TCS

Theoretical Computer Science

- Finite Automata (FA)
- Push Down Automata (PDA)
- Turing machine (TM)
- Linear Bound Automata (LBA) X Syllabus

Finite State Machine (FSM):

① Transition Diagram (DAG) (Directed Acyclic Graph)

② Tuples

③ 2 examples (Accept/Reject)

- Finite Automation is a mathematical model with discrete inputs & o/p.
- It is represented with TD.
- The vertices of DAG are called states & edges are transitions.

Defⁿ: Mathematically, FA is defined as a five tuple collection $(\Sigma, Q, F, \delta, q_0)$

Σ = Finite set of i/p signals.
Q = " " " states.
F = " " " final states.
δ = Transition mapping func defined as $(Q \times \Sigma) \to Q$
q_0 = Initial state.

*) Design FA to accept binary strings ending with '01'.

⇒ Tuple = (Σ, Q, F, δ, q_0)

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

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

$F = \{ q_3 \}$

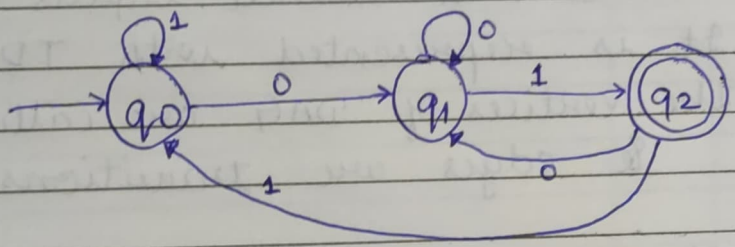
$q_0 = q_0$

δ	0	1
ε	q_0	q_1 q_0
0	q_1	q_1 q_2
01	q_2	q_1 q_0

$\varepsilon 0$ $\varepsilon 1$

$\emptyset 0$ 01

$\emptyset 10$ $\emptyset 11$



Eg.

1101

$\delta(q_0, 1) = q_0$

$\delta(q_0, 1) = q_0$

$\delta(q_0, 0) = q_1$

$\delta(q_1, 1) = q_2$

$\because q_2 \in F$

$\therefore$ Accept

1000

$\delta(q_0, 1) = q_0$

$\delta(q_0, 0) = q_1$

$\delta(q_1, 0) = q_1$

$\delta(q_1, 0) = q_1$

$\because q_1 \notin F$

$\therefore$ Reject.

*) Design FSM for set of binary strings (a) ending with '10'.

⇒ Tuple = $(\Sigma, Q, F, \delta, q_0)$

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

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

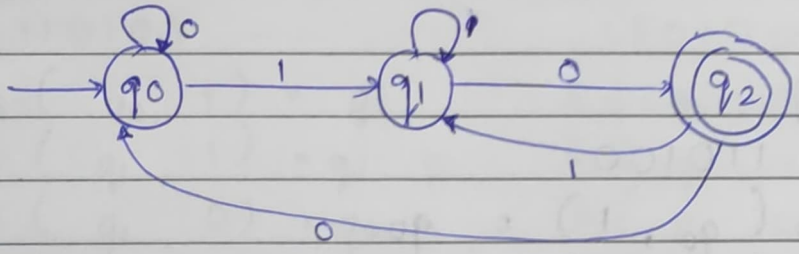
$F = \{q_2\}$

$q_0 = q_0$

$\delta =$

		0	1
Σ	q_0	q_0	q_1
1	q_1	q_2	q_1
10	q_2	q_0	q_1

ΣΣ ΣΣ1
10 Σ1
ΣΣΣ ΣΣ1



Eg.

1110

$\delta(q_0, 1) = q_1$

$\delta(q_1, 1) = q_1$

$\delta(q_1, 1) = q_1$

$\delta(q_1, 0) = q_2$

$\because q_2 \in F$

$\therefore$ Accept.

00001

$\delta(q_0, 0) = q_0$

$\delta(q_0, 0) = q_0$

$\delta(q_0, 0) = q_0$

$\delta(q_0, 1) = q_1$

$\because q_1 \notin F$

$\therefore$ Reject.

b) Ending with 00

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

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

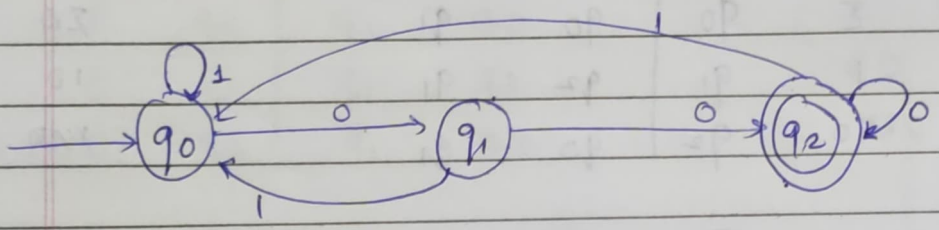
$Q = \{q_0, q_1, q_2\}$

$F = \{q_2\}$

$q_0 = q_0$

δ	0	1
$\in$ q_0	q_1	q_0
0 q_1	q_2	q_0
00 q_2	q_2	q_0

$\Sigma 0$	$\Sigma 1$
00	01
ϕ00	$\phi\phi$1



Eg.

110100

$\delta(q_0, 1) = q_0$

$\delta(q_0, 1) = q_0$

$\delta(q_0, 0) = q_1$

$\delta(q_1, 1) = q_0$

$\delta(q_0, 0) = q_1$

$\delta(q_1, 0) = q_2$

$\because q_2 \in F$

$\therefore$ Accept.

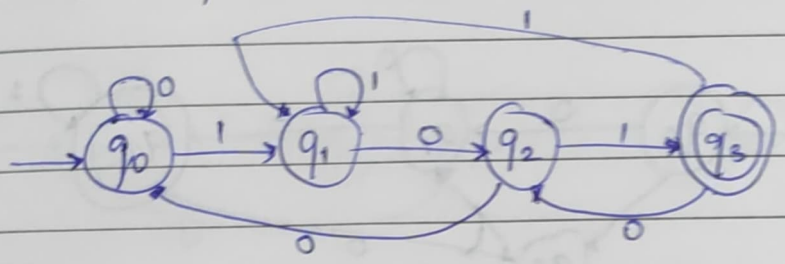
c) Finding with 101

$\Sigma = \{0, 1\}$ $F = \{q_3\}$

$Q = \{q_0, q_1, q_2, q_3\}$ $q_0 = q_0$

δ		0	1
Σ	q_0	q_0	q_1
1	q_1	q_2	q_1
10	q_2	q_0	q_3
101	q_3	q_2	q_1

$\Sigma 0$ $\Sigma 1$
10 $\lambda 1$
100 101
1010 1001



Eg.

110101

$\delta(q_0, 1) = q_1$
$\delta(q_1, 1) = q_1$
$\delta(q_1, 0) = q_2$
$\delta(q_2, 1) = q_3$
$\delta(q_3, 0) = q_2$
$\delta(q_2, 1) = q_3$

$\because q_3 \in F$
$\therefore$ Accept.

110100

$\delta(q_0, 1) = q_1$
$\delta(q_1, 1) = q_1$
$\delta(q_1, 0) = q_2$
$\delta(q_2, 1) = q_3$
$\delta(q_3, 0) = q_2$
$\delta(q_2, 0) = q_0$

$\because q_0 \notin F$
$\therefore$ Reject.

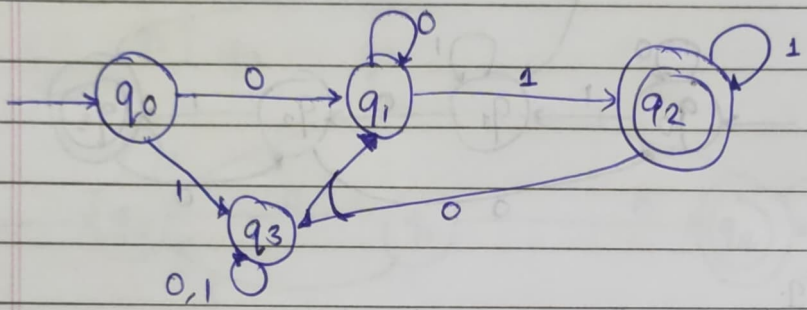
21/07

Design FSM starting with 0 ending 1.

$\Rightarrow \Sigma = \{0, 1\}$ $\quad F = \{q_2\}$

$Q = \{q_1, q_0, q_0, q_3\}$ $\quad q_0 = q_0$

	δ	0	1
Σ	q_0	q_1	q_3
0	q_1	q_1	q_2
01	q_2	q_{01}	q_2
Reject.	q_3	q_3	q_3



Eg. 01101

$\delta(q_0, 0) = q_1$

$\delta(q_1, 1) = q_2$

$\delta(q_2, 1) = q_2$

$\delta(q_2, 0) = q_1$

$\delta(q_1, 1) = q_2$

$\because q_2 \in F$

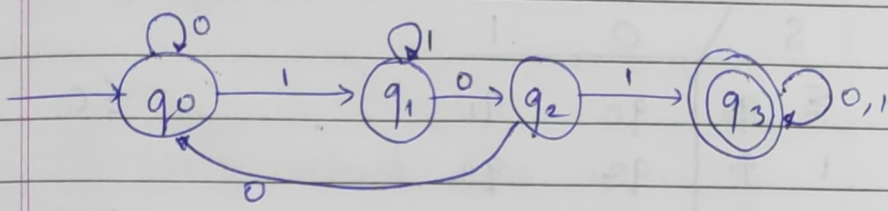
$\therefore$ Accept.

Design FSM for substring '101'.

$\Sigma = \{0, 1\}$ $F = \{q_3\}$ $q_0 = q_0$

$Q = \{q_0, q_1, q_2, q_3\}$

δ		0	1	
ε	q_0	q_0	q_1	$\varepsilon 0$ $\varepsilon 1$
1	q_1	q_2	q_1	
10	q_2	q_0	q_3	~~100~~ 101
101	q_3	q_3	q_3	~~1010~~ ~~1011~~

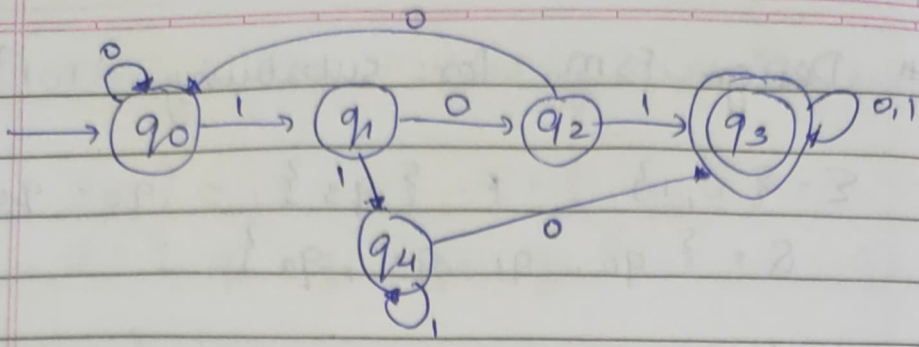


Design FSM for accepting substring '101' or '110'

$\Sigma = \{0, 1\}$ $F = \{q_3\}$ $q_0 = q_0$

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

δ		0	1
ε	q_0	q_0	q_1
1	q_1	q_2	q_4
10	q_2	q_0	q_3
101	q_3	q_3	q_3
110	q_4	q_3	q_4

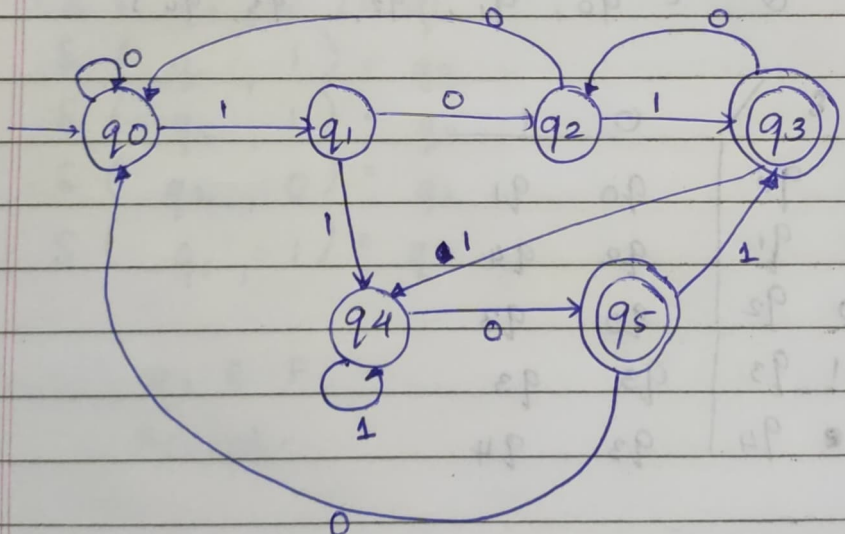


*) Design FSM ending with '101' or '110'.

⇒ $\Sigma = \{0, 1\}$ $F = \{q_3, q_5\}$ $q_0 = q_0$

$Q = \{q_0, q_1, q_2, q_3, q_4, q_5\}$

S	0	1		
Σ q_0	q_0	q_1	$\Sigma 0$	$\Sigma 1$
1 q_1	q_2	q_4		
10 q_2	q_0	q_3	100	101
101 q_3	q_2	q_4	1010	1011
11 q_4	q_5	q_4	110	111
110 q_5	q_0	q_3	1100	1101



21/07 HW.

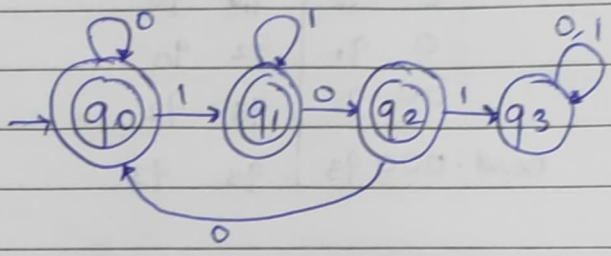
*) Design FSM for

a) not having '101' as substring.

$\Sigma = \{0, 1\}$ $F = \{q_0, q_1, q_2\}$ $q_0 = q_0$

$Q = \{q_0, q_1, q_2, q_3\}$

δ	0	1
q_0	q_0	q_1
1 q_1	q_2	q_1
10 q_2	q_0	q_3
101 q_3	q_3	q_3



b) having atleast two '0'.

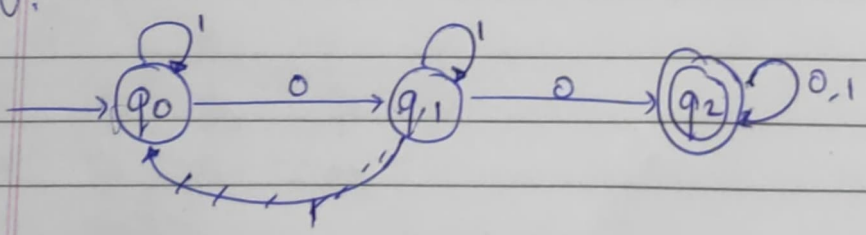
$\Sigma = \{0, 1\}$ $F = \{q_2\}$ $q_0 = q_0$

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

δ	0	1
ϵ q_0	q_1	q_0
0 q_1	q_2	q_0
00 q_2	q_2	q_2

$\Sigma 0$	$\Sigma 1$
00	01
000	001

10!0.

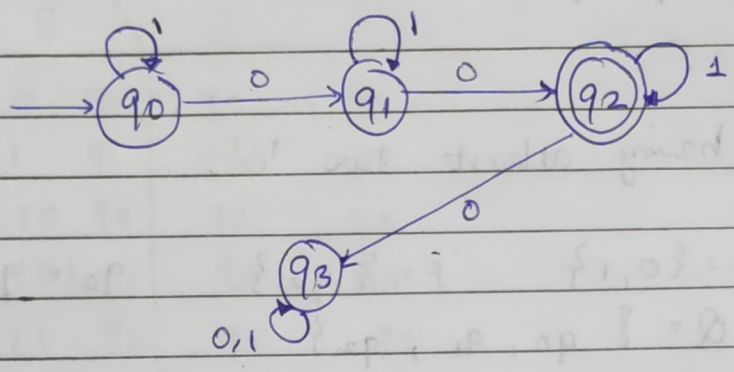


c) Exactly two '0'.

$\Sigma = \{0, 1\}$ $F = \{q_2\}$ $q_0 = q_0$

$Q = \{q_0, q_1, q_2, q_3\}$

	δ	0	1
ε	q_0	q_1	q_0
0	q_1	q_2	q_0
00	q_2	q_3	q_2
Dead. 000	q_3	q_3	q_3

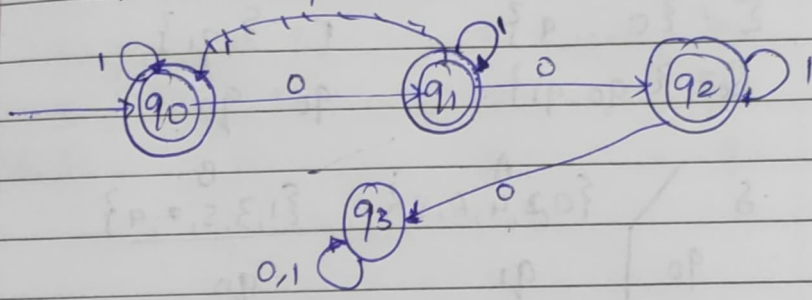


d) Atmost two '0'.

$\Sigma = \{0, 1\}$ $F = \{q_0, q_1, q_2\}$ $q_0 = q_0$

$Q = \{q_0, q_1, q_2, q_3\}$

δ		0	1
Σ	q_0	q_1	q_0
0	q_1	q_2	q_0
00	q_2	q_3	q_2
000	q_3	q_3	q_3



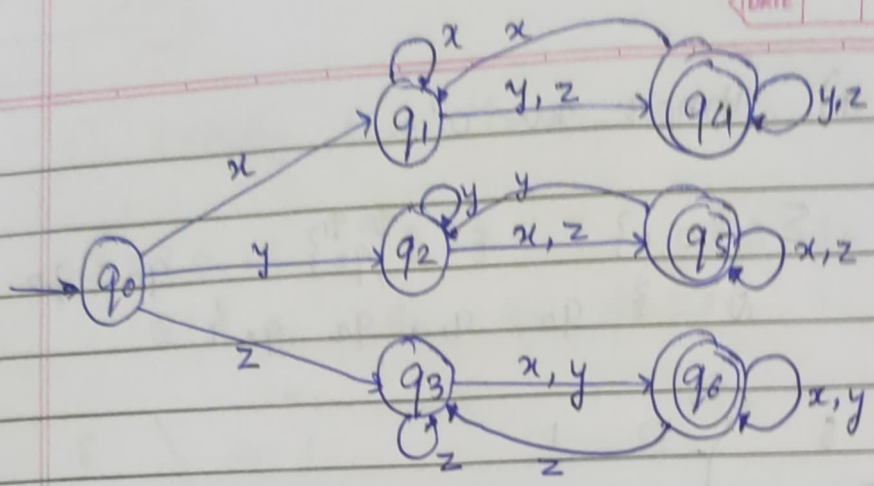
*) Design FSM to accept strings beginning & ending with diff symbols.

$\Sigma = \{x, y, z\}$ $F = \{q_4, q_5, q_6\}$

$q_0 = q_0$

$Q = \{q_0, q_1, q_2, q_3, q_4, q_5, q_6\}$

δ		x	y	z
Σ	q_0	q_1	q_2	q_3
x	q_1	q_1	q_4	
y	q_2			
z	q_3			
xyz	q_4			
yxz	q_5			
zxy	q_6			



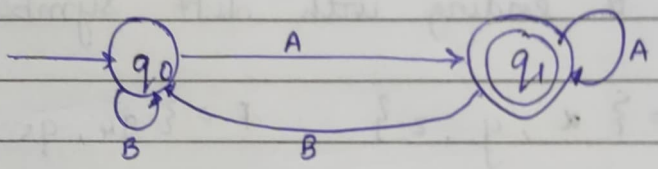
22/07

* Design FSM given decimal no is even.

$\Sigma = \{0 \ldots 9\}$ $F = \{q_1\}$

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

δ	A $\{0,2,4,6,8\}$	B $\{1,3,5,7,9\}$
q_0	q_1	q_0
q_1	q_1	q_0
~~q_2~~		



Eg:

13456

$\delta(q_0, 1) = q_0$

$\delta(q_0, 3) = q_0$

$\delta(q_0, 4) = q_1$

$\delta(q_1, 5) = q_0$

$\delta(q_0, 6) = q_1$

$\therefore q_1 \in F$

$\therefore$ Accept.

*) Design FSM to check whether decimal no. is divisible by 3.

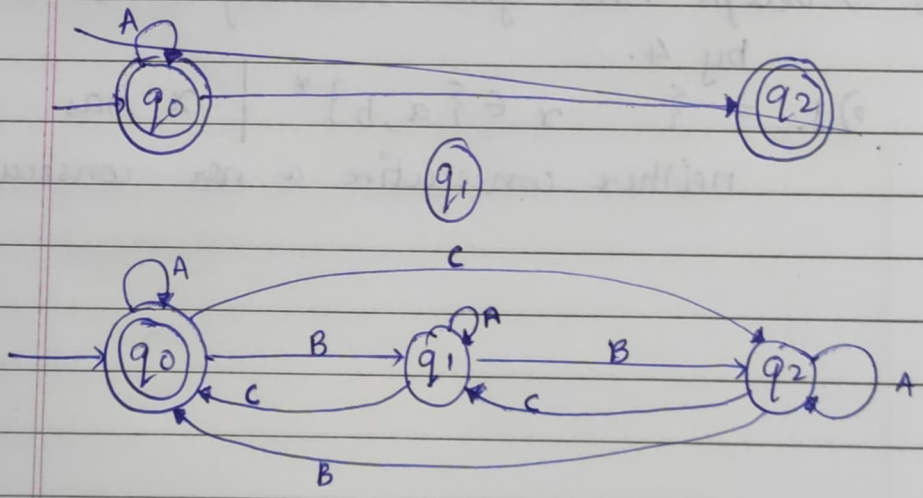
NOTE:

num %. n = Remainder { 0 .... n-1 }

$\Sigma = \{0 \ldots 9\}$ $F = \{q_0\}$

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

δ	A $\{0,3,6,9\}$	B $\{4,7,\}$	C $\{2,5,8\}$
Rem=0 q_0	q_0	q_1	q_2
Rem=1 q_1	q_1	q_2	q_0
Rem=2 q_2	q_2	q_0	q_1



Eg. 1002

$\delta(q_0, 1) = q_1$

$\delta(q_1, 0) = q_1$

$\delta(q_1, 0) = q_1$

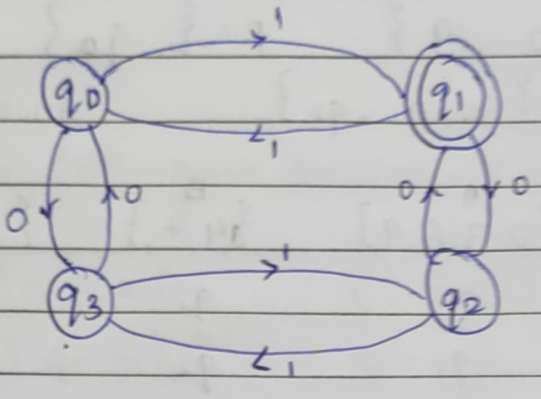
$\delta(q_1, 2) = q_0$

$\therefore$ Accept.

*) Design FSM for binary string for having even no. of '0' & odd no. of '1'.

$\Sigma = \{0, 1\}$ $Q = \{q_0, q_1, q_2, q_3\}$

$F = \{q_0, \}$



HW 1) Design FSM given ternary no. is divisible by 4.

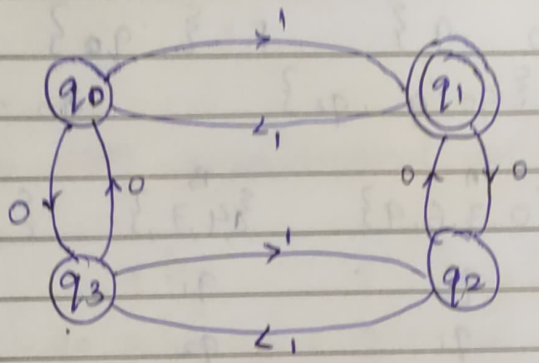
2) $L = \{ x \in \{a, b\}^* \mid x$ has neither consecutive a nor consecutive b.

{1} 11

*) Design FSM for binary string for having even no. of '0' & odd no. of '1'.

$\Sigma = \{0, 1\}$ $Q = \{q_0, q_1, q_2, q_3\}$

$F = \{q_0, \}$



HW 1) Design FSM given ternary no. is divisible by 4.

2) $L = \{ x \in \{a, b\}^* \mid x$ has neither consecutive a nor consecutive b.

*) Divisibility by 4 for decimal no.

$\Rightarrow$ $\Sigma = \{0, \ldots 9\}$ $q_0 = q_0$ $Q = \{q_0, q_1, q_2, q_3\}$

$F = \{q_0\}$

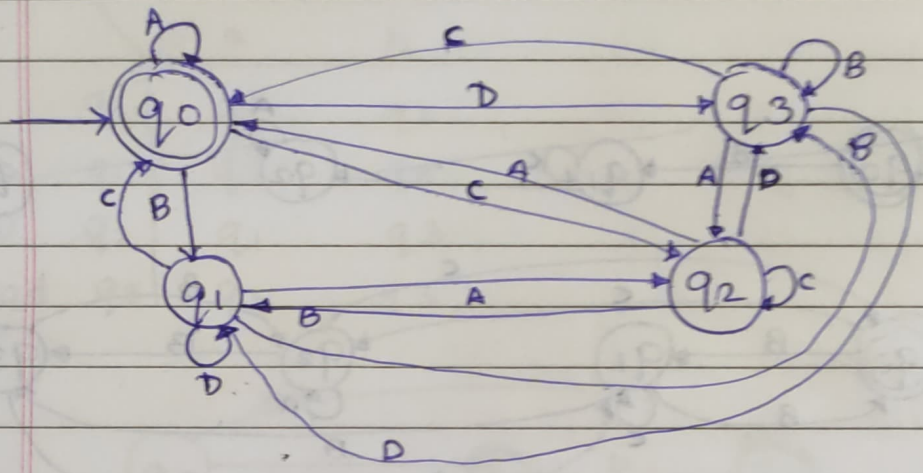
Logic : q_0 = Remainder 0

q_1 = Remainder 1

q_2 = Remainder 2

q_3 = Remainder 3.

		A {4,8,0}	B {1,5,9}	C {2,6}	D {3,7}
0	q_0	q_0	q_1	q_2	q_3
1	q_1	q_2	q_3	q_0	q_1
2	q_2	q_0	q_3	q_2	q_3
3	q_3	q_2	q_3	q_0	q_1



Eg: 1024

$\delta(q_0, 1) = q_1$

$\delta(q_1, 0) = q_2$

$\delta(q_2, 2) = q_2$

$\delta(q_2, 4) = q_0$

$\therefore q_0 \in$ Final state $\therefore$ Accept.

*) Ternary divisibility by 4.

$\Sigma = \{0, 1, 2\}$ $q_0 = q_0$ $Q = \{q_0, q_1, q_2, q_3\}$

$F = q_0$

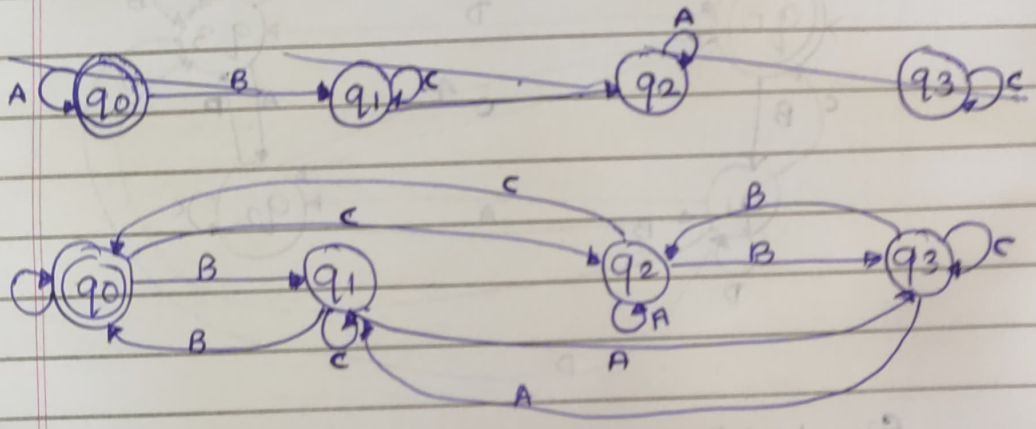
Logic: q_0 = Remainder 0

q_1 = Remainder 1

q_2 = " 2

q_3 = " 3

		A $\{(3R+0)\%4\}$	B $\{3R+1\%4\}$	C $\{(3R+2)\%4\}$
0	q_0	q_0	q_1	q_2
1	q_1	q_3	q_0	q_1
2	q_2	q_2	q_3	q_0
3	q_3	q_1	q_2	q_3



Eg: $(10202)_3 \Rightarrow (101)_{10}$

$\delta(q_0, 1) = q_1$

$\delta(q_1, 0) = q_3$

$\delta(q_3, 2) = q_3$

$\delta(q_3, 0) = q_1$

$\delta(q_1, 2) = q_1$

$\therefore$ Reject.

Eg: $(11121)_3 \Rightarrow (124)_{10}$

$\delta(q_0, 1) = q_1$

$\delta(q_1, 1) = q_0$

$\delta(q_0, 1) = q_1$

$\delta(q_1, 2) = q_1$

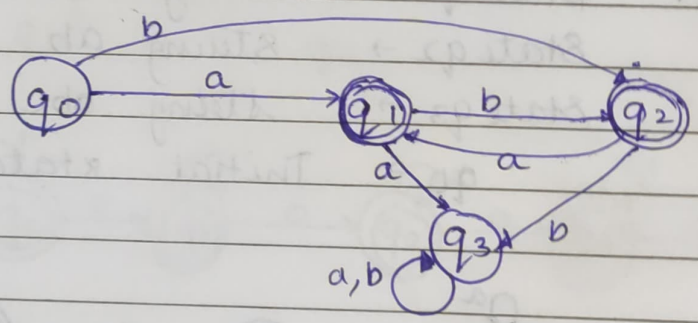
$\delta(q_1, 1) = q_0$ $\therefore$ Accept.

*) $L = \{ x \in \{a, b\} \mid x$ has neither consecutives a nor b . $\}$

$\Rightarrow \Sigma = \{a, b\}$ $\quad Q = \{q_0, q_1, q_2, q_3\}$

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

Σ	a	b
q_0	q_1	q_2
q_1	q_3	q_2
q_2	q_1	q_3
Dead q_3	q_3	q_3



Eg.

abab

$\delta(q_0, a) = q_1$

$\delta(q_1, b) = q_2$

$\delta(q_2, a) = q_1$

$\delta(q_1, b) = q_2$

$q_2 \in F$

$\therefore$ Accept.

abbaa

$\delta(q_0, a) = q_1$

$\delta(q_1, b) = q_2$

$\delta(q_2, b) = q_3$

$\delta(q_3, a) = q_3$

$\delta(q_3, a) = q_3$

$\therefore$ Reject

1*) $L = \{ x \in \{a,b,c\} \mid$ 'abc' is a substring $\}$

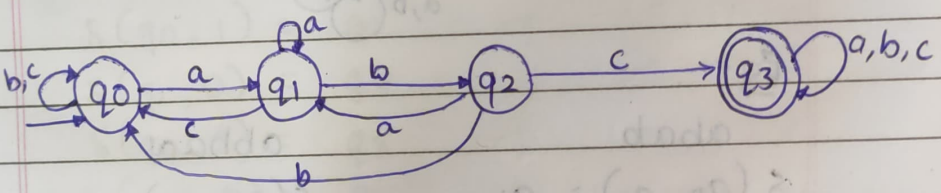
$\Rightarrow \Sigma = \{a,b,c\}$ $\quad q_0 = q_0$ $\quad F = \{q_3\}$

$Q = \{q_0, q_1, q_2, q_3\}$

Σ		a	b	c
ε	q_0	q_1	q_0	q_0
a	q_1	q_1	q_2	q_0
ab	q_2	q_1	q_0	q_3
abc	q_3	q_3	q_3	q_3

~~a~~ ab ~~ac~~
aba abb abc

Logic: State q_1 → String a
State q_2 → String ab
State q_3 → String abc (Final)
q_0 → Initial state.



Eg.
aabca
$\delta(q_0, a) = q_1$
$\delta(q_1, a) = q_1$
$\delta(q_1, b) = q_2$
$\delta(q_2, c) = q_3$
$\delta(q_3, a) = q_3$

$\because q_3 \in F$
$\therefore$ Accept.

Eg.
aaba
$\delta(q_0, a) = q_1$
$\delta(q_1, a) = q_1$
$\delta(q_1, b) = q_2$
$\delta(q_2, a) = q_1$

$\because q_1 \notin F$

$\therefore$ Reject.

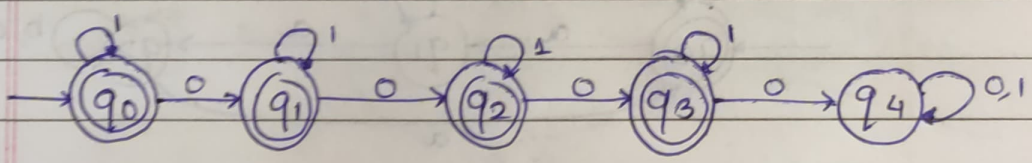
2*) $L = \{ x \in \{0,1\} \mid x$ does not contain more than 3 '0' $\}$

$\Rightarrow \Sigma = \{0, 1\}$ $\quad q_0 = q_0$

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

$F = \{q_0, q_1, q_2, q_3\}$

		0	1
ϵ	q_0	q_1	q_0
0	q_1	q_2	q_1
00	q_2	q_3	q_2
000	q_3	q_4	q_3
Dead	q_4	q_4	q_4



Eg.

100

$\delta(q_0, 1) = q_0$

$\delta(q_0, 0) = q_1$

$\delta(q_1, 0) = q_2$

$\because q_2 \in F$.

$\therefore$ Accept

Eg.

00001

$\delta(q_0, 0) = q_1$

$\delta(q_1, 0) = q_2$

$\delta(q_2, 0) = q_3$

$\delta(q_3, 0) = q_4$

$\delta(q_4, 1) = q_4$

$\because q_4 \notin F$

$\therefore$ Reject.

$|x|$

3x) $L = \{ x \in \{a, b, c\} \mid |x|_a \text{ is divisible by } 3 \}$

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

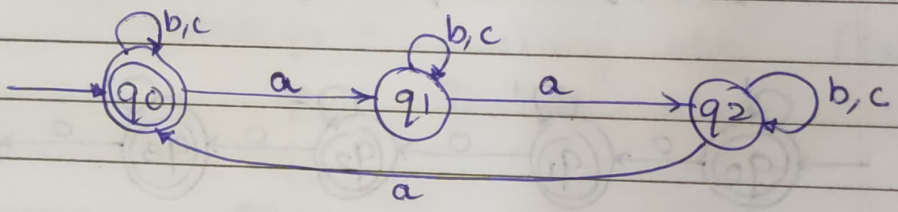
$Q = \{q_0, q_1, q_2\}$

$F = \{q_0\}$

$q_0 = q_0$

		a	b	c
	q_0	q_1	q_0	q_0
a	q_1	q_2	q_1	q_1
aa	q_2	q_0	q_2	q_2

Logic: No. of a's should be 3, 6, 9, etc.



Eg.

aaba

$\delta(q_0, a) = q_1$

$\delta(q_1, a) = q_2$

$\delta(q_2, b) = q_2$

$\delta(q_2, a) = q_0$

$\because q_0 \in F$

$\therefore$ Accept $\because$

aabbaa

$\delta(q_0, a) = q_1$

$\delta(q_1, a) = q_2$

$\delta(q_2, b) = q_2$

$\delta(q_2, a) = q_0$

$\delta(q_0, a) = q_1$

$\because q_1 \notin F$

$\therefore$ Reject.

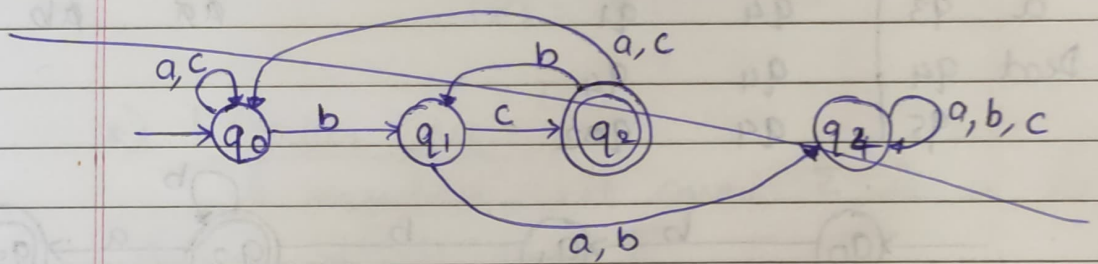
4*) $L = \{ x \in \{a,b,c\} \mid$ where every 'b' in x is immediately followed by 'c' $\}$.

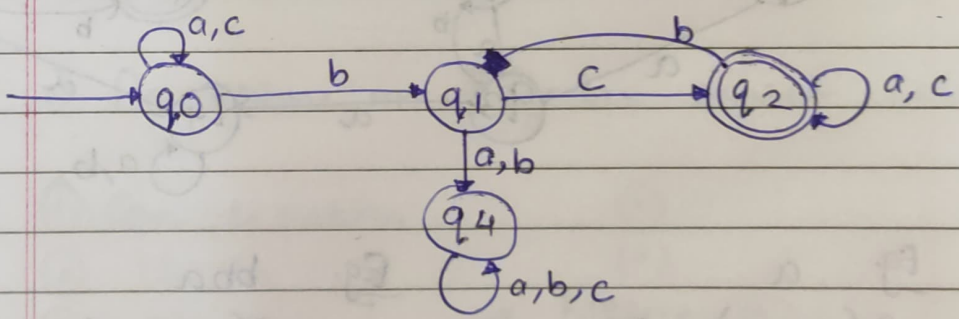
$\Rightarrow$ $\Sigma = \{a, b, c\}$ $\quad q_0 = q_0$

$Q = \{ q_0, q_1, q_2, q_4 \}$

$F = \{ q_2 \}$

		a	b	c
	q_0	q_0	q_1	q_0
b	q_1	q_4	q_4	q_2
bc	q_2	q_2	q_1	q_2
Dead	q_4	q_4	q_4	q_4





Eg: bcbca

$\delta(q_0, b) = q_1$

$\delta(q_1, c) = q_2$

$\delta(q_2, b) = q_1$

$\delta(q_1, c) = q_2$

$\delta(q_2, a) = q_2$

$\therefore$ Accept.

Eg: bcba

$\delta(q_0, b) = q_1$

$\delta(q_1, c) = q_2$

$\delta(q_2, b) = q_1$

$\delta(q_1, a) = q_4$

$q_4 \notin F$

$\therefore$ Reject.

5*) $L = \{ x \in \{a, b\} \mid x$ contains atleast two consecutive 'b' and NO two consecutive 'a'.$\}$

$\Rightarrow \Sigma = \{a, b\}$ $\quad q_0 = q_0$

$F = \{q_2, q_5\}$

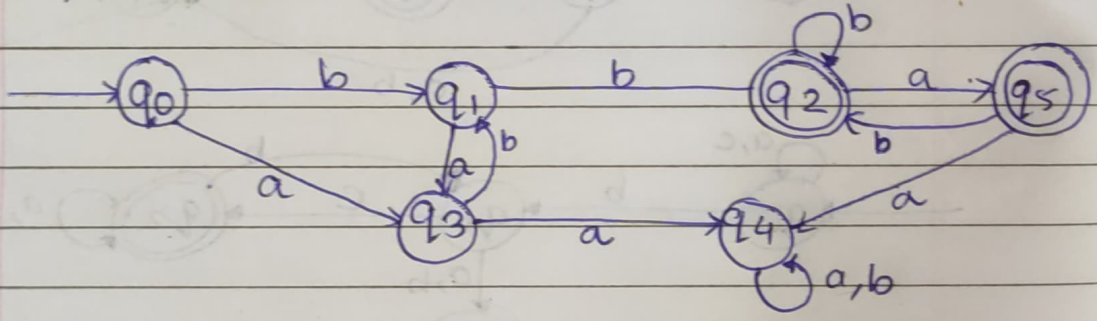
$Q = \{q_0, q_1, q_2, q_3, q_4, q_5\}$

		a	b
	q_0	q_3	q_1
b	q_1	q_3	q_2
bb	q_2	q_5	q_2
a	q_3	q_4	q_1
Dead	q_4	q_4	q_4
	q_5	q_4	q_2

ba bb

bba bbb

aa ab



Eg. a

$\delta(q_0, a) = q_3$

$\because q_3 \notin F$

$\therefore$ Reject.

Eg. bba

$\delta(q_0, b) = q_1$

$\delta(q_1, b) = q_2$

$\delta(q_2, a) = q_5$

$\because q_5 \in F$

$\therefore$ Accept.

28/07

MOD-2 :

Regular Expressions :-

* Regular Sets & Regular Expressions :

Let Σ be a finite set of symbols called as alphabets.

- A word x over Σ is any finite sequence of letters from Σ, length of the word is no. of alphabets within the word indicated as $|x|$

Eg For $\Sigma = \{0, 1\}$, 010 is a word
$|x| = 3$.

*) Regular set :
A regular set over Σ is a finite set of words over Σ.

*) Operations on Regular ~~expres~~ Sets.

1. Concatenation (.)
2. Union (+)
3. Closure → Kleene (*)
   → Positive (+)

1] Concatenation :

$L_1 = \{010, 000\}$

$L_2 = \{00, 10\}$

$L_1 \cdot L_2 = \{01000, 00000, 01010, 00010\}$

$L_2 \cdot L_1 = \{ 00010, 00000, 10010, 10000 \}$

$\therefore L_1 \cdot L_2 \neq L_2 \cdot L_1$ all the time.

2] Union:

$L_1 + L_2 = L_2 + L_1$ always.

3] Closure:

① Kleene closure

A^* = { set of all possible combinations including ϵ [null] }

② Positive closure:

A^+ = { excludes null ϵ }

*] REGULAR EXPRESSION:

① Design RE for all strings of binary symbols.

→ $(0+1)^*$

② Design RE for binary strings with atleast two consecutive '0'.

→ $(0+1)^* \; 00 \; (0+1)^*$

③ Design RE for binary strings with exactly two consecutive '0'.

$1^* 00\ 1^*$

④ End with '00'.

→ $(0+1)^* 00$

⑤ Strings ending with '011'

→ $(0+1)^* 011$

⑥ Not having 2 consecutive '0' and beginning with '1'.

→ ~~$1(11)^* 0 (1)^*$~~ $(10+1)^*$

⑦ $\Sigma = \{0, 1, 2\}$, Design RE for strings having any no. of '0' followed by any no. of '1', followed by any no. '2'.

→ $0^* \cdot 1^* \cdot 2^*$

⑧ same ↑, where atleast one occurance is imp.

$0^+ 1^+ 2^+$

⑨ Strings containing exactly two 0. $\Sigma = \{0, 1\}$

→ ~~$1^* (01)^* (01)^*$~~

$1^* 0 1^* 0 1^*$

(10) Strings containing atleast 2 '0'.

→ $(0+1)^* \cdot 0 \cdot (0+1)^* \cdot 0 \cdot (0+1)^*$

HW

1) String begin or end with '00' or '11'.

2) String begin and end with '00' or '11'.

3) String not ending with '01'.

4) Strings having every 0 immediately followed by '11'.

*) String beginning or end with '00' or '11'.

→ ~~$(00+11) \cdot (00+11)$~~

$(00+11)^+ (0+1)^* (00+11)^+$ ✗

$(00+11)(0+1)^* + (00+11)(0+1)^*$ ✓

*) String not ending with '01'.

→ $(0+1)^* \cdot (10+00+11)^{\bullet+}$

*) Having 0 immediately foll. by '11'.

~~$1^* \cdot (011) 1^*$~~

$(1+011)^*$

1) Start or end with '00' or '11'.

→ $(00+11)(0+1)^* + (0+1)^*(00+11)$

2) String not ending with '01'.

→ $(0+1)^* \cdot (10+00+11) + 0+1$

02/08

1) Strings of even length.

→ $(11+00+10+01)^*$

OR

$((0+1)^2)^*$

2) Strings of length multiple of 3.

→ $((0+1)^3)^*$

3) Strings beginning & ending with same symbol.

→ $0 \cdot (0+1)^* 0 + 1 \cdot (0+1)^* \cdot 1$

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

4) Strings having 0 placed at 3rd place from R.H.S.

→ $(0+1)^* \cdot 0 \cdot (0+1)^2$

5) Strings having atmost 2 '0'.

→ $((0)^2)^* + 1^*$ ✗

$(0 + 00) + (1)^*$ ✗

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

HW 1) Strings containing both '11' and '010' as substrings.

2) Even no. of 0s are foll by odd no. of 1s

3) Not having consecutive 0s.

(1) Strings having both '11' & '010' as substrings.

→

$(0+1)^* 11 (0+1)^* 010 (0+1)^* + ( )^* 010 ( )^* 11 ( )^*$

(3) $(1 + 01)^* (0 + \varepsilon)$

03/08

Mod: 3

FA with output:

1. Moore machine.
2. Mealy machine.

* Find 1's compliment of a Binary string with moore machine.

→ $\Sigma = \{0, 1\}$ $\quad Q = \{q_0, q_1\}$

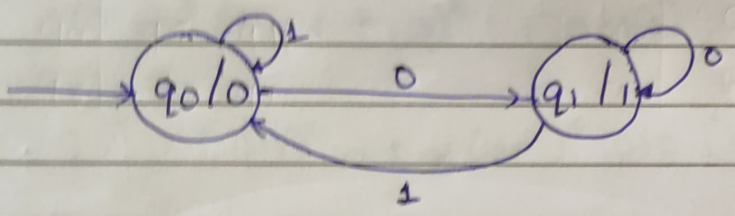
$\Delta = \{0, 1\}$ $\quad q_0 = q_0$

δ

	0	1
q_0	q_1	q_0
q_1	q_1	q_0

λ

	Δ
q_0	0
q_1	1



Eg: 1000

$\delta(q_0, 1) = q_0$

$\delta(q_0, 0) = q_1$

$\delta(q_0, 0) = q_1$

$\delta(q_1, 0) = q_1$

$\lambda(q_0) = 0$

$\lambda(q_1) = 1$

$\lambda(q_1) = 1$

$\lambda(q_1) = 1$

$\therefore$ Ans = 0111

Δ - dot delta
λ - lambda
δ - dell

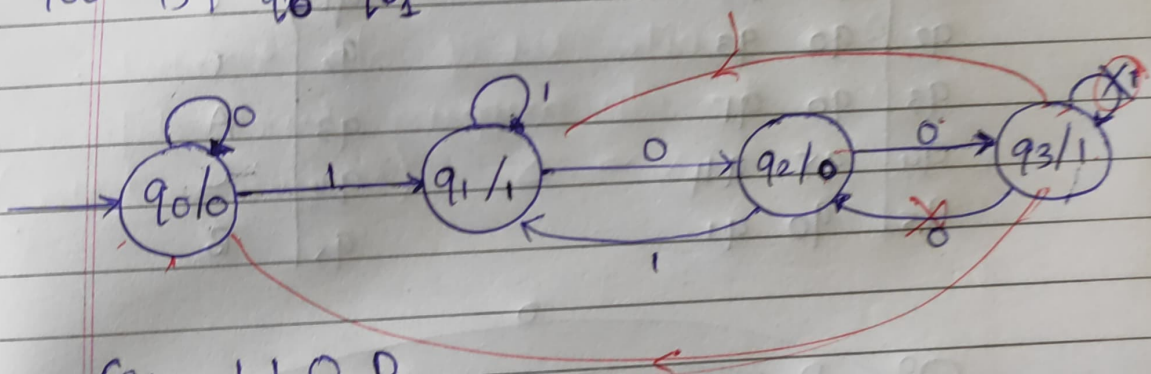
*) Design Moore machine to convert every occurance of '100' by '101'.

⇒ $\Sigma = \{0, 1\}$ $Q = \{q_0, q_1, q_2, q_3\}$
$\Delta = \{0, 1\}$ $q_0 = q_0$

δ	0	1
q_0	q_0	q_1
q_1	q_2	q_1
q_2	q_3	q_1
q_3	q_0	q_1

1
10
100

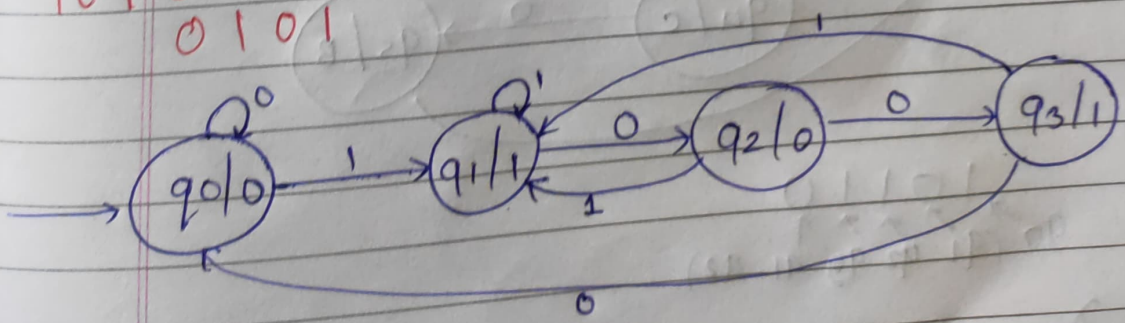
λ	Δ
q_0	0
q_1	1
q_2	0
q_3	1



Eg: 1 1 0 0
δ (q_1 q_1 q_2 q_3)
λ (1 1 0 1)

1000000
1010000
0101

100100
1



04/08/22

*) Design Moore machine such that if it ends with '101' o/p A, if it ends with '110' o/p B, otherwise o/p C.

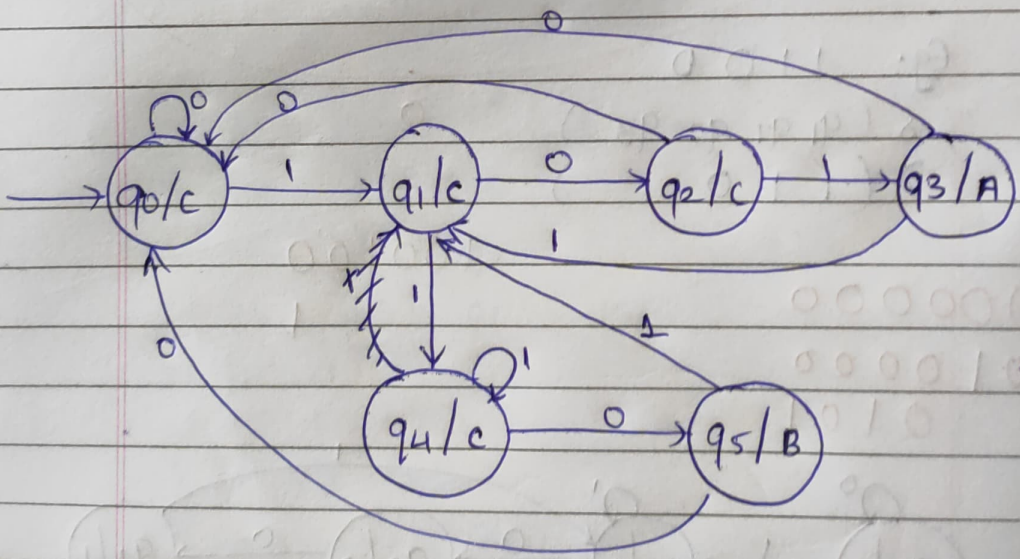
→ $\Sigma = \{0, 1\}$ $\Delta = \{A, B, C\}$

$Q = \{q_0, q_1, q_2, q_3, q_4, q_5\}$

$q_0 = q_0$

δ	0	1
q_0	q_0	q_1
q_1	q_2	q_4
q_2	q_0	q_3
q_3	q_0	q_1
q_4	q_5	q_4
q_5	q_0	q_1

λ	Δ
q_0	C
q_1	C
q_2	C
q_3	A
q_4	C
q_5	B



1 0 1 1 0

q_0 (q_1 q_2 q_3 q_1 q_2)

λ C C A C C

//

04/08/22

*) Design Moore machine such that if it ends with '101' o/p A, if it ends with '110' o/p B, otherwise o/p C.

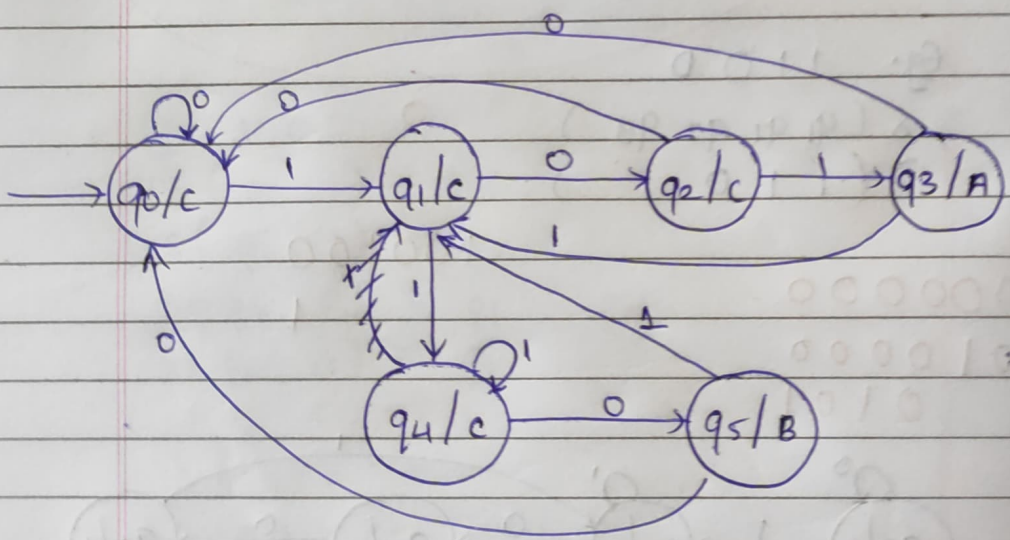
→ $\Sigma = \{0, 1\}$ $\Delta = \{A, B, C\}$

$Q = \{q_0, q_1, q_2, q_3, q_4, q_5\}$

$q_0 = q_0$

δ	0	1
q_0	q_0	q_1
q_1	q_2	q_4
q_2	q_0	q_3
q_3	q_0	q_1
q_4	q_5	q_4
q_5	q_0	q_1

λ	Δ
q_0	C
q_1	C
q_2	C
q_3	A
q_4	C
q_5	B



1 0 1 1 0

q_0 (q_1 q_2 q_3 q_1 q_2)

λ C C A C C

//.

*) Design MM that reads a seq. of vowels & give us an o/p same seq. except when 'i' is followed by 'e'; 'e' changes to 'u'.

→ $\Sigma = \{a, e, i, o, u\}$ $\Delta = \{a, e, i, o, u\}$

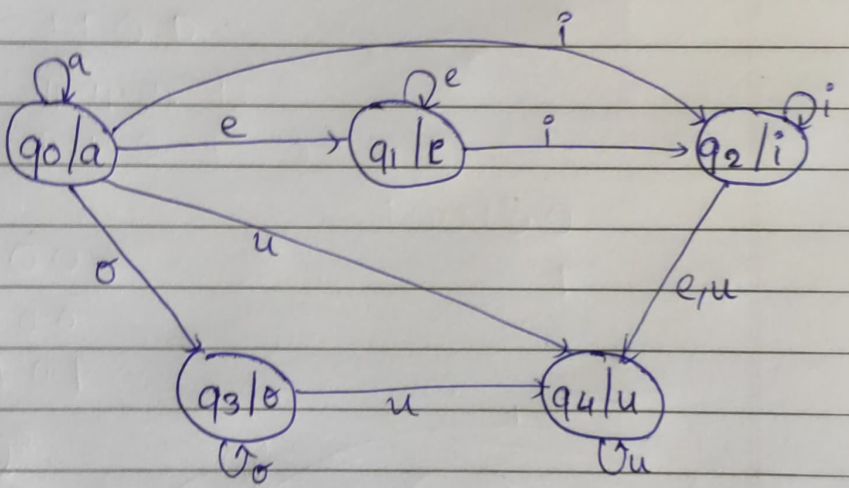
$Q = \{q_0, q_1, q_2, q_3, q_4\}$

$q_0 = q_0$

	a	e	i	o	u
q_0	q_0	q_1	q_2	q_3	q_4
q_1	q_0	q_1	q_2	q_3	q_4
q_2	q_0	q_4	q_2	q_3	q_4
q_3					
q_4					

λ

	Δ
q_0	a
q_1	e
q_2	i
q_3	o
q_4	u



v) Define Moore Machine :

Formally, a Moore machine is defined as a 6- tuples collection. Tuples are.

$(Q, \Sigma, q_0, \delta, \Delta, \lambda)$

where

Q = Set of

Σ =

q_0 = Initial state

δ =

Δ : Finite set of o/p symbols.

λ = o/p mapping func^n defined as $Q \rightarrow \Delta$ (Q giving delta)

*) Define Moore Machine :

Formally, a Moore machine is defined as a 6- ~~tuple~~ tuples collection. Tuples are.

$(Q, \Sigma, q_0, \delta, \Delta, \lambda)$

where

Q = Set of

Σ =

q_0 = Initial state

δ =

Δ : Finite set of o/p symbols.

λ = ~~Fo~~ O/p mapping func^n defined as $Q \rightarrow \Delta$ (Q giving delta.) *)

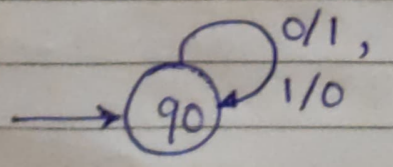
05/08

*) Design a Mealy Machine to find 1's complement of binary string.

(*) Mealy can be smaller/simpler than Moore machine.

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

$Q = \{q_0\}$ $q_0 = q_0$



Mealy → Output is associated with transition
→ λ is a 2D func^n.

δ	0	1
q_0	q_0	q_0

$Q \times \Sigma \to Q$

λ	0	1
q_0	1	0

$Q \times \Sigma \to \Delta$

Eg: 0001

$\delta(0\ 0\ 0\ 1)$

$(q_0\ q_0\ q_0\ q_0)\ q_0$

$\lambda(q_0\ q_0\ q_0\ q_0)$

0 0 0 1

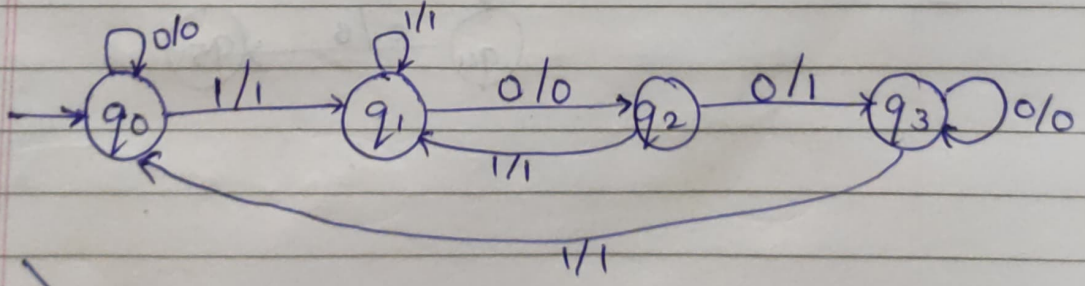
1 1 1 0

*) Design Mealy machine to change every '00' to '01'.

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

$Q = \{q_0, q_1, q_2, q_3\}$

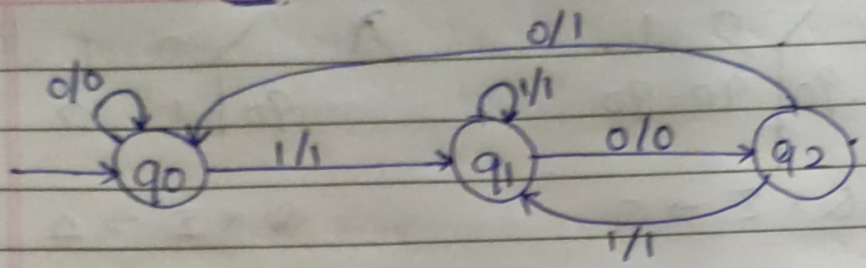
$q_0 = q_0$



δ	0	1
q_0	q_0	q_1
q_1	q_2	q_1
q_2	q_3	q_1
q_3	q_3	q_1

λ	0	1
q_0		
q_1		
q_2		
q_3		

OR.

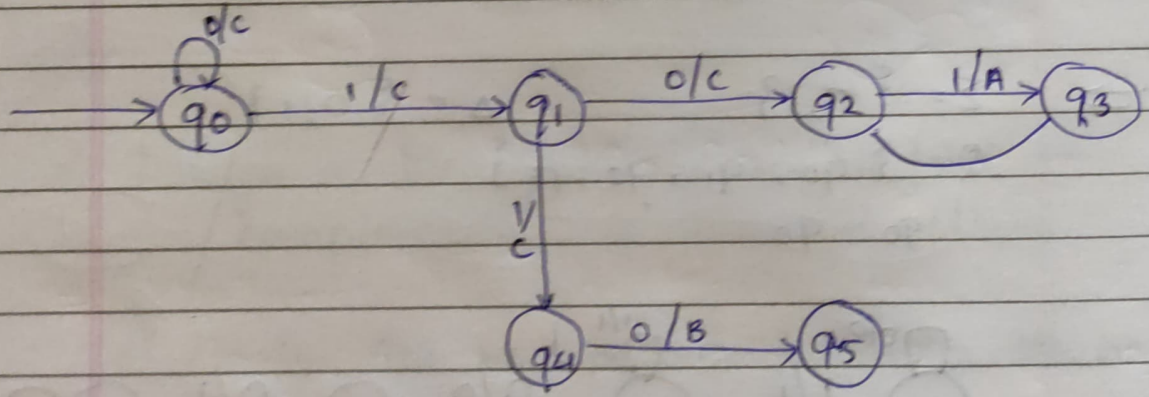


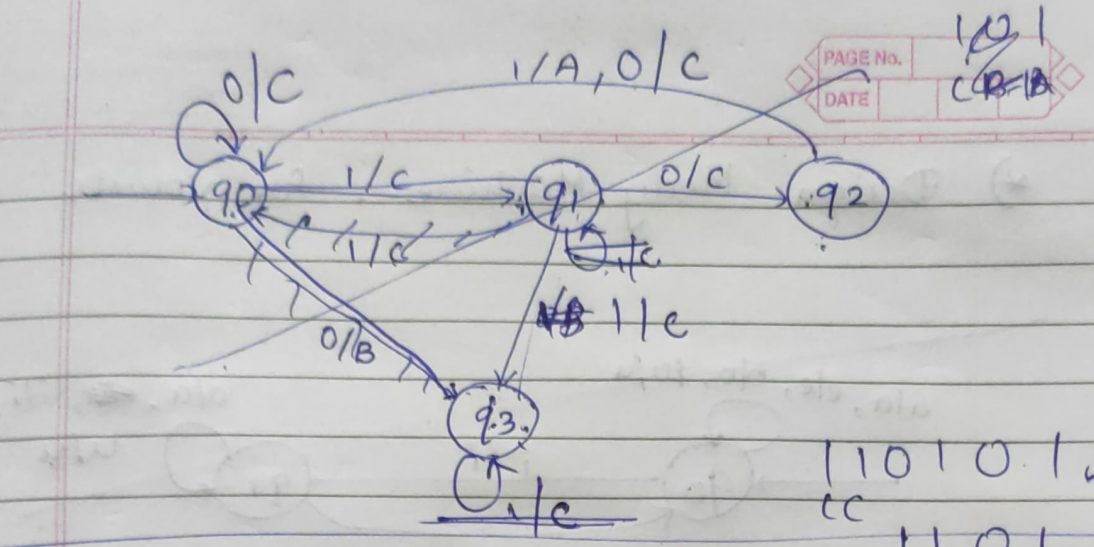
Reduced version.

*) Design Mealy machine, if ends with '101', o/p A, ends with '110', o/p B, else o/p C.

⇒ $\Sigma = \{0, 1\}$ $\Delta = \{A, B, C\}$

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

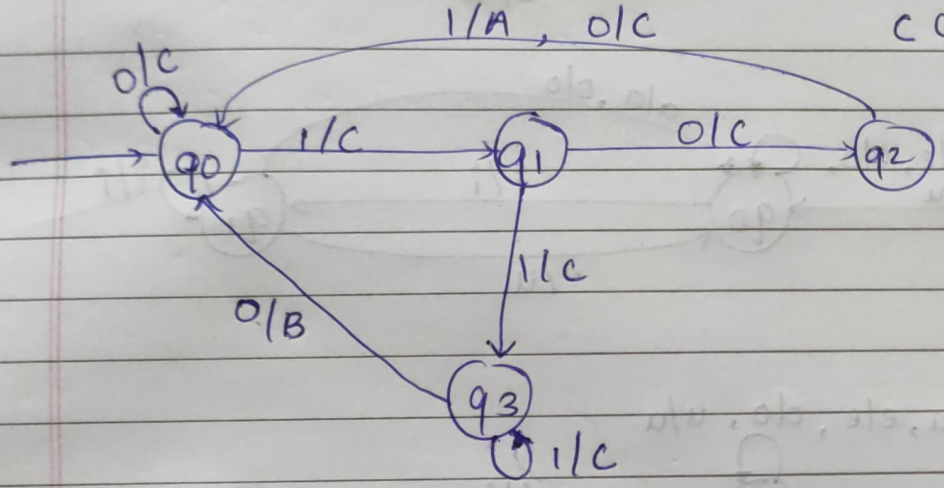




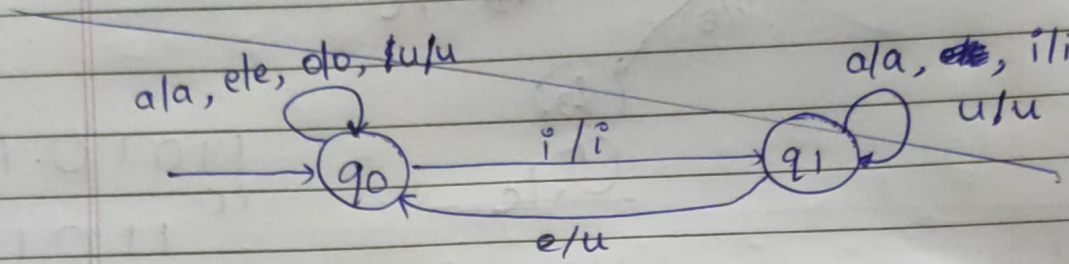
1101

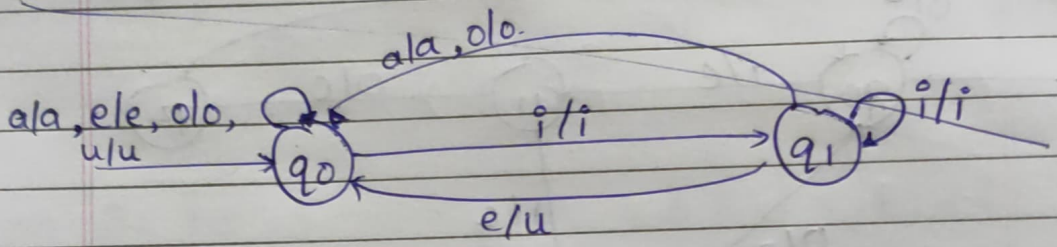
CC 1101

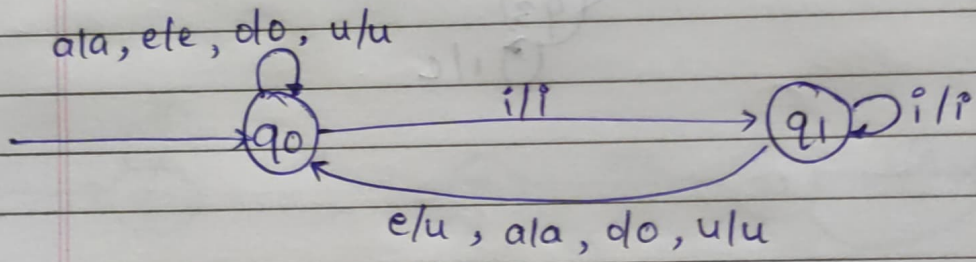
CCCAX



*) Design Mealy Machine, for vowels.







Mathematically, a Mealy machine is defined as a combination of 6 tuples,

$(Q, \Sigma, q_0, \delta, \Delta, \lambda)$

where,

Q =

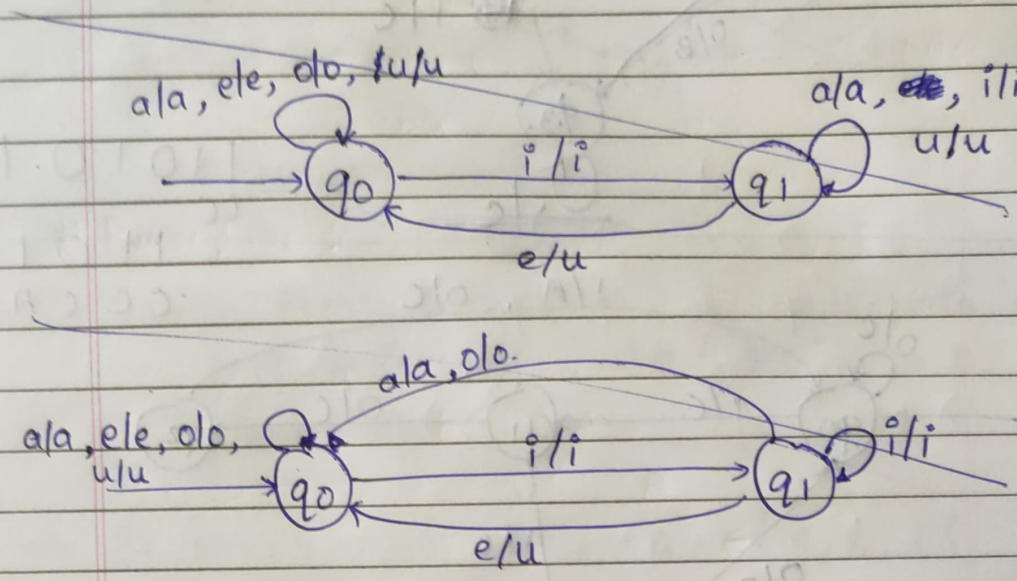
Σ =

δ =

Δ = Finite set of o/p signals.

λ = o/p mapping func defined as $(Q \times \Sigma \rightarrow \Delta)$.

*) Design Mealy Machine, for vowels.



Meathematically, a Mealy machine is defined as a combination of 6 tuples,

$(Q, \Sigma, q_0, \delta, \Delta, \lambda)$

where,

Q =

Σ =

δ =

Δ = Finite set of o/p signals.

λ = O/p mapping func defined as $(Q \times \Sigma \rightarrow \Delta)$.

10/08/22

PAGE No.
DATE

CONTEXT FREE GRAMMAR (CFG)

* Grammar (G) in AT is defined as a four-tuple collection

$G = \{ S, V, P, T \}$ where,

S = start symbol that belongs to V.
V = Finite set of non-terminals.
P = Set of rules called as production Rules.
T = Finite set of terminals.

*) Design CFG for $L = \{ w \in L \mid w$ begins with 'a' $\}$

$T = \{ a, b \}$

⇒ The set of Production rules for achieving the condition can be mentioned as:

P: $S \rightarrow aA$ — Rule 1

$A \rightarrow aA \mid bA \mid \epsilon$ — Rule 2, Rule 3, Rule 4

$V = \{ S, A \}$

$S = S$

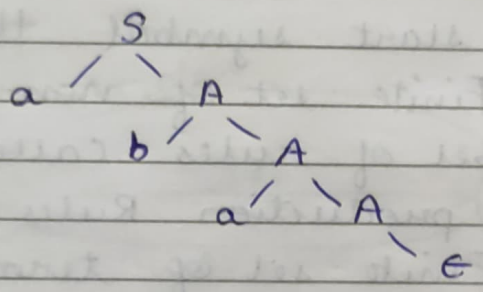
Eg: Taking into consideration an example 'aba',

ϵ - epsilon (null)

PAGE No.
DATE

S → aA - Rule 1
→ abA Rule 3
→ abaA Rule 2
→ abaϵ Rule 4
→ aba

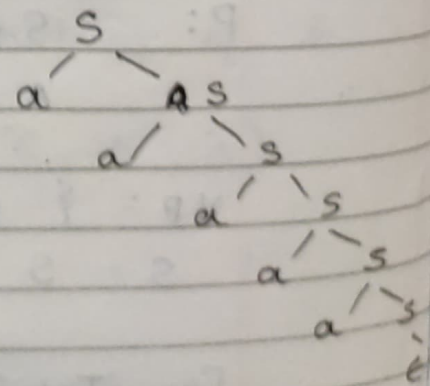
Parse Tree :



* Construct CFG for language having any no. of 'a'. T = { a }

⇒ Set of production rules :
P : S → aS | ϵ

S : S
V = { S }



Eg. aaaaa
S → aS
→ aaS
→ aaaS
→ aaaaS
→ aaaaaS
→ aaaaaϵ
→ aaaaa

*) Construct CFG for binary palindrome str.

⇒ Binary palindrome = 10101 / 0101010
(same from both sides).

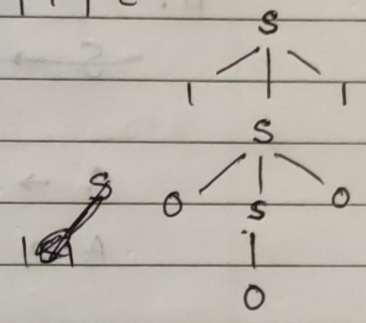
T = {0, 1}

P:

S → 0S0 | 1S1 | 0 | 1 | ϵ

V = {S}

S : S



Eg: 10001

S → 1S1

→ 10S01

→ 10001

) CFG for L = { w ∈ {a,b} | w begins & ends with same symbol }

⇒ T = {a, b}

S : S V = {S, A}

P:

~~S → aAa~~ | BAB | 0 | 1

~~A → 0~~

S → aAa | bAb | a | b | ϵ

A → aA | bA | ϵ

Eg: abbaa

S → aAa

→ abAa

→ abbAa

→ abbaAa

20-22-24

→ abbaϵa

→ abbaa

HW 1] CFG for set of odd length strings where 'a' is central symbol.

⇒ T = { a, b } V = { S }

P:

~~S → aAaAa | bAaAb~~

~~S → AaA | a | ϵ~~

A → aA | bA

S → aSa | aSb | bSa | bSb | a
(1) (2) (3) (4) (5)

Eg: aaaba

S → aSa ← Rule 1

→ aaSba ← Rule 2

→ aaaba ← Rule 5

2] Binary strings ending with 0.

⇒ T : { 0, 1 }

P:

S → 0S | 1S | 0
(1) (2) (3)

Eg: 110

S → 1S ← Rule 2

→ 11S ← Rule 2

→ 110 ← Rule 3

PAGE No.
DATE

*) $(0+1)^*$

$S \rightarrow 0S \mid 1S \mid 0 \mid 1 \mid \epsilon$

$S \rightarrow 0S \mid 1S \mid \epsilon$

*) $L = \{ w c w^R \mid w \in \{0,1\} \}$

*) $L = \{ a^n b^n \mid n \geq 1 \}$

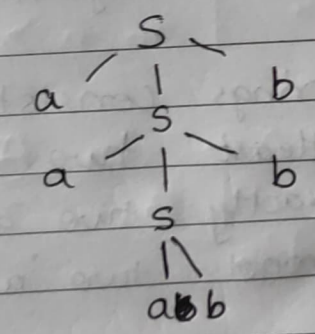
$\Rightarrow$ ~~$S \rightarrow aSb \mid \epsilon$~~

P:

$S \rightarrow aSb \mid ab$
(1) (2)

Eg. aaabbb

$S \rightarrow aSb$ - Rule 1
$\rightarrow aaSbb$ - Rule 1
$\rightarrow aaabbb$ - Rule 2



*) CFG for alternate seq. of a & b.

$\Rightarrow$ P: $S \rightarrow aB \mid bA$

$A \rightarrow aB \mid \epsilon$

$B \rightarrow bA \mid \epsilon$

silence

*) Palindrome : General

$S \rightarrow 0S0 \mid 1S1 \mid 0 \mid 1 \mid \epsilon$

Even :

$S \rightarrow 0S0 \mid 1S1 \mid \epsilon$

Odd :

$S \rightarrow 0S0 \mid 1S1 \mid 0 \mid 1$

*) Strings beginning with 'a'.

$\rightarrow$

$S \rightarrow aA$

$A \rightarrow aA \mid bA \mid \epsilon$

OR

$S \rightarrow Sb \mid Sa \mid a$

*) $\Sigma = \{0, 1, 2\}$ Strings with begin & end with same symbol.

*) $\Sigma = \{a, b\}$ Strings consisting

a) atleast two 'a'

b) exactly two 'a'

c) atmost two 'a'.

*) Ending with '111'. $\Sigma = \{0, 1\}$

$\rightarrow$

$S \rightarrow A111$

$A \rightarrow 0A \mid 1A \mid \epsilon$ ✓

OR

$S \rightarrow 0S \mid 1S \mid 111$

1) $S \rightarrow 0A0 \mid 1A1 \mid 2A2 \mid 0 \mid 1 \mid 2$
$A \rightarrow 0A \mid 1A \mid 2A \mid \epsilon$ ✓

2) a) Atleast two 'a'.

$S \rightarrow$
$S \rightarrow SaaS \mid .$ ✓
$S \rightarrow AaaA \quad AaAaA$
$A \rightarrow aA \mid bA \mid \epsilon$

b) Exactly two 'a'.

$S \rightarrow SaaS \mid$ ✓
$S \rightarrow AaaA \quad AaAaA$
$A \rightarrow bA \mid \epsilon$

c) Atmost two 'a'.

$S \rightarrow BABAB$
$A \rightarrow a \mid \epsilon$
$B \rightarrow bB \mid \epsilon$

17/08

* CFG for generation of odd length palindrom

$L = \{ wcw^R \mid w \in \{a,b\}\}$

aabaab

abaabb aaaabb

PAGE No.
DATE

1) CFG for strings containing no consecutive 'b' but 'a' can be consecutive.

→ S → baS | aS | ε | b

2) Strings with exactly twice as many a's as that of b'.

→ S → aabS | abaS | baaS | ∈

S → aSa | bS

S → aSaSbS | aSbSaS | bSaSaS | ∈

baa aab aba

3) CFG for language

$L = \{ a^n b^m c^n \mid m, n \geq 1 \}$

⇒ S → aABAc a A c Pabc
A → aBc | B
B → bB

aa bbb cc

S → aAc
A → aBc | B
B → bB | b

OR

S → aSc | aBc
B → bB | b

HW

*) RE $(00^* \cdot 11^*)$. (min (01).).

*) CFG with equal no. of 0's & 1's.

*) CFG for

$\{a^i b^j c^k \mid i, j, k \geq 1\}$

HW

*) CFG for RE $(00^* \cdot 11^*)$. (min (01).).

*) CFG with equal no. of 0's & 1's.

*) CFG for $\{a^i b^j c^k \mid i, j, k \geq 1\}$

$\Rightarrow$

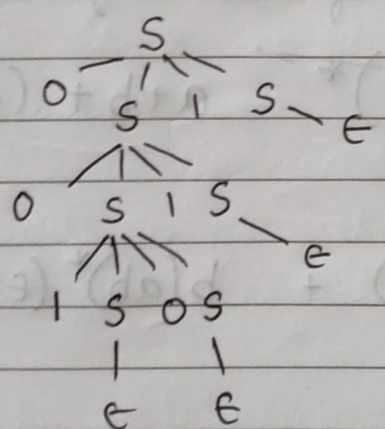
1) $S \rightarrow 0A1B$
   $A \rightarrow 0A \mid \epsilon$
   $B \rightarrow 1B \mid \epsilon$

2) $S \rightarrow 0S1S \mid 1S0S \mid \epsilon$

3) ~~$S \rightarrow$~~

Eg: 001011

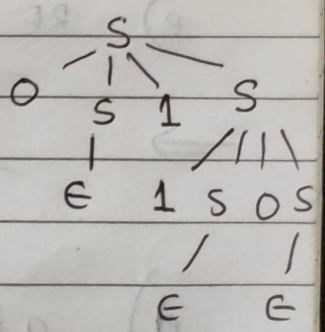
$S \rightarrow 0S1S$
$\rightarrow 00S11$
$\rightarrow 00\epsilon011$
$\rightarrow 001011$



$\Rightarrow 001011$

0110

$S \rightarrow 0S1S$
$\rightarrow 0\epsilon1 1S0S$
$\rightarrow 0110$



$\Rightarrow 0110$

Riya's Prac Sesh

PAGE No.
DATE 17 08 22

Q1 a) Strings ⇒ { ε, aa, b, aab, baa, aaaa, bb, bbb, bbbb, aabb, baab, bbaa }

b) No.
Since, every 'a' is followed by a 'b' & vice versa, and it is a union therefore, substrings aaa & bbb not possible.

c) RE for atleast one double letters aa & bb

⇒ $(a+b)^*\ (aa+bb)\ (a+b)^*$

d) RE for no occurance of double letters.

⇒ $(ab + ba)^* + a + b$ ✗

$a(ba)^* + b(ab)^* + \epsilon$

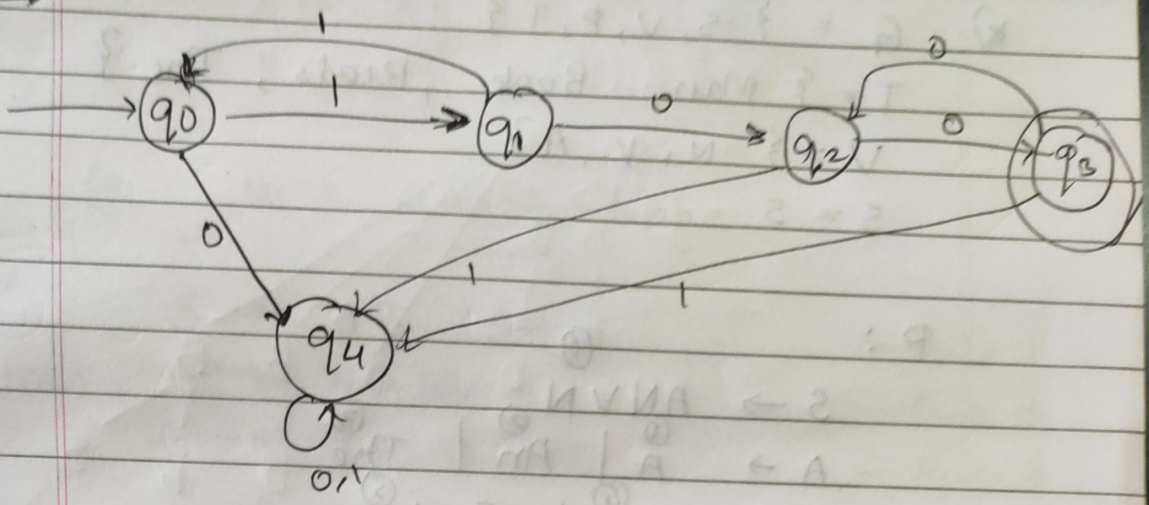
e) RE for strings of ~~exactly~~ length atmost two.

⇒ ~~$(a+b)^2)^*$~~ $a+b+(a+b)^2+\epsilon$

d) $a(ba)^*(\epsilon+b) + b(ab)^*(\epsilon+a) + \epsilon$

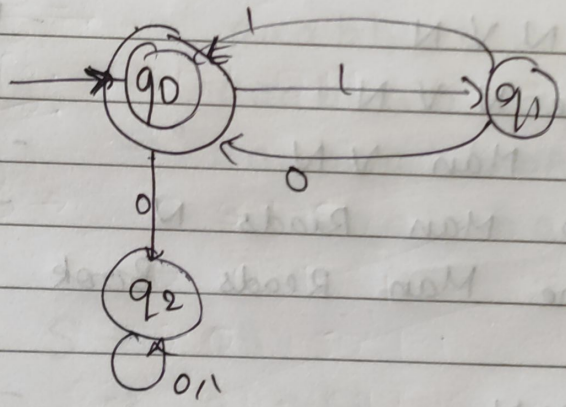
Q2 a) DFA for odd no. of 1 followed by even no. of 0

⇒



b) DFA for RE $(11+10)^*$.

⇒



5 mks / 10

18/08/22

Parse Tree:

→ Leftmost
→ Rightmost.

*) G = { S, V, P, T }
T = { Man, Book, Reads, The }
V = { N, V, A }
S = S

P:

S → ANVN ⓪
A → A ① | An ② | The ③
N → Man ④ | Book ⑤.
V → Reads ⑥

Eg Leftmost Derivation

S → ANVN - Rule 0
→ The NVN - Rule 3
→ The Man VN - Rule 4
→ The Man Reads N - Rule 6
→ The Man Reads Book - Rule 5

Rightmost Deriva"

S → ANVN - Rule 0
→ ANV Book - Rule 5
→ AN Reads Book - Rule 6
→ A Man Reads Book - Rule 4
→ The Man Reads Book. - Rule 3

*) The process of deriving a string using grammar rules is called derivation.

*) The diagramic representation of a derivation is called parse tree.

1 Eg: P:

$S \to aB \ (1) \mid bA \ (2)$

$A \to aS \ (3) \mid bAA \ (4) \mid a \ (5)$

$B \to bS \ (6) \mid aBB \ (7) \mid b \ (8)$

Eg: $w = aaabbabbba$

Leftmost

$S \to aB$ → Rule 1

$\to aaBB$ — Rule 7

$\to aaaBBB$ — Rule 7

$\to aaabBB$ → Rule 8

$\to aaabbB$ — Rule 8

$\to aaabbaBB$ → Rule 7

$\to aaabbabB$ → Rule 8

$\to aaabbabbS$ → Rule 6

$\to aaabbabbbA$ → Rule 2

$\to aaabbabbba$ → Rule 5

Right most

$S \to aB$ - Rule 1

$\to aaBB$ - Rule 7

$\to aaBB$ - Rule 7

$aaBaBB$ - Rule 7

$aaBaBbS$ - Rule 6

$aaBaBbbA$ - Rule 2

$aaBaBbba$ - Rule 5

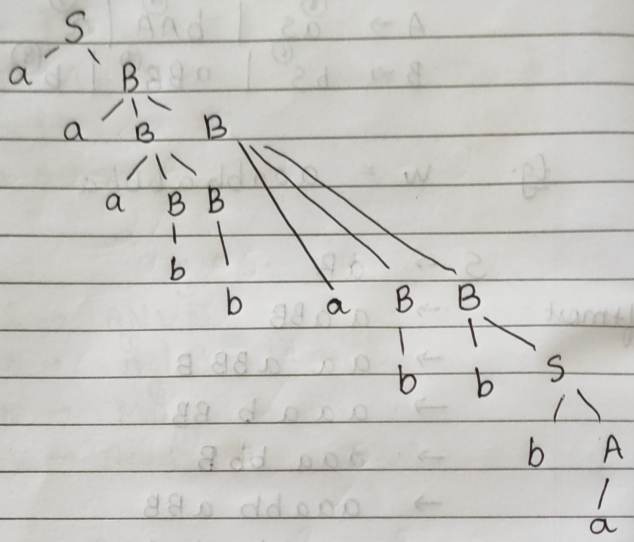
$aaBabbba$ - Rule 8

$aaaBBabbba$ - Rule 7

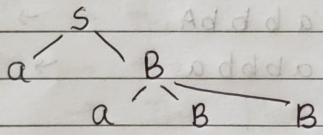
$aaaBbbabbba$ - Rule 8

$aaabbabbba$ - Rule 8.

★) leftmost Parse Tree:



★ Rightmost Parse Tree:



27) P: $S \to bB^{(1)} / aA^{(2)}$

$A \to b^{(3)} / bS^{(4)} / aAA^{(5)}$

$B \to a^{(6)} / aS^{(7)} / bBB^{(8)}$

- Leftmost Derivation:

$w = bbaababa$

$S \to bB$ - Rule 1

$\to bbBB$ - Rule 8

$\to bbaSB$ - Rule 7

$\to bbaaAB$ - Rule 2

$\to bbaabSB$ - Rule 4

$\to bbaabaAB$ - Rule 2

$\to bbaababB$ - Rule 3

$\to bbaababa$ - Rule 6

- Rightmost Derivation:

$S \to bB$ - Rule 1

$\to bbBB$ - Rule 8

$\to bbBaS$ - Rule 7

$\to bbBabB$ - Rule 1

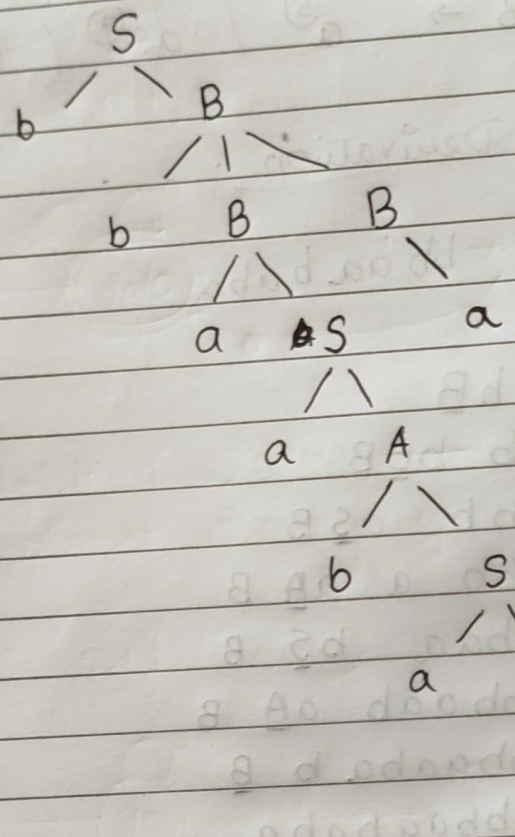
$\to bbBaba$ - Rule 6

$\to bbaSaba$ - Rule 7

$\to bbaAaba$ - Rule 2

$\to bbaababa$ - Rule 3.

*) Parse tree for leftmost derivation :



Riya's Prac Sesh 2

PAGE No.
DATE

Q1

a) (i) Define FA with o/p associated with Transition. (Mealy)

⇒ A mealy

b) Give 3 pts. of difference betⁿ Moore & Mealy machine.

⇒ ① Moore is FA with output associated with state.
Mealy is FA with output associated with transition.

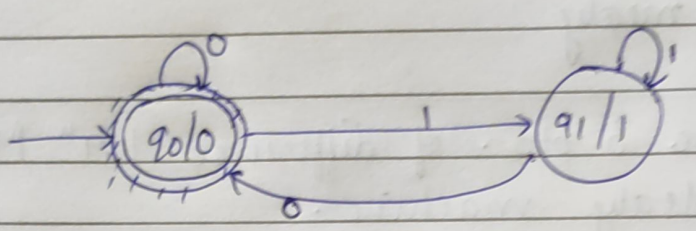
② The output f mapping funcⁿ of Moore machine is a 1D funcⁿ defined as.
$\lambda = Q \rightarrow \Delta$ i.e. Q giving Δ.
state giving output.

whereas,
The o/p mapping funcⁿ of Mealy is a 2D funcⁿ defined as
$\lambda = \Sigma \times Q \rightarrow \Delta$
input & state giving output.

③ The design of Mealy machine can be smaller / simpler (less no. of states) as compared to Moore.

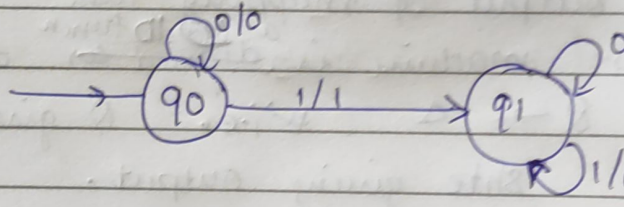
c) Moore Machine to check whether Binary no. is divisible by 2 or not.

⇒ $\Sigma = \{0, 1\}$



d) Mealy Machine to ~~check~~ find two's compliment of a binary no.

⇒ $\Sigma = \{0, 1\}$



Assuming string is read right to left.

Eg. 1101
1's = 0010
+ 1
0011.

Eg. 1101
1's 0010
+ 1
2's 0011

with machine:

$\lambda(q_0, 1) \rightarrow q_1/1$
$\lambda(q_1, 0) \rightarrow q_1/1$
$\lambda(q_1, 1) \rightarrow q_1/0$
$\lambda(q_1, \#) \rightarrow q_1/0$

0011

with machine:

$\lambda(q_0, 1) \rightarrow 1$
$(q_1, 1) \rightarrow 0$
$(q_1, 0) \rightarrow 1$
$(q_1, 1) \rightarrow 0$.

~~1 1 0 1~~

22/08/22

PT1

Ambiguous & Unambiguous Grammar

- A grammar is said to be ambiguous if for any string generated by it, it produces more than one -
  - * Parse Tree OR
  - Leftmost Derivation OR
  - Rightmost Derivation.

Types of Grammer
(on the basis of No. of Derivation trees)

Ambiguous — Unambiguous.

Eg 1 P:

$$E \rightarrow \overset{①}{E+E} \mid \overset{②}{E*E} \mid \overset{③}{id}$$

Eg: w = id + id * id

Leftmost Derivation:

$E \rightarrow$ ~~id + id~~ $E + E$ - Rule ①

$\rightarrow$ id + $E * E$ - Rule ②&③

$\rightarrow$ id + id * id - Rule ③

Right most :

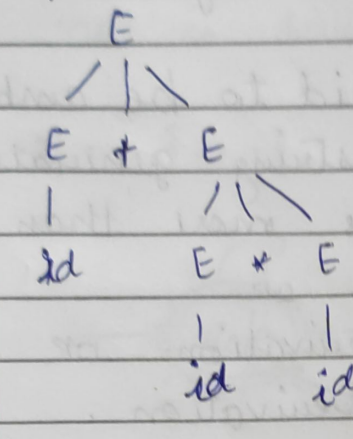
$E \rightarrow E + E$ - Rule ①

$\rightarrow E + E * E$ - Rule ②

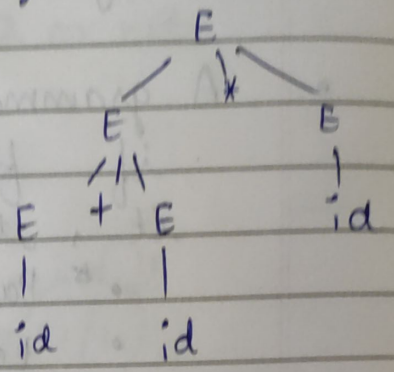
$\rightarrow E +$ id * id - Rule ③

$\rightarrow$ id + id * id - Rule ③

Leftmost :



Rightmost



2nd Leftmost Derivation:

E → E * E - Rule 2
→ E + E * E - Rule 1
~~→ id + id * id~~ - ~~Rule 3~~
→ id + E * E - Rule 3
→ id + id * E - Rule 3
→ id + id * id - Rule 3

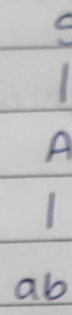
2nd Rightmost Derivation:

E → E * E - Rule 2
→ E * id - Rule 3
→ E + E * id - Rule 1
→ E + id * id - Rule 3
Parse tree → id + id * id - Rule 3.

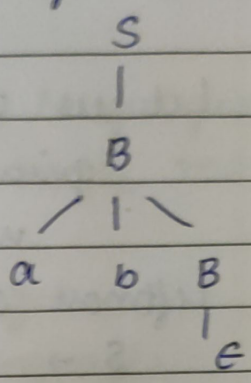
Eg. 2 P: S → A | B
A → aAb | ab
B → abB | ϵ

Consider ~~Example~~ string : ab w =

Parse ~~Leftmost~~ tree 1



Parse ~~Rightmost~~ Tree 2:



* Since two different parse trees exist for string w, the given grammar is ambiguous.

Eg.3.

⇒ P: $S \rightarrow AB \mid C$

$A \rightarrow aAb \mid ab$

$B \rightarrow acBd \mid cd$

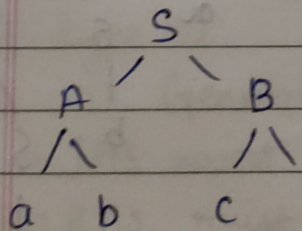
$C \rightarrow aCd \mid aDd$

$D \rightarrow bDc \mid bc$

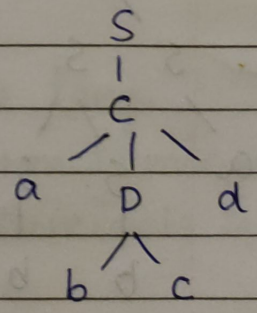
Taking into consideration string

$w = abcd$,

Parse tree 1,



Parse tree 2,



*). Since two .. ..

Eg.4 P: S → SS | a | b

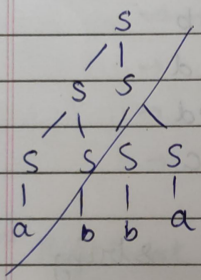
⇒ Let us consider string generated by given grammar:

w = abba

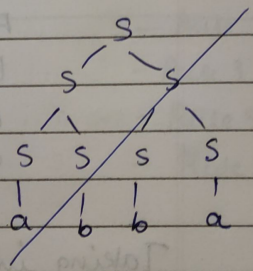
Leftmost derivation,

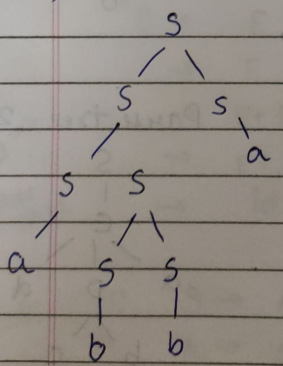
S → SS
→ SS S
→ SS S S
→ a b b a → 3 steps combined

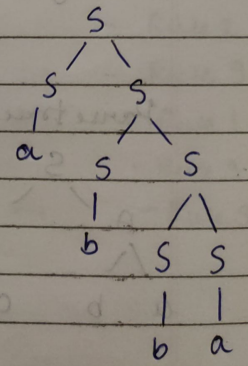
Parse tree 1:



Parse tree 2:







Since, two different PTs ∴ Ambiguous.

23/08/22

Chomsky Hierarchy:

* Comprises four types of languages & their associated grammars & machines.

Type 3:	Regular Languages.	FA
Type 2:	Content-Free Languages.	PDA
Type 1:	Content-sensitive	LBA
Type 0:	Recursively Enumerable	TM

① Type 3:

Generates Regular languages.

→ Must have a single non-terminal on LHS.

RHS must consist of single terminal or single terminal followed by single non-terminal.

Eg.

$$X \rightarrow aY$$

Machine used Finite Automata.

② Type 2:

Generates Content-Free lang.

→ LHS must have single non-terminal

→ RHS must be string of terminals & non-terminals.

Machine used to recognise is PushDown Automata.

③ Type 1:

Generates context sensitive lang.

→ Productions must be in form

$$\alpha A \beta \rightarrow \alpha \gamma \beta$$

$|LHS| <= |RHS|$

strings α & β may be empty but γ cannot be ~~non~~ empty.

Machine used - Linear Bounded Automation

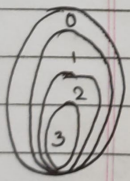
~~④ Type~~

④ Type-0: Unrestricted Grammar.

Generates Recursively enumerable lang.

* Production have no restrictions.

Machine used - Turing Machine



*) Steps for reduction of a given Grammar G.

① Identify non-generating symbols .... symbols not deriving any string.

② Identify non-reachable & eliminate those productions.

Eg.1) Remove useless symbol from given CFG

P: S → aB | bX
A → Bad | bSX | a
B → aSB | bBX
X → SBD | aBx | ad

⇒ • A & X directly derive string a & ad, hence Useful.
• But B doesn't derive any string, so B is a non-generating symbol.

∴ P: S → bX
A → bSX | a
X → ad

∵ Also A is a non-reachable symbol so eliminate A.

∴ Final Grammar is:

P: S → bX
X → ad

∴ the new equivalent grammar G' can be represented as:

Eg 2: $G' = \{ S, V', P', T \}$

$S = S$

$V' = \{ S, X \}$

$T = \{ a, b \}$

$P' = \{$

$S \rightarrow bX$

$X \Rightarrow ad \quad \}$

Eg.2] Find equivalent useful Grammar G'.

$A \rightarrow xyz \mid Xyzz$

$X \rightarrow Xz \mid xYz$

$Y \rightarrow yYy \mid Xz$

$Z \rightarrow Zy \mid z$

⇒ ~~A & X or~~

- A & Z are useful cos they directly derive to a string of terminals xyz & z
- X & Y are not useful.

∴ Eliminating.

[Eg. 3] $S \to AB \mid a$
$A \to BC \mid b$
$B \to aB \mid C$
$C \to aC \mid B$

$\Rightarrow$ Final : $S \to a$

[Eg. 4] $S \to AB \mid AC$
$A \to aAb \mid bAa \mid a$
$B \to bbA \mid aaB \mid AB$
$C \to abCA \mid aDb$
$D \to bD \mid aC$

$\Rightarrow$ After removing non-generating symbols,
$S \to AB$
$A \to aAb \mid bAa \mid a$
$B \to bbA \mid aaB \mid AB$

[Eg. 5] $S \to AB \mid B \mid a$
$A \to aA$
$B \to b$

$\Rightarrow$ $S \to B \mid a$
$B \to b$

[Eg. 6] $S \to abS \mid abA \mid abB$
$A \to cd$
$B \to aB$
$C \to dc$

⇒ S → abS | abA
A → cd
~~C → dc~~ Non-reachable.

Eg. 7] S → aAa | aBC
A → aS | bD
B → aBa | b
C → abb | DD
D → aDa ⇒ Non-generating.

⇒ S → aAa | aBC
A → aS
B → aBa | b
C → abb

Eg. 8] S → aaB | abA | aaS
A → aA
B → ab | b
C → ad // Non-reachable.

→ S → aaB | aaS
B → ab | b

Eg. 9] S → AC | BS | B
A → aA | aF
B → CF | b
C → cC | D
D → aD | BD | C
E → aA | BSA
F → bB | b

Ans:
S → aBS | B
A → aF
B → b
F → bB | b

Final:
S → BS | B
B → b

Eg.10] $S \rightarrow EA$
$A \rightarrow abA \mid ab$
$C \rightarrow EC \mid Ab$
$E \rightarrow bC$
$G \rightarrow EbE \mid CE \mid ba$

$\Rightarrow$ $S \rightarrow EA$
$A \rightarrow abA \mid ab$
$C \rightarrow EC \mid Ab$
$E \rightarrow bC$

29/08 # Removing null productions.

$\Rightarrow$ Simplification of Grammar.

(2) Removal of null productions:

Assuming A gives null production,
$A \rightarrow \epsilon$,

Step 1: Look for the null productions whose right side contains A.

Step 2: Replace each ocurrance of A in each of these productions with ϵ

Step 3: Now, combine the result of step 2 with original production & remove ϵ production.

Eg. 1) $S \rightarrow aA$
$A \rightarrow b \mid \epsilon$

$\Rightarrow$ Here, A is null production.
$\therefore$ Replacing ϵ at the place of A, we get
$S \rightarrow a$
$\therefore$ Combining the new production to original grammar, & removing ϵ.

$S \rightarrow aA \mid a$
$A \rightarrow b$

2) $S \rightarrow aSa$
$S \rightarrow bSb \mid \epsilon$

$\Rightarrow$ Here, S is a null production.

Ans $\rightarrow$ $S \rightarrow aSa \mid bSb \mid aa \mid bb$

3) $S \rightarrow aSb \mid aAb \mid ab \mid a$
$A \rightarrow \epsilon$

$\Rightarrow$ Here, A is the null production.

Ans $\rightarrow$ $S \rightarrow aSb \mid ab \mid a \mid aAb$

4) $S \rightarrow AB$
$A \rightarrow aAA \mid \epsilon$
$B \rightarrow bBB \mid \epsilon$

⇒ S → AB | A | B
A → aAA | aA | a
X A → ∈
B → bBB | bB | b
X B → ∈

5) S → ABAC | BAC | ABC | AC | C | BC | AAC
A → aA ~~| ∈~~ | a
B → bB ~~| ∈~~ | b
C → c
A → ∈
B → ∈

⇒ S → ABAC | ABC | BAC | BC | AAC | AC | C
A → aA | a
B → bB | b
C → c

6) S → XYX | XY | YX | X | Y | XX
X → 0X | 0
Y → 1Y | 1
X → ∈
Y → ∈

⇒ S → XYX | XY | YX | XX | X | Y
X → 0X | 0
Y → 1Y | 1