-terminals.

$(V \cup \Sigma)^* = \alpha$

$\Rightarrow$ $\alpha \xRightarrow[G]{*} \beta$: β is a string derived from α over a no. of substitutions using the rules defined on G

$\rightarrow$ $S \xrightarrow{1} baaS$

$S \xrightarrow{2} baa$

Write the Grammar (4-tuple)?

Ans $G = (V, \Sigma, P, S)$

$V = \{S\}$

$\Sigma = \{a, b\}$

S ~~=~~ is S itself

$P = \{S \rightarrow baaS, S \rightarrow baa\}$

Scanned by CamScanner

*

i) $S \Rightarrow baa$ (Applied rule 2)

ii) $S \Rightarrow baaS$ (rule 1)

$\Rightarrow baabaaS$ (rule 1)

$\Rightarrow baa\, baa\, baa\, S$ (rule 1)

$\Rightarrow baa\, baa\, baa\, baa$ (rule 2)

$L(G) = \{baa,\ baa\,baa\,baa\,baa,\ baabaa, \ldots\}$

H.W

1) $S \rightarrow aSb$

$S \rightarrow \varepsilon$

2) $S \rightarrow aCa$

$C \rightarrow aCa \mid b$

3) $S \rightarrow AB$

$A \rightarrow BB$

$B \rightarrow AA$

Find out the language

4) Following are the productions of the given grammar.

$S \rightarrow aABa$

$A \rightarrow baABb$

$B \rightarrow Aab$

$aA \rightarrow baa$

$bBb \rightarrow abab$

18/17
Tuesday

Finite State Automata

1) Designing of sequential Circuit

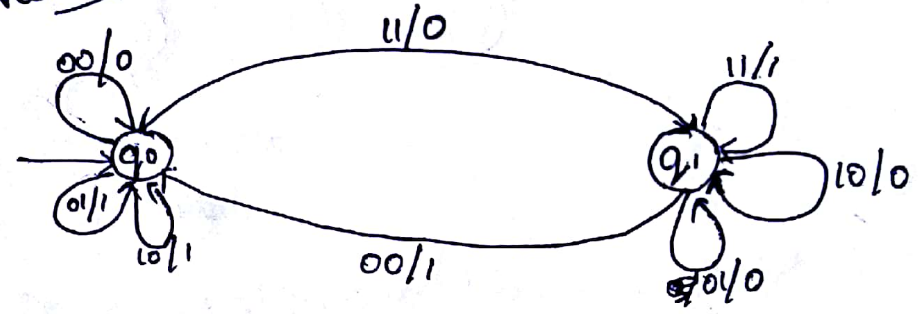
input sets = {00, 01, 10, 11}

output = {0, 1}

$q_0 \to$ state without carry

$q_1 \to$

Mealey machine



Serial adder

$\to$ denotes the initial state

00/0 : $0+0 \to 0$ without carry, so it end up in same state as the initial state.

```
 1   1 1
 0 1 0 0 1 0 1
 0 1 0 1 1 1 0
 -------------
 1 0 1 0 0 1 1
```

* Input string processing finite state automata

$q_0 \xrightarrow{10} q_0 \xrightarrow{01} q_0 \xrightarrow{11} q_1 \xrightarrow{01} q_1 \xrightarrow{00} q_0 \xrightarrow{11} q_1 \xrightarrow{00}$

outputs: 1 1 0 0 1 0 1

Answer : 1010011

→ Least significant bit is generated first

Deterministic Finite State Automata

* Finite state automata can be formally defined as a 5-tuple

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

$Q \rightarrow$ finite set of states

Σ - finite set of input symbols

δ - delta stands for transition

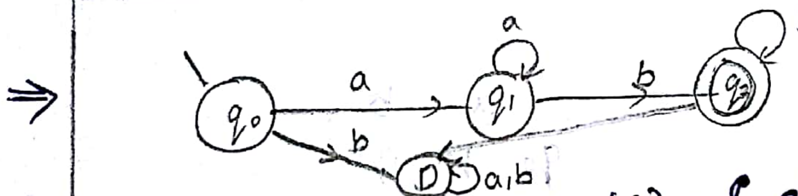
q_0 - initial state (q_0 in Q)

F - set of final state

δ - is a mapping $Q \times \Sigma$ which will result in Q

$$Q \times \Sigma \rightarrow Q$$

$\Rightarrow$



State diagram: for $L(G) = \{ a^n b^m \mid n, m \geq 1 \}$

$\rightarrow$ State transition table

	a	b
$\rightarrow q_0$	q_1	D
q_1	q_1	q_2
q_2	D	q_2
D	D	D

In DFA clearly speak about all states

H.W

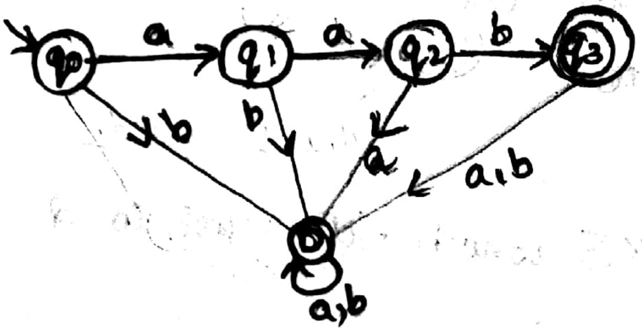
1) $L(M) = \{a^2 b\}$

2) $L(M) = \{\Sigma^*\}$

3) $L(M) = \{(ab)^n \mid n \geq 1\}$

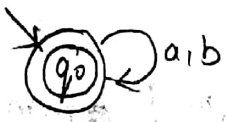
Answers

1) $L(M) = \{a^2 b\}$



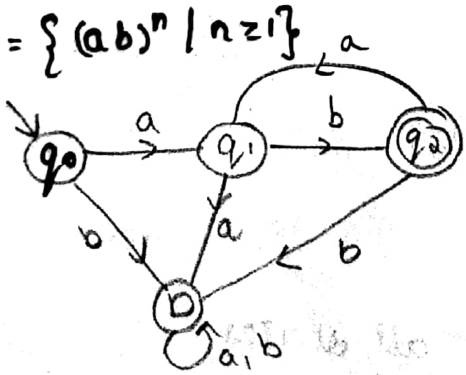
	a	b
→ q0	q1	D
q1	q2	D
q2	D	q3
q3	D	D
D	D	D

2) $L(M) = \Sigma^*$

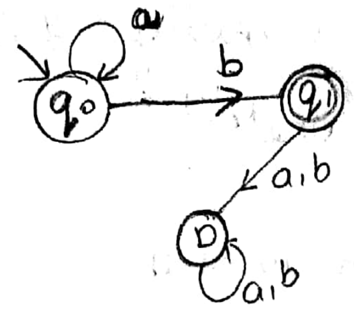


	a	b
q0	q0	q0

3) $L(M) = \{(ab)^n \mid n \geq 1\}$

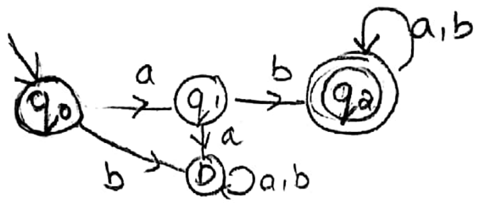


4) $L = \{a^n b \mid n \geq 0\}$

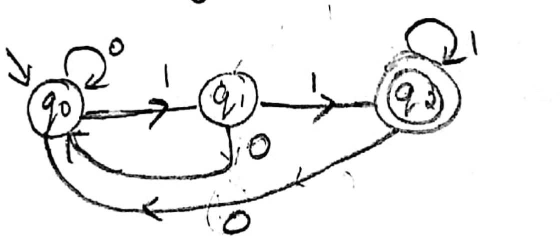


5) Draw a DFA that will accept all string on $\Sigma = \{a, b\}$ starting with prefix ab

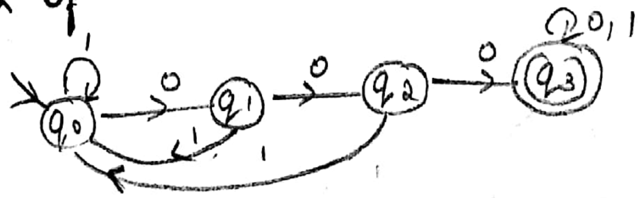
$L = \{ab, abba, aba, abab\}$



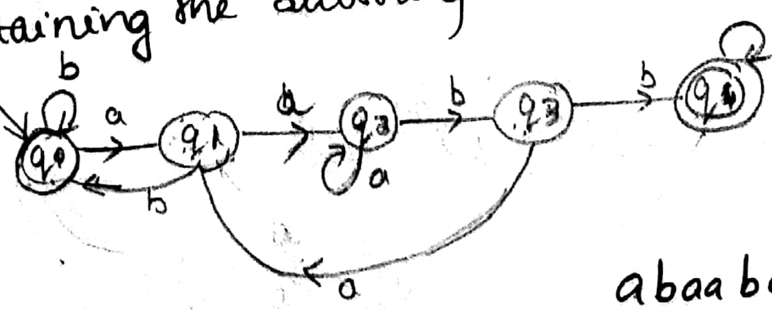
6) Draw all strings ends with Σ, 11



7) strings over of 0's and 1's with three consecutive zeros



8) string containing the substring aabb



abaabaabb

9) All strings of length i) 1, ii) 3

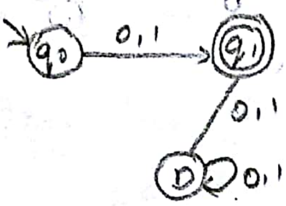
10) All strings of length which is a multiple of 3

11) Strings containing exactly 4 one's

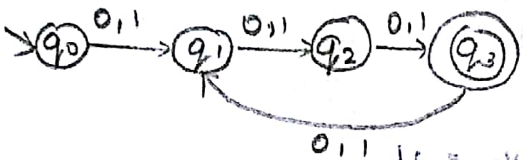
12) strings with odd no: of zero

13) strings that ends with either 00 or 11

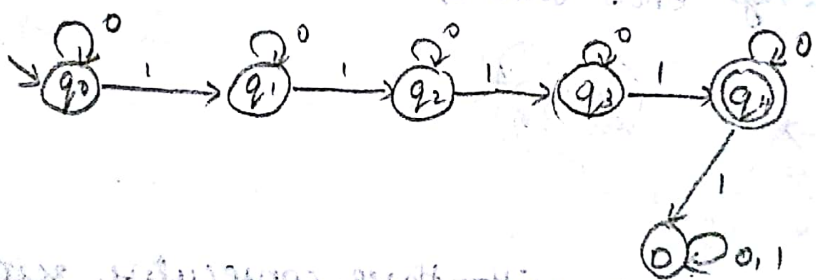
9)



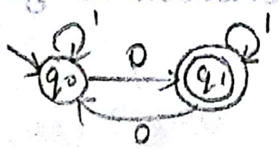
(10)



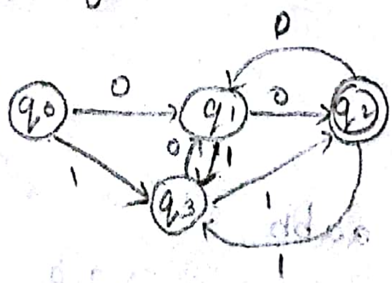
11)



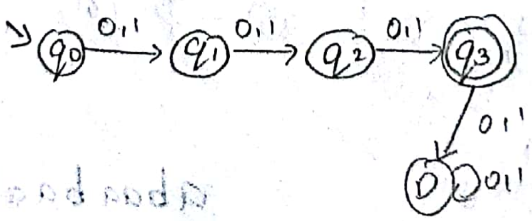
12)



13)

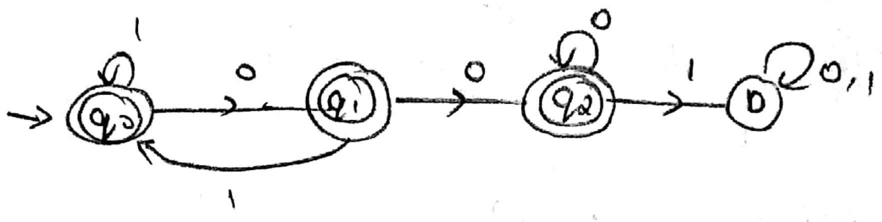


9(ii)

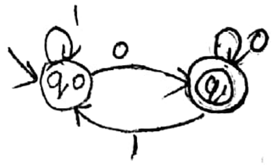


TUTORIALS

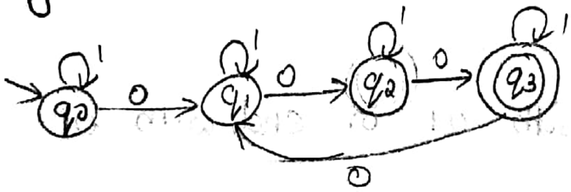
14) All strings except those containing the substring 001.



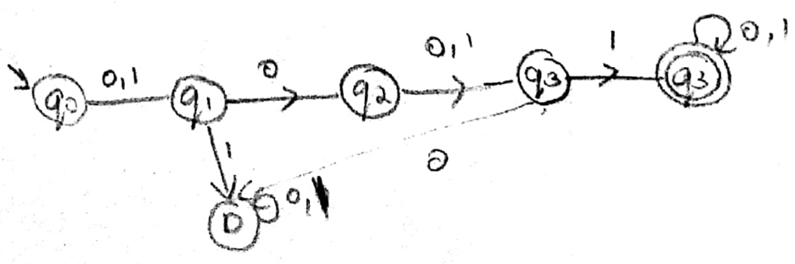
15) FSA with two states accepting even numbers.



16) DFA that accept set of strings where the number of 0's is every string is a multiple of three over the alphabet $\Sigma=\{0,1\}$



17) $L=\{w \in (0,1)$ / second symbol is 0 and fourth input is 1



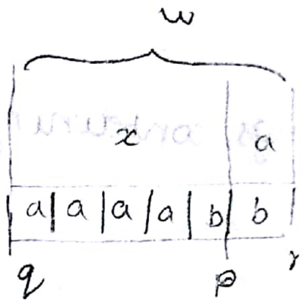
Scanned by CamScanner

19/8/17

$\delta(q_0, a)$ → defined on a single character

$\hat{\delta}(q_0, \omega)$ → defined on a word string

$L(M) = \{ w \in \Sigma^* \mid \hat{\delta}(q_0, \omega) \in F \}$



$\hat{\delta}(q, \omega) = \delta(\hat{\delta}(q, x), a)$

$\delta(q, \varepsilon) = q$

$\hat{\delta}(q, \varepsilon) = q$

⇒ Grammar ~~accepted by~~ supporting FSA is called regular grammar

⇒ Language accepted by FSA is called regular languages

Types of Grammar

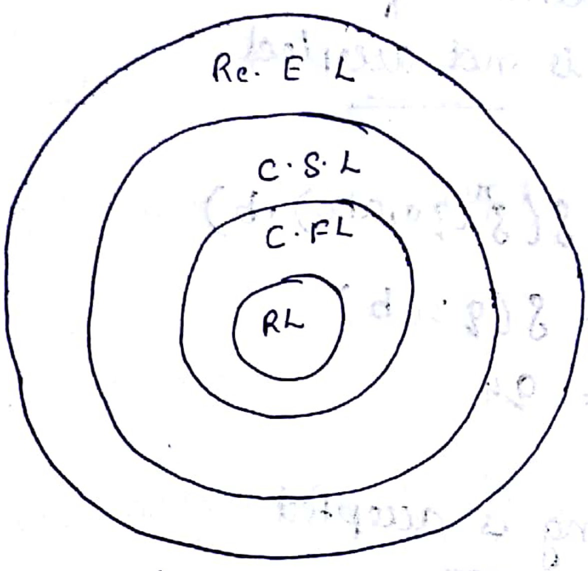
	Types	Grammar	Language	Machine	Reduction rules
1.	Type-0	unrestricted grammar	recursively enumerable languages	Turing machine	$\alpha \to \beta$
2.	Type-1	context sensitive grammar	context sensitive languages	Linear bounded automata	$\alpha A \beta \to \alpha \gamma \beta$
3.	Type-2	context free grammar	context free languages	Push-down automata	$A \to \alpha$
4.	Type-3	regular grammar	regular languages	Finite state automata	$A \to aB$ OR $A \to a$

* $\alpha A \beta \rightarrow \alpha \gamma \beta$: means the replacement of A by γ can be done in the context of α, β.

* $|\alpha| > |\beta|$

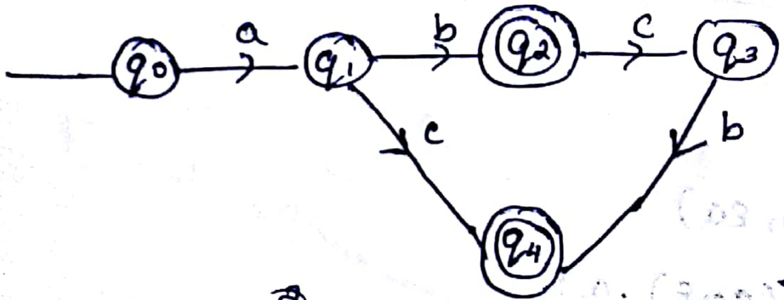
~~Non-Deterministic Finite state automata~~

Chomsky hierarchy



16/8/17
wednesday

Q)



~~[illegible]~~

$Q = \{q_0, q_1, q_2, q_3, q_4)$

$\Sigma = \{a, b, c]$

q_0 - initial state

$F = \{q_2, q_4\}$

	a	b	c
q_0	q_1	D	D
q_1	D	q_2	q_4
q_2	D	D	q_3
q_3	b	q_4	D
q_4	D	D	D

$\hat{\delta}(q_0,a) = \delta(q_0,a)$

Method: I

$\hat{\delta}(q_0,abc) = \delta(\hat{\delta}(q_0,ab),c)$

$= \delta(\delta(\hat{\delta}(q_0,a),b),c)$

$= \delta(\delta(\delta(q_0,a),b),c)$

$= \delta(\delta(q_1,b),c)$

$= \delta(q_2,c)$

$= q_3$

$[\hat{\delta}(q_0,ab) = \delta(\hat{\delta}(q_0,a),b)$

$\hat{\delta}(q_0,a) = ?$

$\delta(q_0,a) = q_1$

$\hat{\delta}(q_0,a) = \hat{\delta}(q_0,\varepsilon a)$

$= \delta(\hat{\delta}(q_0,\varepsilon),a)$

$= \delta(q_0,a) = q_1$

q_3 is not an element of F ($q \notin F$)

$\therefore$ the string is not accepted

Q $\hat{\delta}(q_0,abcb) = \delta(\hat{\delta}(q_0,abc),b)$

$= \delta(q_3,b)$

$= q_4$

$q_4 \in F$

$\therefore$ the string is accepted

Method: II

$\hat{\delta}(q_0,abc)$

$-\hat{\delta}(q_0,a) = \hat{\delta}(q_0,\varepsilon a)$

$= \delta(\hat{\delta}(q_0,\varepsilon),a)$

$= \delta(q_0,a)$

$= q_1$

$\hat{\delta}(q_0,ab) = \delta(\hat{\delta}(q_0,a),b)$

$= \delta(q_1,b)$

$= q_2$

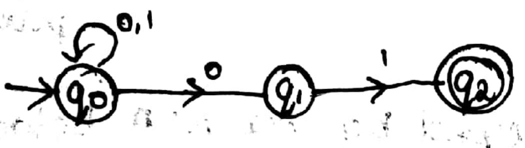
$$\hat{\delta}(q_0, abc) = \delta(\hat{\delta}(q_0, ab), c)$$
$$= \delta(q_2, c)$$
$$= q_3$$

$q_3 \notin F$

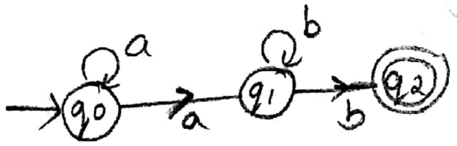
$\therefore$ the string cannot be accepted

Non-deterministic Finite state Automata (NFA)

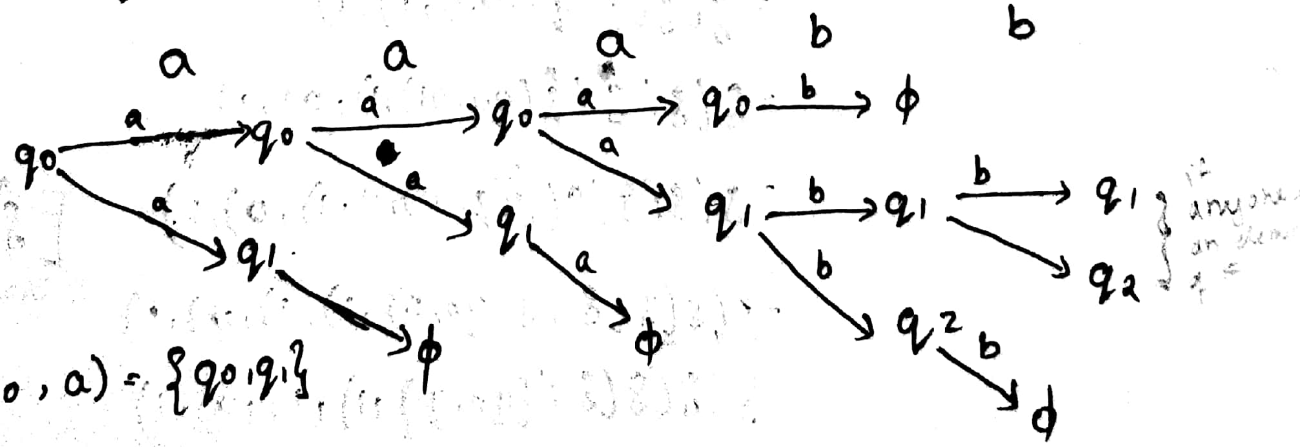
Example



Q) $L(G) = \{ a^n b^m ; n, m \geq 1 \}$ design a NFA



Checking the acceptance of string aaabb



$\delta(q_0, a) = \{q_0, q_1\}$

~~if q1 or q2~~ $q_1 \notin F$
$q_2 \in F$

Hence the string is accepted

★ NFA can be formally defined as a 5-tuple

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

$Q \rightarrow$ ~~state of~~ finite set of states

Σ - finite set of input symbol

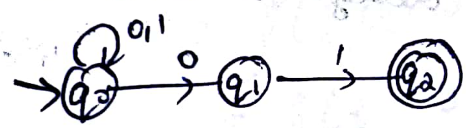
q_0 - initial state

F - set of final state

$\delta : Q \times \Sigma \rightarrow 2^Q$

[$2^Q \rightarrow$ represents any no. of elements in the power set]

Q. check whether 01101 is accepted by the NFA below?



method-I

$$\hat{\delta}(q_0, 01101) = \delta(\hat{\delta}(q_0, 0110), 1)$$
$$= \delta(\delta(\hat{\delta}(q_0, 011), 0), 1)$$
$$= \delta(\delta(\delta(\hat{\delta}(q_0, 01), 1), 0), 1)$$
$$= \delta(\delta(\delta(\delta(\hat{\delta}(q_0, 0), 1), 1), 0), 1)$$

$$\left[\hat{\delta}(q_0, 0) = \hat{\delta}(q_0, \varepsilon 0) = \delta(\hat{\delta}(q_0, \varepsilon), 0) = \delta(q_0, 0)\right]$$

$$= \delta(\delta(\delta(\delta(\delta(q_0, 0), 1), 1), 0), 1)$$
$$= \delta(\delta(\delta(\delta(\{q_0, q_1\}, 1), 1), 0), 1)$$
$$= \delta(\delta(\delta(\delta(q_0, 1) \cup \delta(q_1, 1)), 1), 0), 1)$$
$$= \delta(\delta(\delta(\{q_0, q_2\}, 1), 0), 1)$$
$$= \delta(\delta(\delta(q_0, 1) \cup \delta(q_2, 1)), 0), 1)$$
$$= \delta(\delta(\{q_0\}, 0), 1)$$

Scanned by CamScanner

$\{q_0, q_1\} \rightarrow$ q_0 & q_1 are two different states
$[q_0 q_1] \rightarrow$ $q_0 q_1$ is considered as a single state

$= \delta(\delta(q_0, 0), 1)$
$= \delta(\{q_0, q_1\}, 1)$
$= \delta(q_0, 1) \cup \delta(q_1, 1)$
$= \{q_0, q_2\}$

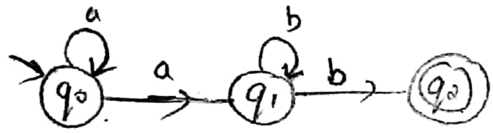
$q_0 \notin F$ but $q_2 \in F$

$\therefore$ the given string can be accepted

11/8/17 Monday

Conversion of NFA to DFA

Q) construct a NFA for language $a^n b^m$; $n, m \geq 1$



[converting to DFA]

Subset construction

① construct the state table ~~from~~ for NFA

	a	b
$\rightarrow q_0$	$\{q_0, q_1\}$	ϕ
q_1	ϕ	$\{q_1, q_2\}$
q_2	ϕ	ϕ

② construct an equivalent state table for DFA

	a	b
$\rightarrow [q_0]$	$[q_0 q_1]$	$[\phi]$
$[q_0 q_1]$	$[q_0 q_1]$	$[q_1 q_2]$
$[\phi]$	$[\phi]$	$[\phi]$
$[q_1 q_2]$	$[\phi]$	$[q_1 q_2]$

> Repeat this same step for the newly created state
[Start the process from the initial state of NFA]

$[q_0] \rightarrow$ square bracket is used to identify q_0 as a converted DFA state

$[q_0 q_1]$ on a
$= [q_0$ on 'a' $\cup$ q_0 on 'b']

i) Scan the state table to confirm that there is ~~some~~ transition representation for all newly created state.

Scanned by CamScanner

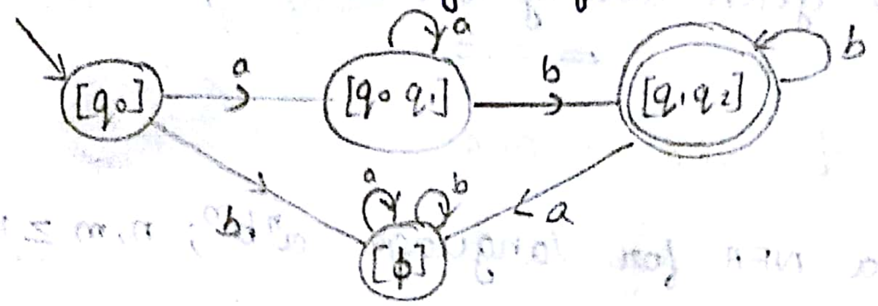
[q1 q2] = [q2 q1]

ii) choose the initial state and final states

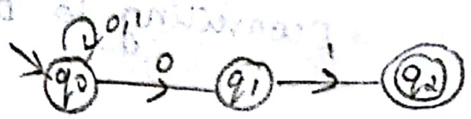
initial state of DFA - is the initial state of NFA -

final state of DFA - ~~combination~~ any state ~~of~~ or combination of any states which contain ~~all the~~ the final states of NFA.

③ Construct the state diagram for DFA



Q) Convert NFA to DFA



Ans

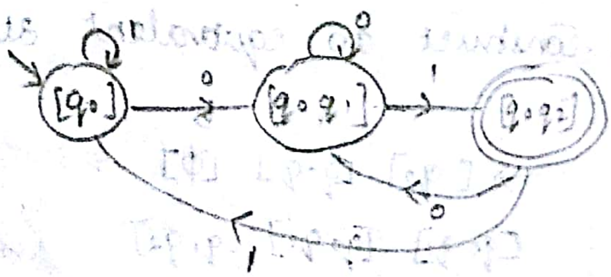
	0	1
→ q_0	$\{q_0 q_1\}$	$\{q_0\}$
q_1	ϕ	$\{q_2\}$
(q_2)	ϕ	ϕ

→ state ~~diagram~~ table for NFA

state table for DFA

	0	1
→ $[q_0]$	$[q_0 q_1]$	$[q_0]$
$[q_0 q_1]$	$[q_0 q_1]$	$[q_0 q_2]$
* $[q_0 q_2]$	$[q_0 q_1]$	$[q_0]$

state diagram

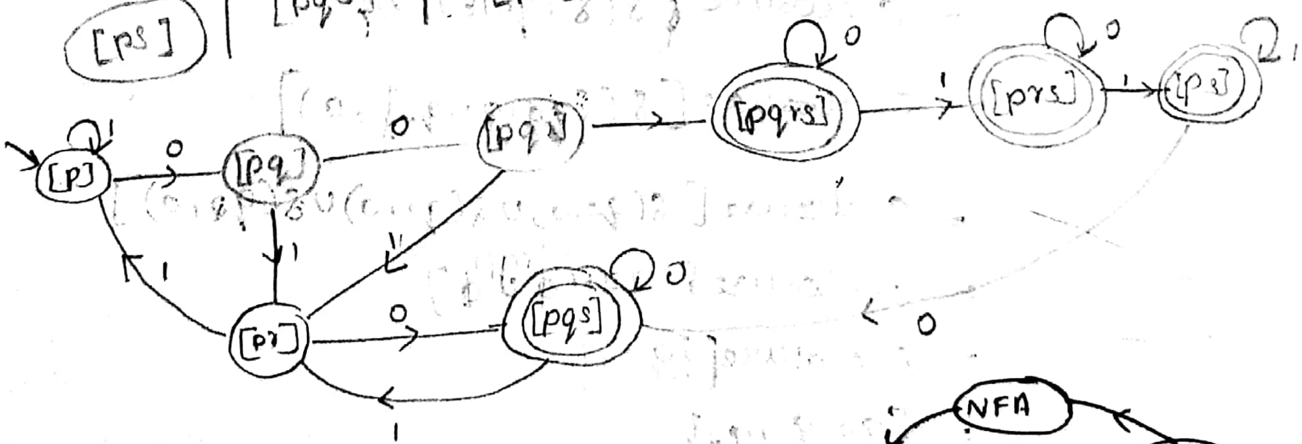


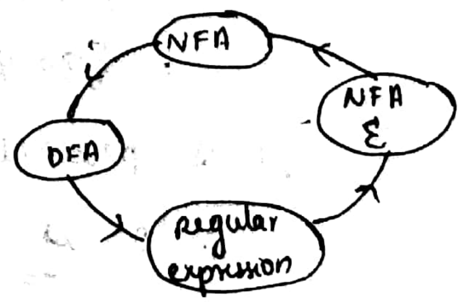
Q) construct state table for DFA from the given state table for NFA

	0	1
→ p	{p,q}	{p}
q	{r}	{r}
r	{s}	φ
(s)	{s}	{s}

Ans

	0	1
→ [p]	[pq]	[p]
[pq]	[pqr]	[pr]
[pqr]	[pqrs]	[pr]
[pr]	[pqs]	[p]
([pqrs])	[pqrs]	[prs]
([pqs])	[pqs]	[pr]
([prs])	[pqs]	[ps]
([ps])	[pqs]	[ps]





Scanned by CamScanner

22/8/17 Tuesday

NFA with Epsilon Transition

ε-closure(q_0)

Set of all states reachable from it just by reading (through ε moves)

$\varepsilon\text{-closure}(q_0) = \{q_0, q_1, q_2\}$

$\varepsilon\text{-closure}(q_1) = \{q_1, q_2\}$

$\varepsilon\text{-closure}(q_2) = \{q_2\}$

$$\boxed{\hat{\delta}(q, \varepsilon) = \varepsilon\text{-closure}(q)}$$

i) $\hat{\delta}(q_0, 0) = \cancel{\delta(\hat{\delta}(q_0,\varepsilon), 0)}$

$= \varepsilon\text{-closure}\{\delta(\hat{\delta}(q_0,\varepsilon), 0)\}$

$= \varepsilon\text{-closure}[\delta(\{q_0, q_1, q_2\}, 0)]$

$= \varepsilon\text{-closure}[\delta(q_0,0) \cup \delta(q_1,0) \cup \delta(q_2,0)]$

$= \varepsilon\text{-closure}[q_0 \cup \phi \cup \phi]$

$= \varepsilon\text{-closure}[\{q_0\}]$

$= \{q_0, q_1, q_2\}$

q_2 is present in F (final state)

∴ '0' is accepted

ii) $\hat{\delta}(q_0, 1) = \varepsilon\text{-closure}\{\delta(\hat{\delta}(q_0, \varepsilon), 1)\}$

$= \varepsilon\text{-closure}\{\delta(\{q_0, q_1, q_2\}, 1)\}$

$= \varepsilon\text{-closure}\{\delta(q_0, 1) \cup \delta(q_1, 1) \cup \delta(q_2, 1)\}$

$= \varepsilon\text{-closure}\{\{q_1\}\}$

$= \{q_1, q_2\}$

q_2 is present in F

$\therefore$ '1' is accepted

Conversion of NFA with ε to NFA without ε

$L(G) = 0^n 1^m 2^p$; $n, m, p \geq 0$

★ Step 1: Findout ε-closure for all states

$\varepsilon\text{-closure}(q_0) = \{q_0, q_1, q_2\}$

$\varepsilon\text{-closure}(q_1) = \{q_1, q_2\}$

$\varepsilon\text{-closure}(q_2) = \{q_2\}$

★ state table for NFA with ε

	0	1	2	ε
q_0	q_0	ϕ	ϕ	q_1
q_1	ϕ	q_1	ϕ	q_2
q_2	ϕ	ϕ	q_2	ϕ

$\hat{\delta}(q_0, 0)$

iii) $\hat{\delta}(q_0, 2) = \varepsilon\text{-closure}[\delta(\hat{\delta}(q_0, \varepsilon), 2)]$

$= \varepsilon\text{-closure}[\delta(\{q_0, q_1, q_2\}), 2)]$

$= \varepsilon\text{-closure}[\{\delta(q_0, 2) \cup \delta(q_1, 2) \cup \delta(q_2, 2)\}]$

$= \varepsilon\text{-closure}[\{\phi \cup \phi \cup q_2\}]$

$= \varepsilon\text{-closure}[\{q_2\}]$

$= \underline{\underline{\{q_2\}}}$

q_2 is present in $\underline{\underline{F}}$

~~8 ... 2 is accep~~

iv) $\hat{\delta}(q_2, 0) = \varepsilon\text{-closure}[\delta(\hat{\delta}(q_2, \varepsilon), 0)]$

$= \varepsilon\text{-closure}[\delta(\{q_2\}, 0)\}$

$= \varepsilon\text{-closure}[d]$

$= \phi$

v) $\hat{\delta}(q_2, 1) = \varepsilon\text{-closure}[\delta(\hat{\delta}(q_2, \varepsilon), 1)]$

$= \varepsilon\text{-closure}[\delta(\{q_2\}, 1]$

$= \underline{\underline{\phi}}$

vi) $\hat{\delta}(q_2, 2) = \varepsilon\text{-closure}[\delta(\hat{\delta}(q_2, \varepsilon), 2)]$

$= \varepsilon\text{-closure}[\delta(q_2, 2)]$

$= \varepsilon\text{-closure}[q_2]$

$= \underline{\underline{\{q_2\}}}$

vii) $\hat{\delta}(q_1, 0) = \varepsilon\text{-closure}[\delta(\hat{\delta}(q_1, \varepsilon), 0)]$

$= \varepsilon\text{-closure}[\delta(\{q_1, q_2\}; 0)]$

$= \varepsilon\text{-closure}[\phi]$

$= \underline{\underline{\phi}}$

viii) $\hat{\delta}(q_1, 1) = \varepsilon\text{-closure}\,[\delta(\hat{\delta}(q_1, \varepsilon), 1)]$

$= \varepsilon\text{-closure}\,[\delta(\{q_1, q_2\}, 1)]$

$= \varepsilon\text{-closure}\,[q_1]$

$= \{q_1, q_2\}$

ix) $\hat{\delta}(q_1, 2) = \varepsilon\text{-closure}\,[\delta(\hat{\delta}(q_1, \varepsilon), 2)]$

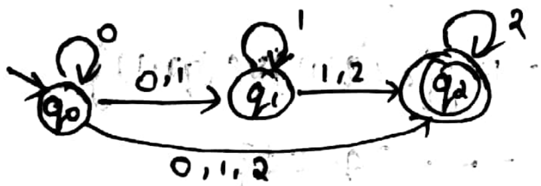
$= \varepsilon\text{-closure}\,[\delta(\{q_1, q_2\}, 2)]$

$= \varepsilon\text{-closure}\,(q_2)$

$= \{q_2\}$

state table for NFA without ε

	0	1	2
q_0	$\{q_0, q_1, q_2\}$	$\{q_1, q_2\}$	$\{q_2\}$
q_1	ϕ	$\{q_1, q_2\}$	$\{q_2\}$
q_2	ϕ	ϕ	$\{q_2\}$



Q) $\hat{\delta}(q_0, 01) = \delta(\hat{\delta}(q_0, 0), 1)\,]$

$[\hat{\delta}(q_0, 0) = \{q_0, q_1, q_2\}]$

$= [\delta(\{q_0, q_1, q_2\}, 1]$

$= [\{\delta(q_0, 1) \cup \delta(q_1, 1) \cup \delta(q_2, 1)\}]$

$= \{\phi \cup q_1 \cup \phi\}$

$= \{q_1\}]$

$= \{q_1, q_2\}$

Q) Check if 011 & 012 is present in the language $0^n 1^m 2^p ; n,m,p \geq$

i) $\hat{\delta}(q_0, 011) = E.C\, \delta(\hat{\delta}(q_0, 01), 1)$

$\hat{\delta}(q_0, 01) = E.C\, \delta(\hat{\delta}(q_0, 0), 1)$

$\hat{\delta}(q_0, 0) = E.C \{\hat{\delta}(q_0, 0)\}$

$= E.C \{\delta(\hat{\delta}(q_0, \epsilon), 0)\}$

$= \{q_0, q_1, q_2\}$

$\hat{\delta}(q_0, 01) = E.C\, \delta(\{q_0, q_1, q_2\}, 1)$

$= E.C \{\delta(q_0, 1) \cup \delta(q_1, 1) \cup \delta(q_2, 1)\}$

$= E.C \{\phi \cup q_1 \cup \phi\}$

$= E.C\{q_1\} = \{q_1, q_2\}$

$\hat{\delta}(q_0, 011) = \delta(\hat{\delta}(q_0, 01), 1) \quad E.C\, \delta(\{q_1, q_2\}, 1)$

$= \delta(\{q_1\}, 1) \quad = E.C \{\delta(q_1, 1) \cup \delta(q_2, 1)\}$

$= \delta\{q_1\} \quad = E.C \{q_1 \cup \phi\}$

$= \{q_1, q_2\}$

$\therefore$ 011 is present in the language

ii) $\hat{\delta}(q_0, 012) = E.C [\delta(\hat{\delta}(q_0, 01), 2)]$

$\hat{\delta}(q_0, 01) = E.C [\delta(\hat{\delta}(q_0, 0), 1)]$

$\hat{\delta}(q_0, 0) = E.C \{\delta(q_0, 0)\}$

$= \{q_0, q_1, q_2\}$

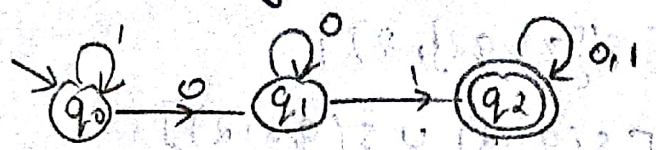
$$\hat{\delta}(q_0, 01) = E\cdot C\,[\delta(\{q_0, q_1, q_2\}, 1)]$$
$$= E\cdot C\,[\{q_1\}]$$
$$= \{q_1, q_2\}$$

$$\hat{\delta}(q_0, 012) = E\cdot C\,[\delta(\{q_1, q_2\}, 2)]$$
$$= E\cdot C\,[\delta(q_1, 2) \cup \delta(q_2, 2)]$$
$$= E\cdot C\,[\phi \cup q_2]$$
$$= E\cdot C\,[q_2]$$
$$= \{q_2\}$$

OPEN BOOK TEST QUESTIONS

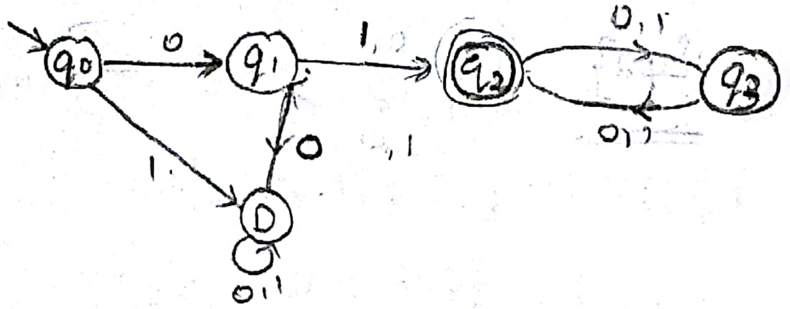
1) Design a DFA that accept all strings

(a) with substring 01



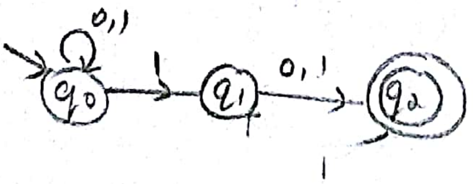
(b) of even length and begin with 01

0100



2. Design an NFA accepting the set of all strings

(a) whose second last symbol is 1



(b) End with substring 11

3. (a)

δ	0	1
$\rightarrow q_0$	q_1	q_4
q_1	q_4	q_2
q_2	q_3	q_3
q_3	q_2	q_2
q_4	q_4	q_4

Check string 011101 in DFA (Accept or Not) ?

As

$\hat{\delta}(q_0, 011101) = \delta(\hat{\delta}(q_0, 01110), 1))$

$= \delta(\delta(\delta(\hat{\delta}(q_0, 0111), 0), 1))$

$= \delta(\delta(\delta(\hat{\delta}(q_0, 011), 1), 0), 1))$

$= \delta(\delta(\delta(\delta(\hat{\delta}(q_0, 01), 1), 1), 0), 1))$

$= \delta(\delta(\delta(\delta(\delta(\hat{\delta}(q_0, 0), 1), 1), 1), 0), 1))$

$= \delta(\delta(\delta(\delta(\delta(q_1, 1), 1), 1), 0), 1))$

$= \delta(\delta(\delta(\delta(q_2, 1), 1), 0), 1))$

$= \delta(\delta(\delta(q_3, 1), 0), 1)$

$= \delta(\delta(q_2, 0), 1)$

$= \delta(q_3, 1)$

$= q_2$

$\hat{\delta}(q_0, 0)$
$= \delta(\hat{\delta}(q_0, \varepsilon), 0)$
$= \delta(q_0, 0)$
$= q_1$

$q_2 \in F$

∴ the string 011101 is accepted

3(b)

δ	0	1
q_0	q_0	$\{q_1, q_2\}$
q_1	q_2	q_2
q_2	ϕ	ϕ

check the string 01010 (Accept/Not)?

As $\hat{\delta}(q_0, 01010) = \delta(\hat{\delta}(q_0, 0101), 0)$

$= \delta(\delta(\hat{\delta}(q_0, 010), 1), 0)$

$= \delta(\delta(\delta(\hat{\delta}(q_0, 01), 0), 1), 0)$

$= \delta(\delta(\delta(\delta(\hat{\delta}(q_0, 0), 1), 0), 1), 0)$

$= \delta(\delta(\delta(\delta(q_0, 1), 0), 1), 0)$

$= \delta(\delta(\delta(\{q_1, q_2\}, 0), 1), 0)$

$= \delta(\delta(\delta(q_1, 0) \cup \delta(q_2, 0), 1), 0)$

$= \delta(\delta(q_2, 1), 0)$

$= \delta(\phi, 0)$

$= \phi$

$\hat{\delta}(q_0, 0)$

$= \delta(\hat{\delta}(q_0, \epsilon), 0)$

$= \delta(q_0, 0)$

$= q_0$

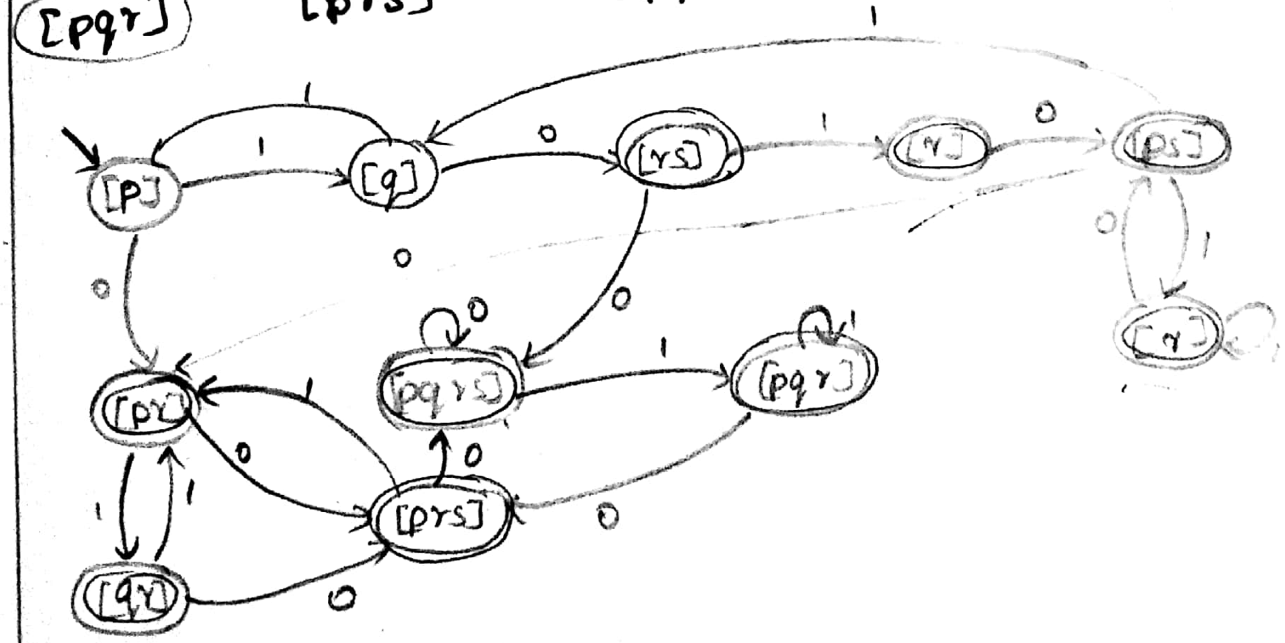
$\therefore$ the string 01010 is not accepted

4) convert to DFA the following NFA

δ	0	1
$\rightarrow p$	$\{p, r\}$	$\{q\}$
q	$\{r, s\}$	$\{p\}$
r	$\{p, s\}$	$\{r\}$
s	$\{q, r\}$	ϕ

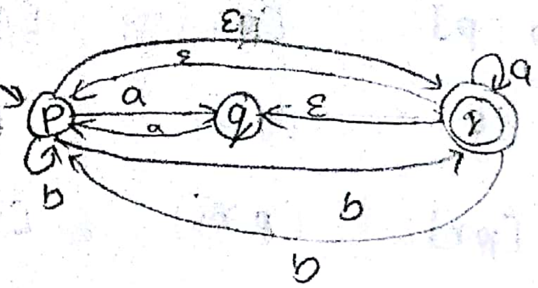
State table for DFA

δ	0	1
→ [p]	[p r]	[q]
[q]	[r s]	[p]
[p r]	[p r s]	[q r]
[r s]	[p s q r]	[r]
[p r s]	[p r s q]	[p q r]
[q r]	[r s p]	[p r]
[p q r s]	[p r s q]	[p q r]
[r]	[p s]	[r]
[p s]	[p q r]	[q]
[p q r]	[p r s]	[p q r]



5) Consider the NFA-ε

δ	ε	a	b
→ p	{r}	{q}	{p,r}
q	φ	{p}	φ
(r)	{p,q}	{r}	{p}



(a) Compute the ε-closure of each state

ε-closure of p = {p, r, q}.

ε-closure of q = {q}

ε-closure of r = {r, q, p}.

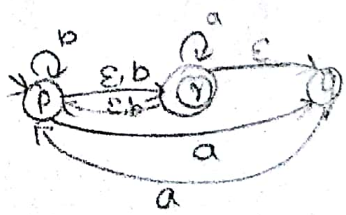
(b) Give the set of all strings of length 3 or less accepted by the automata

[b, a]

[b a, bb]

[a a b, bbb

b b a, baa]



(c) Convert the automata to DFA

$\hat{\delta}(p,a) = \varepsilon\text{-closure}[\delta(\hat{\delta}(p,\varepsilon),a)]$

$= \varepsilon\text{-closure}[\delta(\{p,r,q\},a)]$

$= \varepsilon\text{-closure}[\{\delta(p,a) \cup \delta(r,a) \cup \delta(q,a)\}]$

$= \varepsilon\text{-closure}[\{q,r,p\}]$

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

$\hat{\delta}(p,b) = \varepsilon\text{-closure}[\delta(\hat{\delta}(p,\varepsilon),b]$

$= \varepsilon\text{-closure}[\delta(\{p,q,r\}),b]$

$= \varepsilon\text{-closure}[\{p,r\}]$

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

$\hat{\delta}(q,a) = \varepsilon\text{-closure}[\delta(\hat{\delta}(q,\varepsilon),a)]$

$= \varepsilon\text{-closure}[\delta(\{q\},a)]$

$= \varepsilon\text{-closure}[\{p\}]$

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

$\hat{\delta}(q,b) = \varepsilon\text{-closure}[\delta(\hat{\delta}(q,\varepsilon),b)]$

$= \varepsilon\text{-closure}[\delta(\{q\},b)]$

$= \varepsilon\text{-closure}[\phi]$

$= \phi$

$\hat{\delta}(r,a) = \varepsilon\text{-closure}[\delta(\hat{\delta}(r,\varepsilon),a)]$

$= \varepsilon\text{-closure}(\delta(\{p,q,r\},a)]$

$= \varepsilon\text{-closure}[\{p,q,r\}$

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

$\hat{\delta}(r,b) = \varepsilon\text{-closure}[\delta(\hat{\delta}(r,\varepsilon),b)]$

$= \varepsilon\text{-closure}[\delta(\{p,q,r\},b]]$

$= \varepsilon\text{-C}[\{p,r\}]$

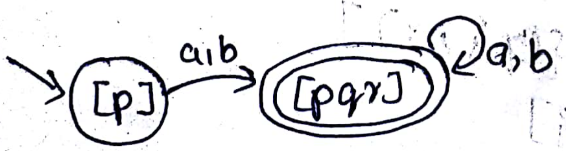
$= \{p,q,r\}$

State table of NFA without ε

δ	a	b
→ p	{p,q,r}	{p,q,r}
q	{p,q,r}	ϕ
(r)	{p,q,r}	{p,q,r}

State table of DFA

δ	a	b
[p]	[pqr]	[pqr]
[pqr]	[pqr]	[pqr]



26/8/17 Saturday

Regular Expression

Three primitive expressions

1. ϕ
2. ε
3. a

⇒ +, · & *

⇒ '+' denotes union
⇒ '·' denotes contatenation
⇒ '*' denotes closure

new expressions can be formed from primitive expressions using these operators

Q) L_1, L_2

$L_1 = \{001, 10, 111\}$

$L_2 = \{\varepsilon, 001\}$

- $L_1 + L_2 = \{001, 10, 111, \varepsilon\} = L_3$
- $L_1 \cdot L_2 = \{001, 10, 111, 001001, 10001, 111001$
-

Q) $L_1 = \{0, 11\}$

$L_1^* = \{\varepsilon, 0, 00, 11, 000, 011, 110, 0000, 0011, 1111, 1100, 0110, \ldots\}$

→ $(0+1)^* = \{\varepsilon, 0, 1, 00, 11, 01, 10, 000, 001, 010, 011, 100, 101, 110, 111 \ldots\}$

↳ equivalent to $\Sigma = \{0, 1\}$ then finding Σ^*

Scanned by CamScanner

$L_1^* = L_1^0 \cup L_1^1 \cup L_1^2 \cup L_1^3 \ldots$

⇒ $0(0+1)^*$

↳ all strings that starts with zero

⇒ strings of zero and one that ends with 11

$(0+1)^* 11$

⇒ Reg exp that denotes all strings that contains atleast one consecutive 1

$(0+1)^* 11 (0+1)^*$

⇒ set of all strings which have atleast 1 consecutive pairs of 1's or consecutive pairs of 0's

$(0+1)^*(11+00)(0+1)^*$

⇒ set of strings that end with 11 or 00

$(0+1)^*(11+00)$

⇒ set of all strings which no two consecutive 1's

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

29/8/17
Tuesday

Q) set of all strings from $0^* 1 + 1^* 0$

$= \{1, 0, 01, 10, 001, 110, 0001, 1110 \ldots\}$

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

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

Q) $(0+1)^* . 0$ $= \{0, 00, 10, 000, 010, 100, 110 \ldots\}$

(set of all strings that end with zero

Precedence

$0^* 1 + 0 \neq 0^*(1+0)$

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

$L_1 = (0+1)^* = \{\varepsilon, 0, 1, 00, 01, 10, 11, 000 \ldots\}$

$L_2 = \{1\}$

$L_3 = L_1 \cdot L_2 = \{01, 11, 001, 011, 101, 111, \ldots\}$

$L_4 = (L_1 \cdot L_2)^* = (L_3)^*$ ~~$= \{\varepsilon, 1, 01, 11, 10, 00$~~

$= \{\varepsilon, 1, 01, 11, 001, 011, 101, 111$

How to convert a regular expression ~~with~~ into a NFA-ε

Let Σ be a given alphabet, then

1) $\phi, \varepsilon, a \in \Sigma$ are all regular expressions. These are called primitive regular expression

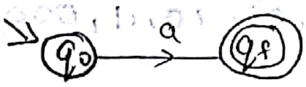
2) If R_1, R_2 are regular expression, then so are $R_1 + R_2$, R_1^*, $R_1 \cdot R_2$,

3) A string is a regular expression, if and only if it can be derived from the primitive regular expression by finite number of applications of the rules mentioned in 2)

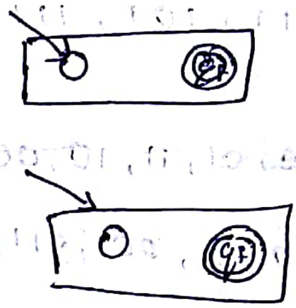
1) ϕ

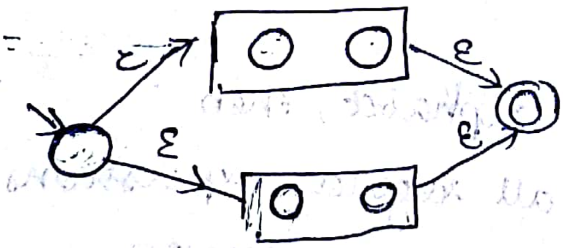
2) ε

3) $a \in \Sigma$

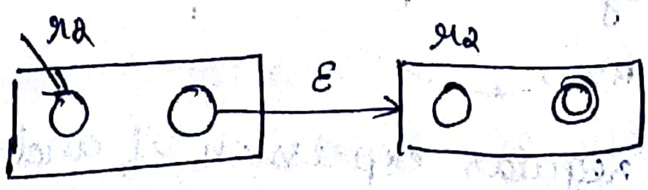


★ Union of $(r_1 + r_2)$

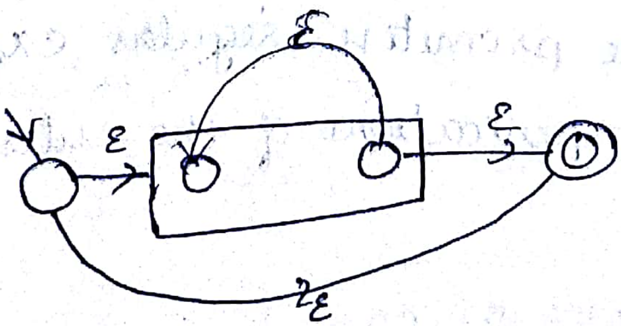




★ $r_2 r_1$

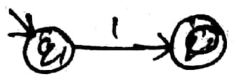


★



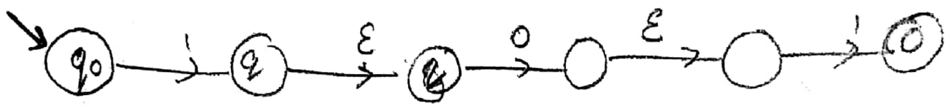
Q) Convert the following NFA-ε to regular expressions.

1) 01 + 101

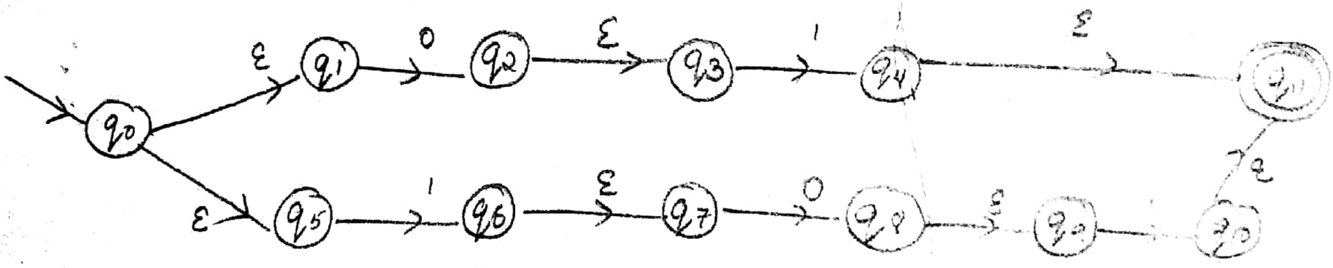


⇒ 01

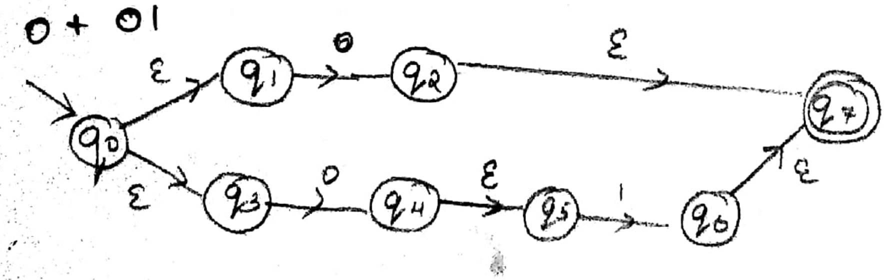
101



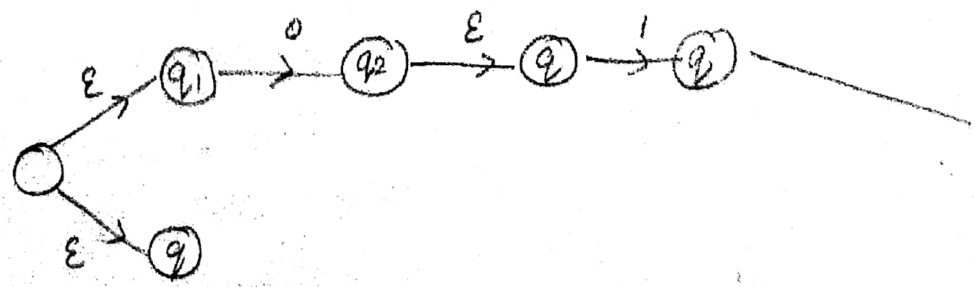
01+101



ii) 0 + 01

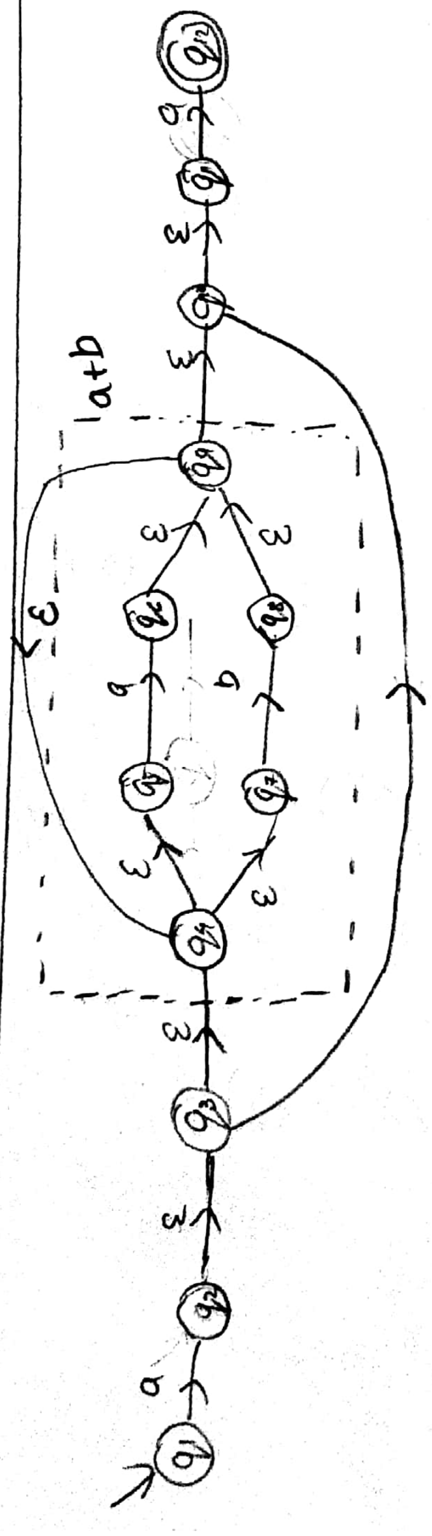


iii) (01 + 101)*



30/8/19

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



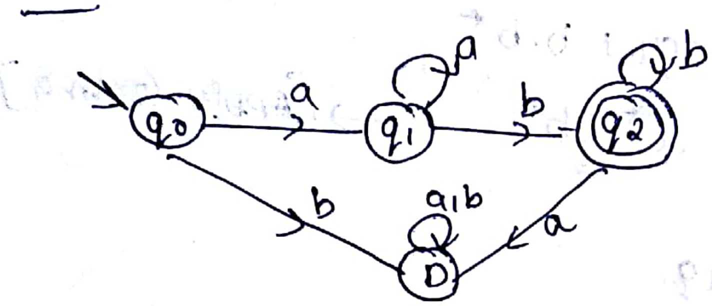
$b \cdot \phi = \phi$
$b + \phi = b$

DFA to regular expression

Lemma

If $\varepsilon \notin A$, then the expression $X = AX + B$ has a unique solution $X = A^* B$

(Q)



step 1: form equation for each state.

$$q_0 = a \cdot q_1 + bD$$

$$q_1 = a q_1 + b q_2$$

λ - denotes empty string

$$q_2 = aD + bq_2 + \lambda$$

to signify q_2 is final state

$$D = aD + bD$$

step 2: done using lemma form

$$D = (a+b)D$$

$$= (a+b)D + \phi \quad \text{- of lemma form}$$

$$\therefore D = (a+b)^* \cdot \phi = \phi$$

$\therefore$ the equations reduces to

$$q_0 = a q_1 + b \cdot \phi = a q_1$$

$$q_1 = a q_1 + b q_2$$

$$q_2 = a\phi + bq_2 + \lambda = bq_2 + \lambda$$

Take $q_2 = bq_2 + \lambda$

$q_2 = b^* \lambda = b^*$ [Apply lemma]

$q_0 = aq_1$

$q_1 = aq_1 + bq_2$

$= aq_1 + b.b^*$

$= a^*.bb^*$ → [Apply lemma]

$q_0 = aq_1$

$= a.a^*.bb^*$

regular expression : aa^*bb^*

NOTE
q_0 will give you the required regular expression.

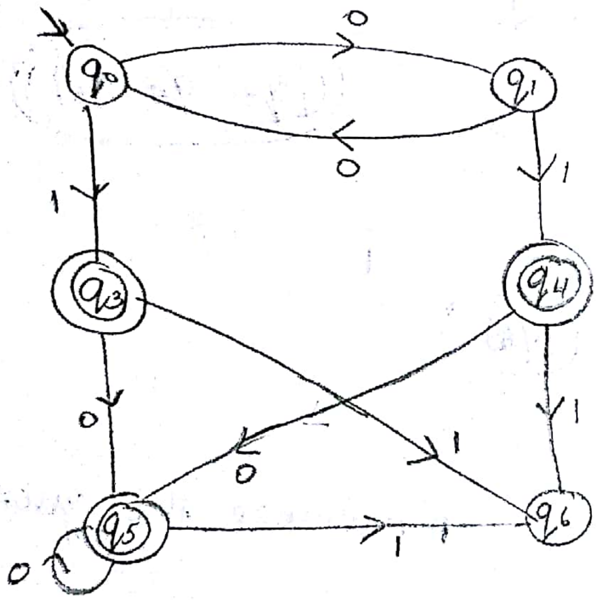
Scanned by CamScanner

10/8/17

Minimum State Automata

⇒ Two states p and q are equivalent if and only if, for ~~every~~ any string x in Σ^* $\hat{\delta}(p,x)$ and $\hat{\delta}(q,x)$ and both in F or not both in F.

⇒ p and q are said to distinguishable if there exist a string x, such that $\hat{\delta}(p,x) \in F$ and $\hat{\delta}(q,x) \notin F$ or vice-versa.



Step 1

Group the non-final and final states

$[q_0 \; q_1 \; q_6]$ $[q_3 \; q_4 \; q_5]$

→ Taking (q_0, q_1)

$\delta(q_0, 0) = q_1$ NF

$\delta(q_1, 0) = q_3$ F

$\delta(q_0, 1) = q_0$ NF

$\delta(q_1, 1) = q_4$ F

q_0 and q_1 are equivalent.

Scanned by CamScanner

Take (q_0, q_6)

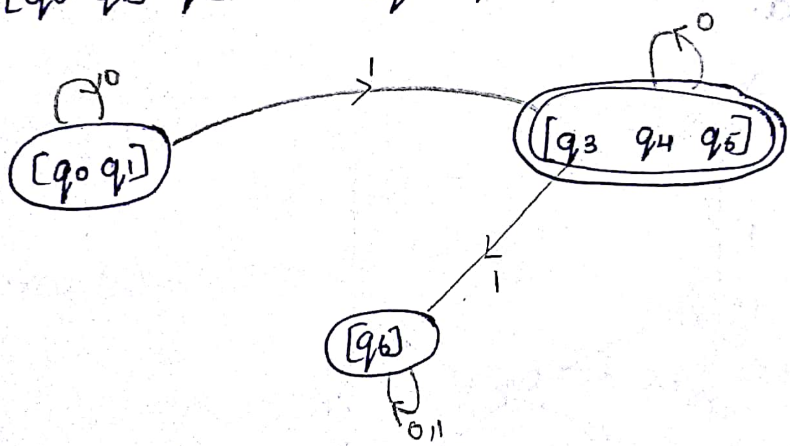
$\delta(q_0, 0) = q_1$ NF $\quad \delta(q_0, 1) = q_6$ NF

$\delta(q_6, 0) = q_3$ F $\quad \delta(q_6, 1) = q_6$ NF

q_0, q_6 are distinguishable

Hence breaking the partition

$[q_0\ q_1]\ [q_6] \quad [q_3\ q_4\ q_5]$



Q) Minimize the given DFA

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

[A B D E F G H] [C]

→ A B

$\delta(A,0)=B$ $\delta(B,0)=G$

$\delta(A,1)=F$ $\delta(B,1)=C$

not equivalent

→ A D X

→ A E ✓ $\delta(A,0)=B$ $\delta(A,1)=F$ ~~$\delta(A,01)$~~

$\delta(E,0)=H$ $\delta(E,1)=F$

→ ~~AF~~ X

$\delta(A,01)=C$ $\delta(E,01)=C$

$\delta(A,10)=C$ $\delta(E,10)=C$

$\delta(A,11)=G$ $\delta(E,11)=G$

$\delta(A,00)=G$ $\delta(E,00)=G$

→ A F X

→ A G X

$\delta(A,0)=B$ $\delta(G,0)=G$

$\delta(A,1)=F$ $\delta(G,1)=E$

$\delta(A,01)=C$ $\delta(G,01)=E$

G E $\delta(G,0)=G$ $\delta(E,0)=H$

$\delta(G,1)=E$ $\delta(E,1)=F$

$\delta(G,01)=E$ $\delta(E,01)=C$

X

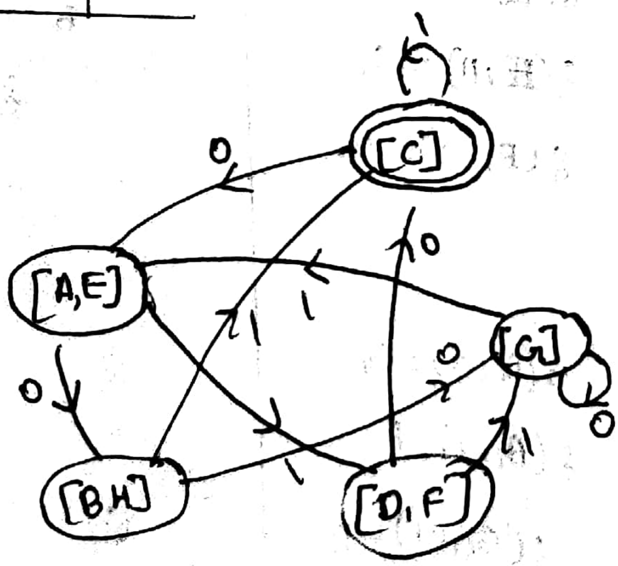
BD =

~~E~~ $\delta(E$

	A	B	C	D	E	F	G
B	X						
C	X	X					
D	X	X	X				
E		X	X	X			
F	X	X	X		X		
G	X	X	X	X	X	X	
H	X		X	X	X	X	X

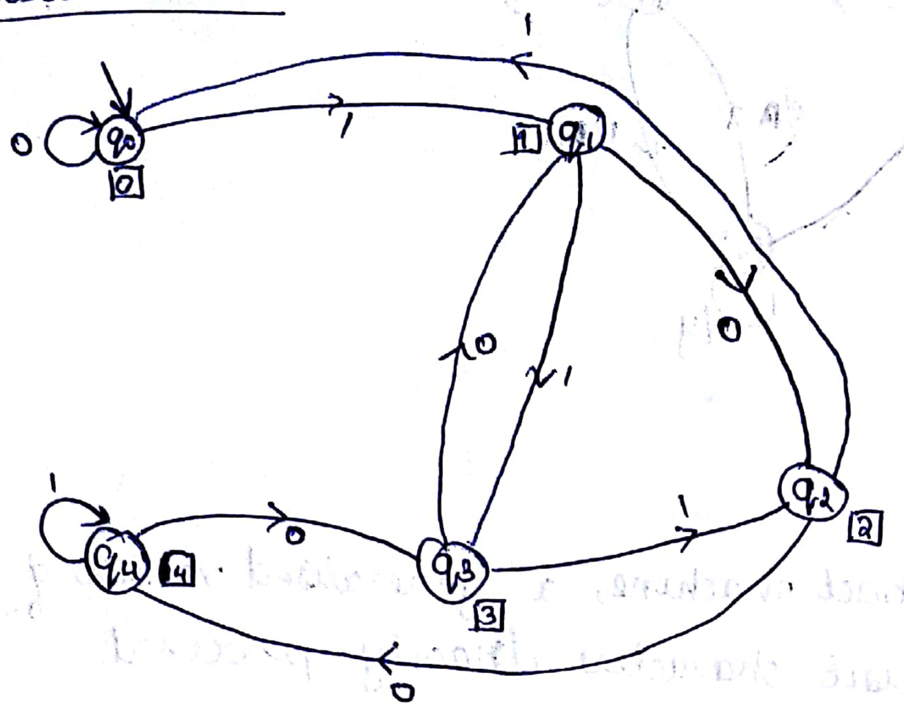
[A,E] [B,H] [D,F] [G] [C]

	0	1
[A,E]	[B,H]	[F]
[C]	[A]	[C]
[B,H]	[G]	[C]
[D,F]	[C]	[G]
[G]	[G]	[E]



Moore machine

2, 5, 7, 12, 10



2 - o/p state $\boxed{2}$

5 - $\boxed{0}$

7 - $\boxed{2}$

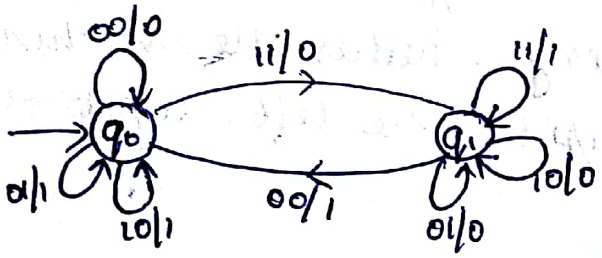
12 - $\boxed{2}$

B

1101

1010

Mealy machine



Formal description - FSA with o/p

$$M = (K, \Sigma, \Delta, \delta, \lambda, q_0)$$

Q/ K - finite set of states

Σ - finite set of inputs

Δ = o/p symbol

δ - $K \times \Sigma$

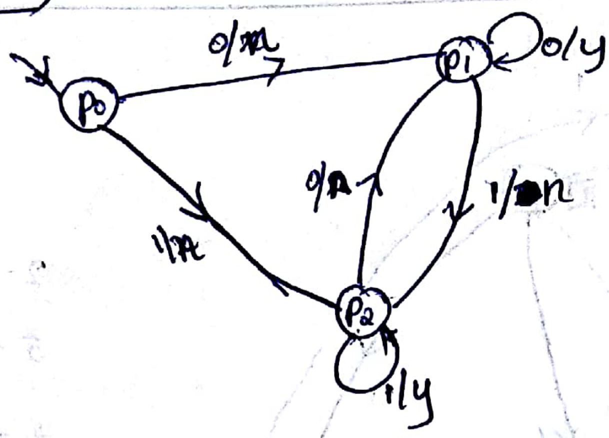
λ - o/p function

q_0 - initial state

moore - $\lambda : K \to \Delta$

mealy - $\lambda : K \times \Sigma \to \Delta$

Mealy



Two

It is an abstract machine, a generalised version of DFA which can revisit characters already processed.

⇒ There are finite number of states with transitions b/w them based on the current i/p. But each transition is also labeled with a value indicating whether the machine will move its position in the i/p to the left, right or stay at the same position

$$M = (Q, \Sigma, \vdash, \dashv, \delta, s, t, r)$$

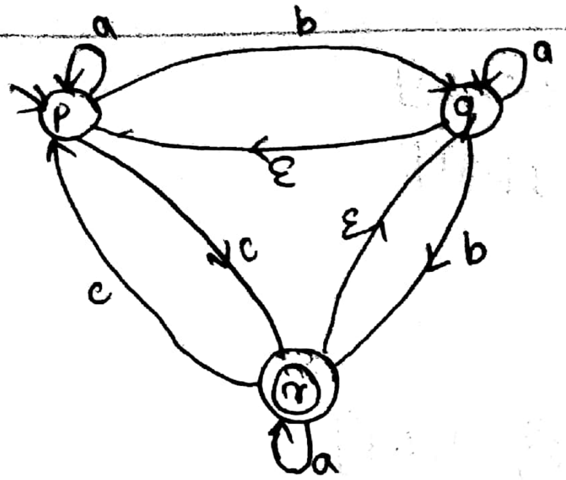
left & right end markers

s - start state

t - end state

r - reject state

In addition to this, two conditions should be satisfied



ε-closure $(p) = \{p\}$

ε-closure of $q = \{q, p\}$

ε-closure of $r = \{p, q, r\}$

$$\hat{\delta}(p,a) = \varepsilon.c \text{ of } [\delta(\hat{\delta}(p,\varepsilon),a)]$$
$$= \varepsilon.c \text{ of } [\delta(\{p\},a)]$$
$$= \varepsilon.c \text{ of } [p]$$
$$= \{p\}$$

$$\hat{\delta}(p,b) = \varepsilon\text{-}c \text{ of } [\delta(\hat{\delta}(p,\varepsilon),b)]$$
$$= \varepsilon.c \text{ of } [\delta(\{p\},b)]$$
$$= \varepsilon.c \text{ of } \{q\}$$
$$= \{p,q\}$$

$$\hat{\delta}(p,c) = \varepsilon\text{-}c \text{ of } [\delta(\hat{\delta}(p,\varepsilon),c)]$$
$$= \varepsilon\text{-}c \text{ of } [\delta(\{p\},c)]$$
$$= \varepsilon\text{-}c \text{ of } r$$
$$= \{p,q,r\}$$

$$\hat{\delta}(q,a) = \varepsilon\text{-}c \cdot q[\delta(\hat{\delta}(q,\varepsilon),a)]$$
$$= \varepsilon\text{-}c\, q[\delta(\{p,q\},a)]$$
$$= \varepsilon\text{-}c\, q\{p,q\}$$
$$= \{p,q\}$$

$$\hat{\delta}(q,b) = \varepsilon\text{-}c\, q[\delta(\hat{\delta}(q,\varepsilon),b)]$$
$$= \varepsilon\text{-}c\, q(\delta(\{p,q\},b)]$$
$$= \varepsilon\text{-}c\, \{q,r]$$
$$= \{p,q,r\}$$

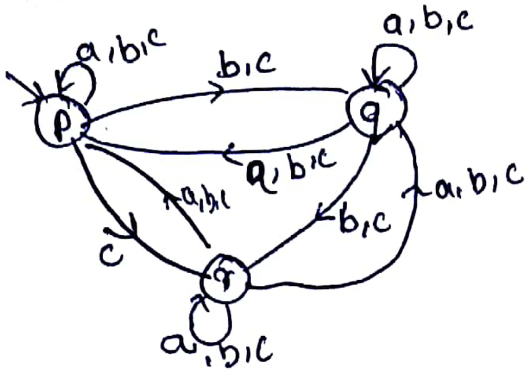
$$\hat{\delta}(q,c) = \varepsilon\text{-}c\, q\, [\delta(\hat{\delta}(q,\varepsilon),c)]$$
$$= \varepsilon c\, q[\delta(\{p,q\},c)]$$
$$= \varepsilon\text{-}c\, q\{\{r\}$$
$$= \{p,q,r\}$$

$$\hat{\delta}(r,a) = \varepsilon\text{-}c\, q[\delta(\hat{\delta}(q,\varepsilon),a)]$$
$$= \varepsilon\text{-}c\, q[\delta(\{p,q,r\},a)]$$
$$= \varepsilon\text{-}c\, q[\{p,q,r\}]$$
$$= \{p,q,r\}$$

$$\hat{\delta}(r,b) = \varepsilon\text{-}c\, q[\delta(\hat{\delta}(r,\varepsilon),b)]$$
$$= \varepsilon\text{-}c \cdot q[\delta\{p,q,r\},b]$$
$$= \varepsilon\text{-}c\, q[\{p,r\}]$$
$$= \{p,q,r\}$$

$$\hat{\delta}(r,c) = \varepsilon\text{-}c\, q[\delta(\hat{\delta}(r,\varepsilon),c)]$$
$$= \varepsilon c\, q[\delta(\{p,q,r\},c)]$$
$$= \varepsilon\text{-}c\, q[\{r,p\}] \quad = \{p,q,r\}$$

	a	b	c
→ p	{p}	{p,q}	{p,q,r}
q	{p,q}	{p,q,r}	{p,q,r}
* r	{p,q,r}	{p,q,r}	{p,q,r}



NFA with ε to DFA conversion

→ Start with initial state

→ Take ε closure

→ Apply δ_n on all inputs → set of states → take ε closure

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