$\langle + | + \rangle : \; (7\langle 0| - 3i\langle 1|)\,(|0\rangle + (2+i)|1\rangle)$

$= 7\langle 0|0\rangle + (14+7i)\langle 0|1\rangle - 3i\langle 1|0\rangle - 6i - 3i^2\langle 1|1\rangle \qquad i^2 = -1$

$= 7 - 6i + 3$

$= 10 - 6i.$

$$\begin{bmatrix} 7 & -3i \end{bmatrix} \begin{bmatrix} 1 \\ 2+i \end{bmatrix} = 7 + (-3i)(2+i) = 7 + (-6i + 3).$$

$\boxed{\text{TOA}}$

25-01-24

Language:- Set of strings involving symbols from an alphabet.

Alphabet: → An alphabet is a finite non empty set of symbols, which is used to ~~replace~~) represent the input of the machine

Denoted by Σ

Common Alphabets:

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

$\Sigma = \{a, b, c, \ldots z\}$

$\Sigma = \{\text{ASCII char}\}$

Scanned with CamScanner

STRING: string ~~ons~~ are the sequence of symbol of the alphabet.

"aaba" is an valid string from an alphabet $\Sigma = \{a, b\}$

"0100" is an valid string from an alphabet $\Sigma = \{0, 1\}$

Empty String:-

→ Every Alphabet has a special char called empty string.

→ which means zero occurence of an element.

Denoted by Λ, ε (Epsilon, Null)

Power of an Alphabet:

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

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

0 → length of the string.

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

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

Kleen Star : Σ^*

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

$\Sigma^* = \Sigma^0 \cup \Sigma^1 \cup \Sigma^2 \ldots\ldots \Sigma^\infty$

$= \{\varepsilon, 0, 1, 00, 10, 11, 01, 000, 111, \ldots\}$

$\bigcup_{i=0}^{i=\infty} \{\underset{\text{string}}{W} \mid |W| = i\}$

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

$= \{\varepsilon, a, b, aa, ab, ba, bb, aaa, \ldots\}$

$\Sigma = \{(ab)^2, c\}$

$\{\varepsilon, ab, c, abab, cc, abc, cab, \ldots\}$

Kleen Plus: Σ^+

$\Sigma^+ = \Sigma^1 \cup \Sigma^2 \cup \Sigma^3 \cdot \text{—} \; \Sigma^\infty$

$\bigcup_{i=1}^{i=\infty} \{w \mid |w| = i\}$

$\{a, b, ab, ba, \text{—}\}$

$L^+ = L^* . L \qquad L = \{a, b\}$

$\{\varepsilon, a, b, aa, bb, \ldots\} . \{a, b\}$

$= \{a, b, aa, ab, ba, bb, \text{—}\}$

~~concatetat~~ concatenation is dot prod

10011 r 111

A = 1100

B = 1011

A.B = 1100 1011

(i): $\{x \in \{a,b\}^* \mid |x| \leq 8\}$ $2^8 =$

$= \{\varepsilon, a, b, aa, ab, ba, bb, aaa, aab, bbb, \ldots\}$

aaaaaaaa

(ii): $\{x \in \{a,b\}^* \mid |x| \text{ is odd}\}$

$= \{\varepsilon, a, b, aaa, bbb, aab, aba, bab, \ldots\}$

baa

(iii): $\{x \in \{a,b\}^* \mid n_a(x) \geq n_b(x)\}$

$= \{\varepsilon, a, aa, aaa, aab, aba, baa, \ldots\}$

ab, ba

abc

num of substring. $= \frac{n(n+1)}{2}$ =6 n=10

$\frac{10(10+1)}{2} = \frac{\cancel{10}^{5}(11)}{\cancel{2}}$

$= 55$

ε, a, b, c, ab, bc, ac, abc,

$$\left|\frac{2}{\sqrt{7}}\right|^2$$

$1 \rightarrow \frac{2}{7}$

$0 \rightarrow \frac{5}{7}$

30-01-24

$\frac{10(10+1)}{2}$, $\frac{110}{2}$, 55

{a, b, c, d, e,}

$\frac{n(n+1)}{2}$

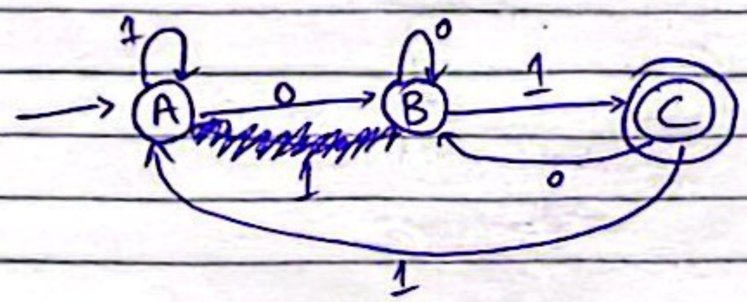
TOA

DFA (Deterministic Finite Automata)

- → It is a study of abstract computational module.
- → Machines are not equally powerful.
- → Finite in term of memory.

  Finite memory → Finite states.

  - → State Diagram.
  - → State Transition Diagram.
  - → Finite State Machine.
  - → Transition Graph.



State :- A

Transition :- A → B

Final State :- (A)

Initial state :- $\longrightarrow$ (A)

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

$\cancel{0}\cancel{1}\cancel{1}\cancel{0}\cancel{1}$
$BCAB\boxed{C}$

DFA (5 Tuples)

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

Q = Set of Finite states $Q = \{A, B, C\}$

Σ = Set of non-empty Finite set of input Alphabets. $\{0, 1\}$

q_0 = Initial state. $q_0 = A$

F = Set of Final state. $F = \{C\}$

δ = Transition Function.

Transition Func
$\delta(Q \times \Sigma) \rightarrow$ (

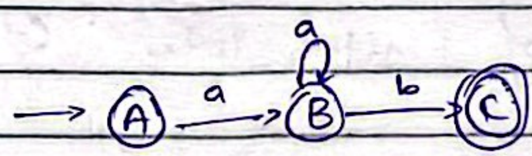
Transition Table :

δ	0	1
→A	B	A
B	B	C
*C	B	A

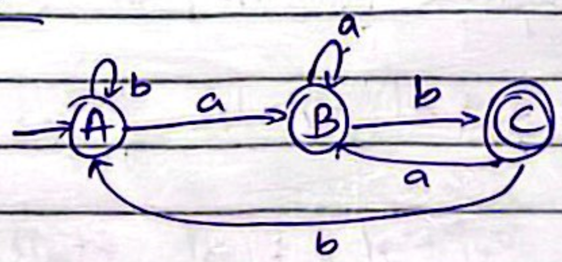
ε ko accept karana hai to initial state final state hogi.

Ends with ab

b aab

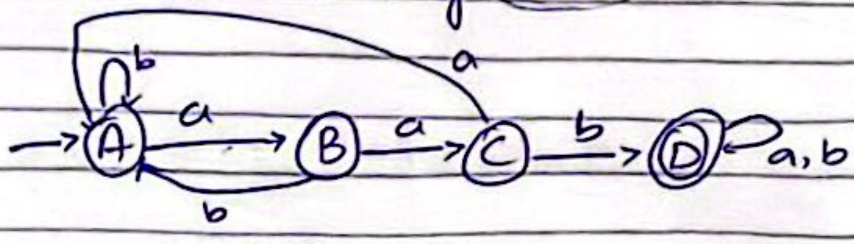


Ends with ab



ab b ab

Substring (aab)



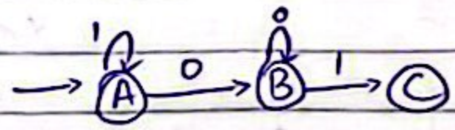
ab a a

aa a a ab

Q7: even num of zero or ends with 1

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

001   001   1 01

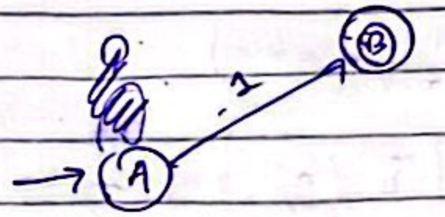


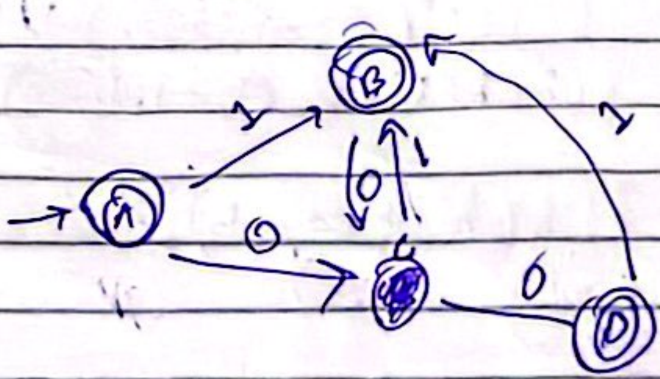
00 1

11 00

0
2
4

101





TOA 01-02-24

Transition Function

string = abc

$\delta(Q, \Sigma) \rightarrow Q$

$\delta(q, a) \rightarrow q_1$

$\delta(q_1, b) \rightarrow q_2$

$\delta(q_2, c) \rightarrow q_3$

Extended Transition Function.

$$\rightarrow q \xrightarrow{a} q_1 \xrightarrow{b} q_2 \xrightarrow{c} q_3$$

$\hat{\delta}(Q, \Sigma^*) \rightarrow Q$ — all possible strings $\varepsilon, a, aba, \ldots$

$\hat{\delta}(q, abc) \rightarrow q_3$

$\hat{\delta}(A, \varepsilon) \rightarrow A$ — A: state, ε: Epsilon

$\hat{\delta}(q, ya)$ $\quad y \in \Sigma^*$, $a \in \Sigma$

$\downarrow$

$\delta(\hat{\delta}(q, y), a)$

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

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

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

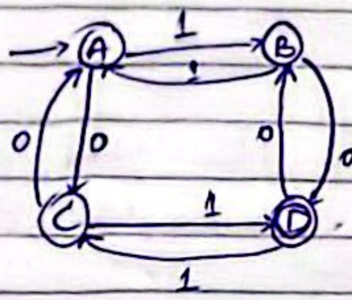
$\delta(\delta(\delta(\hat{\delta}(q, \varepsilon), a), b), c)$

$\delta(\delta(\delta(q, a), b), c)$

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

$\delta(q_2, c) \rightarrow q_3.$

Extended Transition Function :



0100

$\delta^\wedge(A, 0100)$

$\delta(\delta^\wedge(A, 010), 0)$

$\delta(\delta(\delta^\wedge(A, 01), 0), 0)$

$\delta(\delta(\delta(\delta^\wedge(A, 0), 1), 0), 0)$

$\delta(\delta(\delta(\delta(\delta^\wedge(A, \varepsilon), 0), 1), 0), 0)$

$\delta(\delta(\delta(\delta(A, 0), 1), 0), 0)$

$\delta(\delta(\delta(C, 1), 0), 0)$

$\delta(\delta(D, 0), 0)$

$\delta(\delta(B, 0) \rightarrow D.$

1.0000

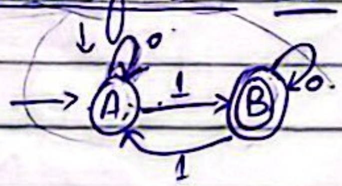
111

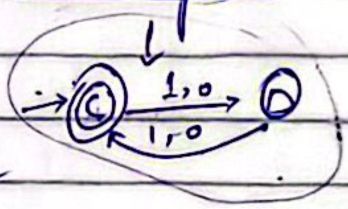
10000

Closure Property of DFA

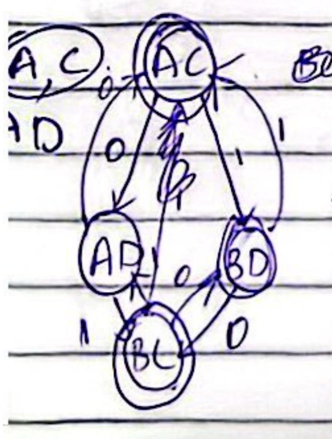
→ an odd num of 1's and even length.

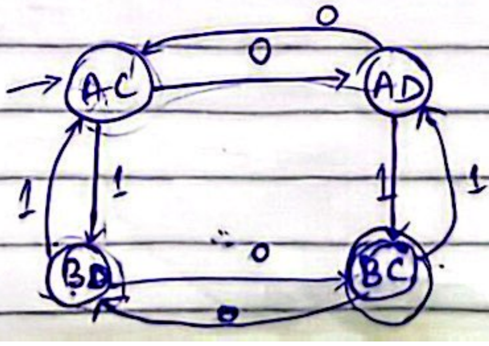
$\boxed{01}$





BC





A → starting

C

AND → Intersection

Final state : {B} {C}

{BC}

OR → Union

Final state : B . C   jahan jahan dono hon ge wo final state

Complement :-

Final states → Non Final

Non Final state → Final state

No Transition change.

Difference

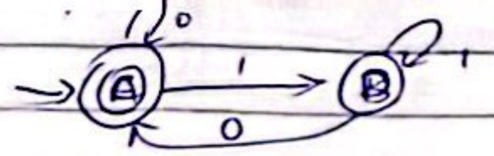
$$\boxed{L_1 - L_2} = \boxed{M_1 \cap \overline{M_2}}$$

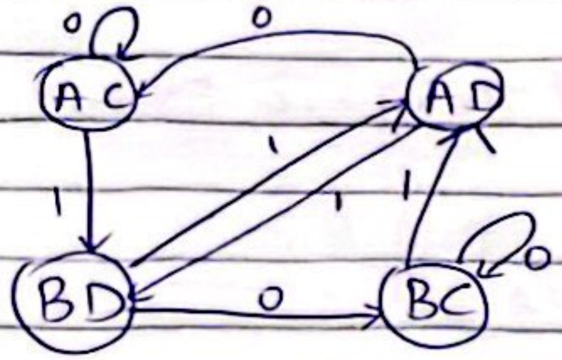
phele complement phir intersection with M_1.

$$L_1 - L_2 = M_1 \cap \overline{M_2}$$

1001 ✓

(a) :





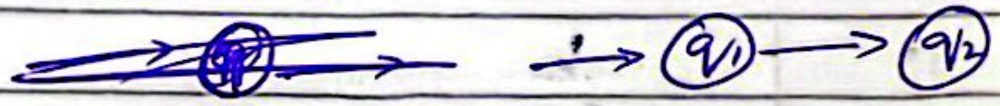
✓ Intersection AC

TOA 13-02-24

NFA (Non-Deterministic Finite Automata

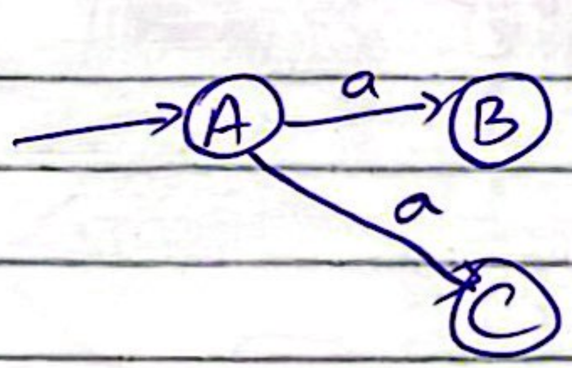
$\delta \; Q \times \Sigma \rightarrow 2^{Q}$

$\{q_1, q_2\}$

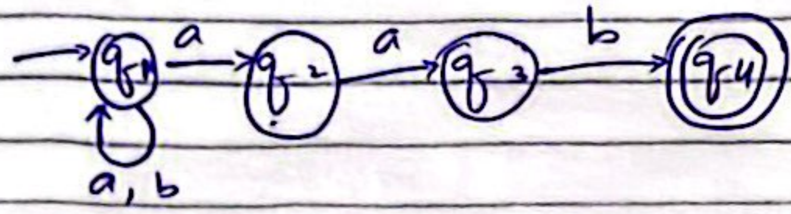


$\{\varepsilon\}$ $\{q_1\}$ $\{q_2\}$ $\{q_1, q_2\}$

$\{a, b\}$



Start and ends with B



Conversion of NFA TO DFA

Transition Table for NFA

δ	a	b
$\rightarrow q_1$	q_1, q_2	q_1
q_2	q_3	—
q_3	—	q_4
$*q_4$	—	—

Transition Table For DFA

δ	a	b
$\rightarrow \{q_1\}$	$\{q_1, q_2\}$	$\{q_1\}$
$\{q_1, q_2\}$	$\{q_1, q_2, q_3\}$	$\{q_1\}$
$\{q_1, q_2, q_3\}$	$\{q_1, q_2, q_3\}$	$\{q_1, q_4\}$
$*\{q_1, q_4\}$	$\{q_1, q_2\}$	$\{q_1\}$

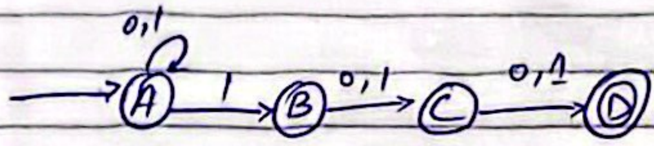
$\{q_3 q_4\} = \phi.$

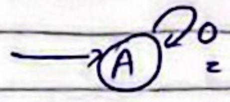
$\boxed{\{0,1\}^* \{0\}}$

$\{0,1\}^* \{1\} \{0,1\}^2$

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

$\{100, 101, 110, 111, \ldots\}$





$= \{0\}^*$

TOA

15-02-24

Regular Language
- Regular Grammar (RG)
- Finite Machine (FM)
  - NFA-E
  - NFA
  - DFA
- Regular Expression (

R.E → NFA-E → NFA → DFA (MDFA

NFA-E: Epsilon

MDFA: Minimal

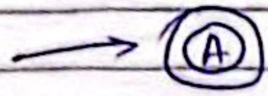
DFA: $\delta\ Q \times \Sigma \to Q$

NFA: $\delta: Q \times \Sigma \to 2^Q$

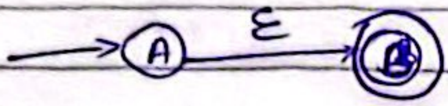
NFA-E: $\delta: Q \times \{\Sigma \cup \{\varepsilon\}\} \to 2^Q$

ε: Epsilon (Λ)

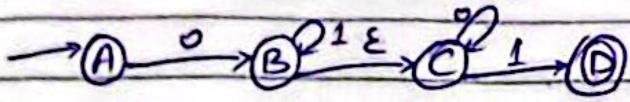
Scanned with CamScanner

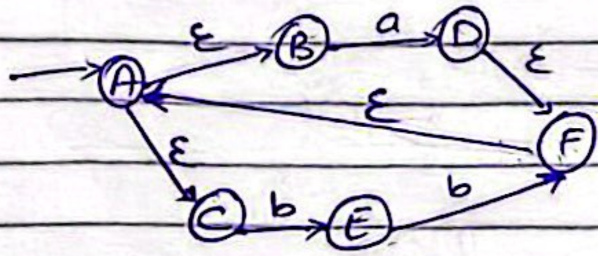


Epsilon اپنے پر اور آگے کر جاتا ہے.



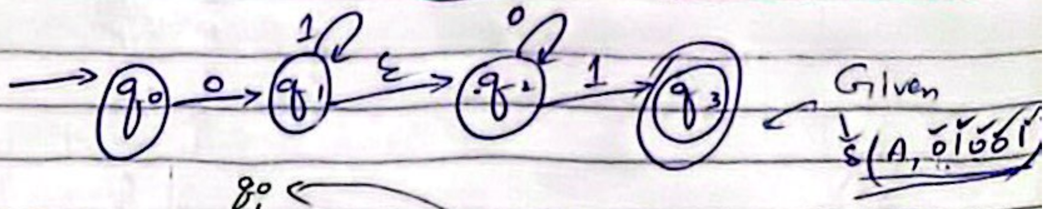
$\{0\}\{1\}^*\{0\}^*\{1\}$

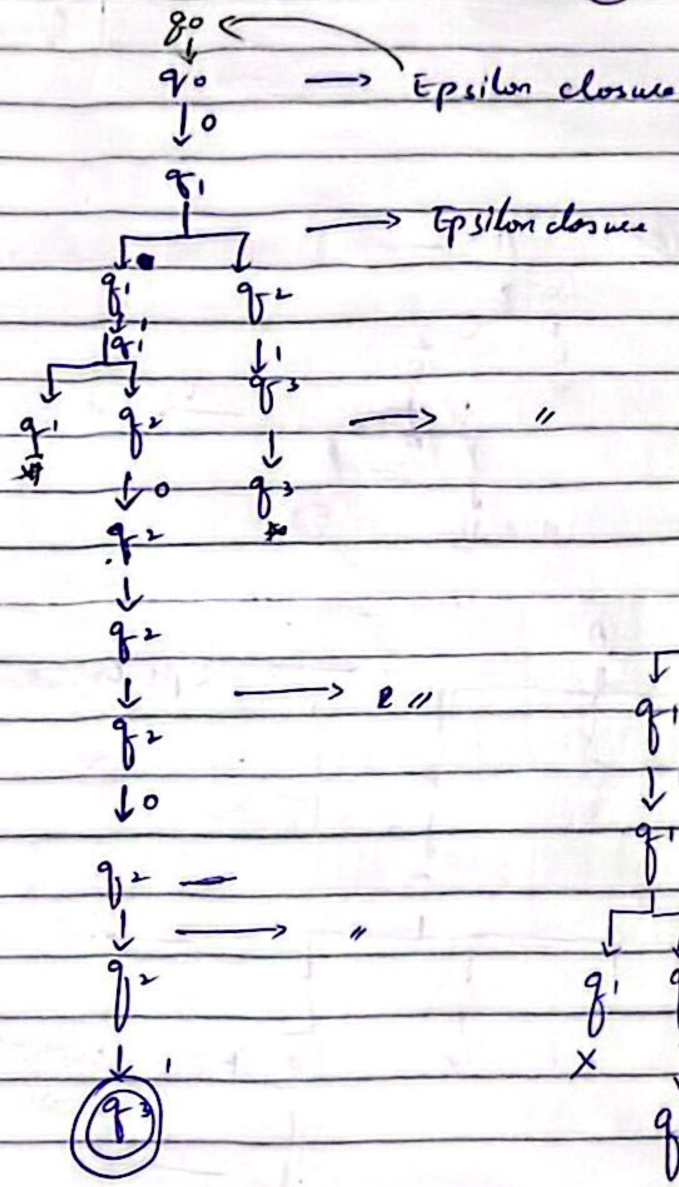


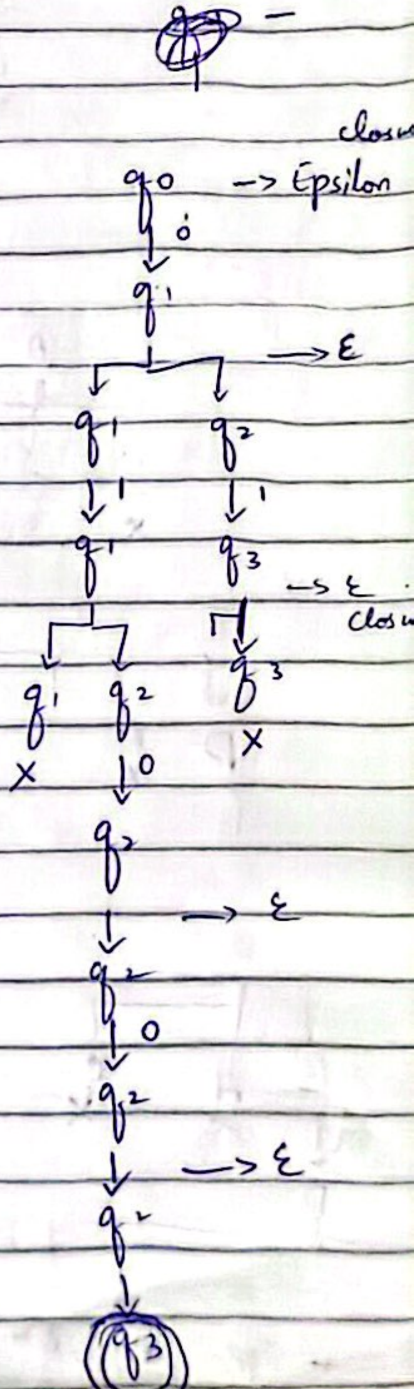


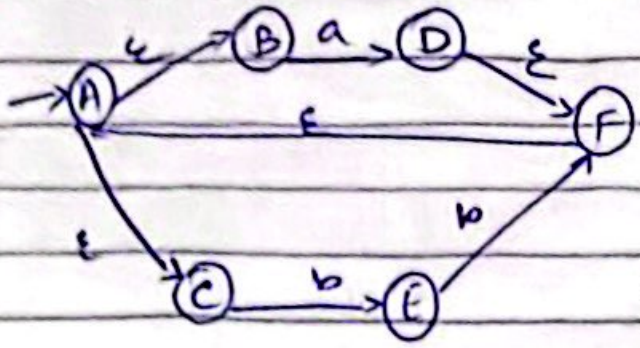
	ε-closure
A	{A, B, C}
B	{B}
D	{D, F, A, B, C}
F	{F, A, B, C}
C	{C}
E	{E}

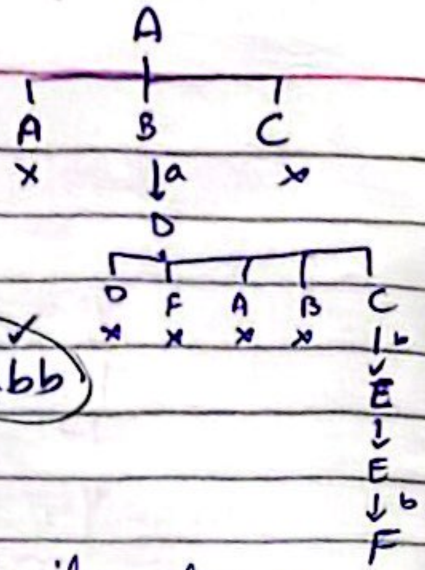
NFA-ε->NULL Tree



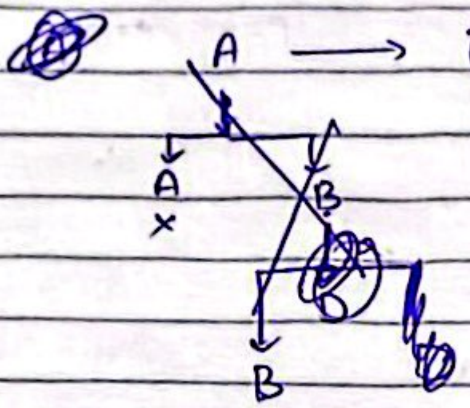




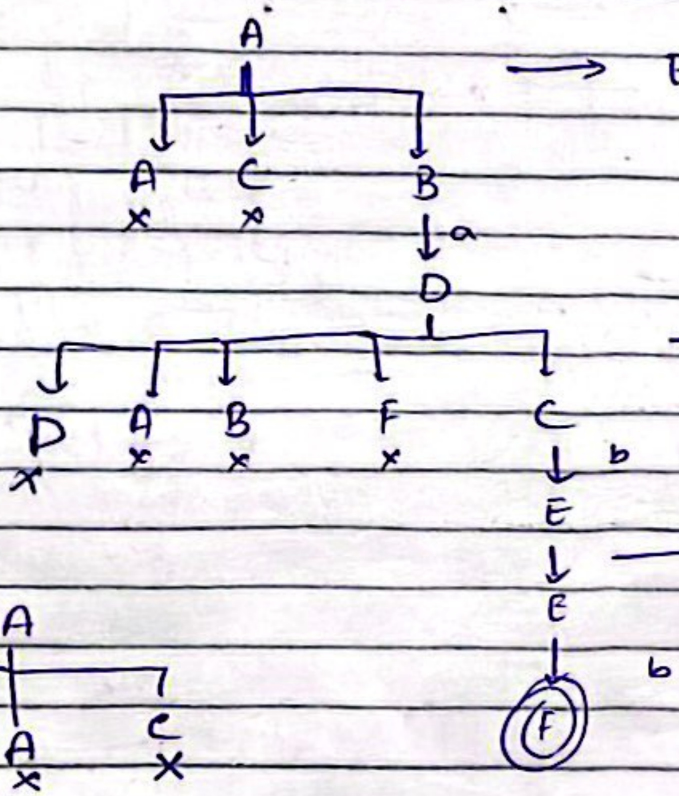




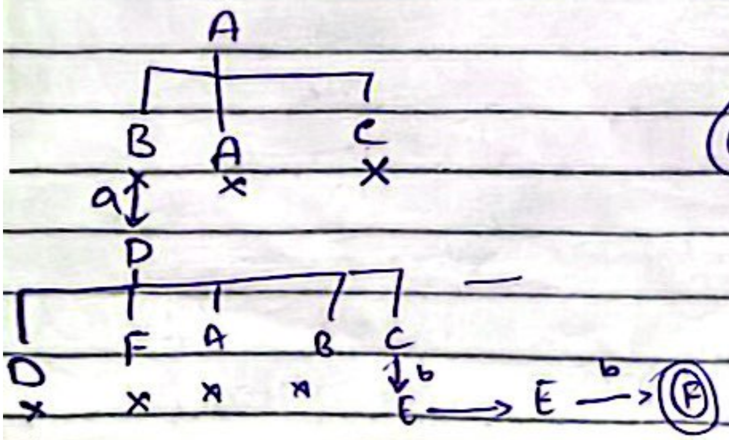
abb



Epsilon closure



Epsilon closure

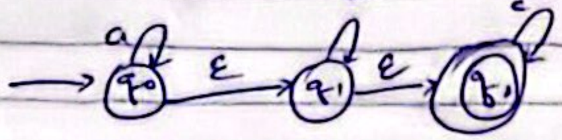


abb

(q_2, a) $q_2 \downarrow q_2, a \downarrow \times$ (q_0, b) $q_0 \downarrow q_0, q_1, q_2, b \downarrow q_1 \downarrow q_1, q_2$

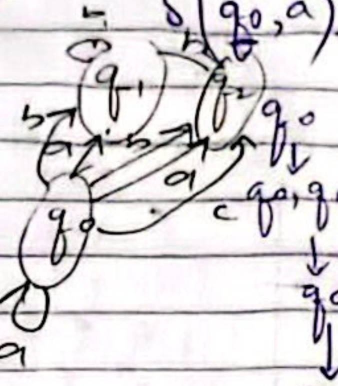
NFA-ε to NFA

$a^* b^* c^*$



	ε-closure
q_0	$\{q_0, q_1, q_2\}$
q_1	$\{q_1, q_2\}$
q_2	$\{q_2\}$

$\delta(q_0, a)$



$q_0 \downarrow q_0, q_1, q_2, a \downarrow q_0 \downarrow q_0, q_1, q_2$

Transition Table for NFA-ε

	a	b	c
q_0	$\{q_0, q_1, q_2$	$\{q_1, q_2\}$	$\{q_2\}$
q_1	—	$\{q_1, q_2\}$	$\{q_2\}$
q_2	—	—	$\{q_2\}$

(q_2, c) $q_2 \downarrow q_2$

$\delta(q_0, a)$

(q_0, b)

q_0, q_1, q_2, b

q_1

$\delta(q_1, b)$
$\downarrow$ ε
q_1, q_2 , b
$\downarrow$
$q_1 \rightarrow$ ε
$\downarrow$
q_1, q_2

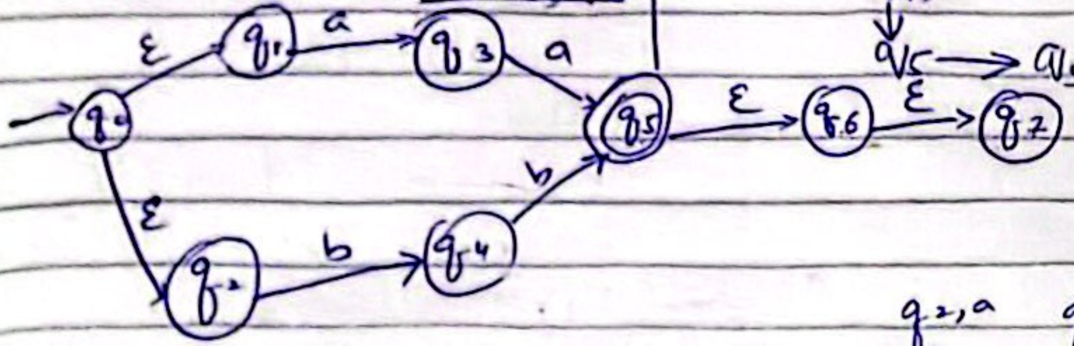
q_0
$\downarrow$
q_0, q_1, q_2 , a
$\downarrow$
q_0
$\downarrow$ → epsilon closure
q_0, q_1, q_2

q_0
$\downarrow$
q_0, q_1, q_2, a
$\downarrow$
q_0
$\downarrow$
q_0, q_1, q_2

$q_0,$
$\downarrow$
$q_0, q_1,$
$\downarrow$
a,
$\downarrow$

(q3, a) q3
↓
q3, a ε closure
↓
q5 → $\boxed{q_5, q_6, q_7}$

q0, q1, q2
q3
(q4, b) q4
↓
q4, b
↓
q5 → q5, q6, q7



q2, a q3, a
q2 q3
q3, b q2, a
↓ q5
q2, b ↓

q4, a
↓
q4, b

	ε-closure
q0	{q0, q1, q2}
q1	{q1}
q2	{q2}
q3	{q3}
q4	{q4}
q5	{q5, q6, q7}
q6	{q6, q7}
q7	{q7}.

q0, a
↓
ε q0, q1, q2
↓
q3
ε ↓
q3

δ (q0, a) = ε-closure
for NFA (δ (ε-closure (q0, a))

q0, a ~~q0, a~~ q0, b
↓ ∅
q0, q1, q2, a
↓ q1, a
q3 ↓
q

	a	b
q0	q3	q4
q1	q3	–
q2	–	q4
q3	q5, q6, q7	–
q4	–	q5, q6, q7
q5	–	–
q6	–	–
q7	–	–

q0, b
↓
q0, q1, q2, b

q1, b q1, a
q1, b ~~q1, a~~
q1, a
↓
q3

Universal

$\{\varepsilon, a, aa, aaa, aaaa,$

TOA 20-02-24

Properties of Regular Set

$U\{\varepsilon, a, aa, aaa$

$U - L_1$

Union:

$RE1 = (aa)^*$ $R.E2 = aa(aa)^*$ $L1 \{\varepsilon, aa, aaa ...\} \cup$

$\{aa, aaa, aaaa, aaaa ...\} \rightarrow \{\varepsilon, a, aa, aaa, aaaa ...\}$ $R.E = a^*$

Intersection:

$R.E1 = (aa)^*$ $R.E2 = aa(aa)^*$ $\{\varepsilon, aa, aaaa ....\} \cap \{aa, aaaa, aaaaaa$

$\rightarrow \{aa, aaaa, aaaaaa ....\}$ $R.E = aa(aa)^*$ or $(aa)^+$

$a($

Compliment:

$R.E = aa(aa)^*$ $L_1 = \{aa, aaaa, aaaaaa ...\}$ $\overline{L_1} = \{\varepsilon, a, a(aa), aaaaa-$

$R.E = (\varepsilon + a(aa)^*)$

Concatenation:

$R.E1 = 0(0+1)^*$ $R.E2 = (0+1)^* 01$ $L_1 = \{0, 00, 01, ...\}$.

$L_2 = \{01, 001, 101, ...\} \rightarrow \{001, 0001, 0101, ...\}$ $R.E = 0(0+1)^*$

Reverse:

$R.E = 10 + 01 + 11$ $L = \{10, 01, 11\}$ $L^R = \{01, 10, 11\}$

$R.E = 01 + 10 + 11$

Minimization of DFA

→ MDFA is unique for every language.

→ Productive states. Its presence and absence effect the accepting. the capability of the machin

→ Non Productive State:

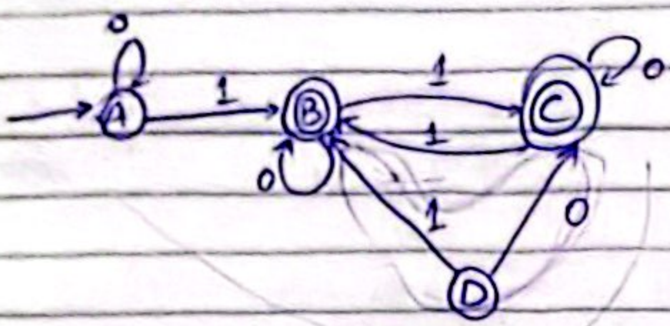
: Dead State (Is se koi answer ni jayega)

: Unreachable state (Bhakchodis State)

: Equal State (If 'a' of both states goes to same & if 'b' of " " " " state)

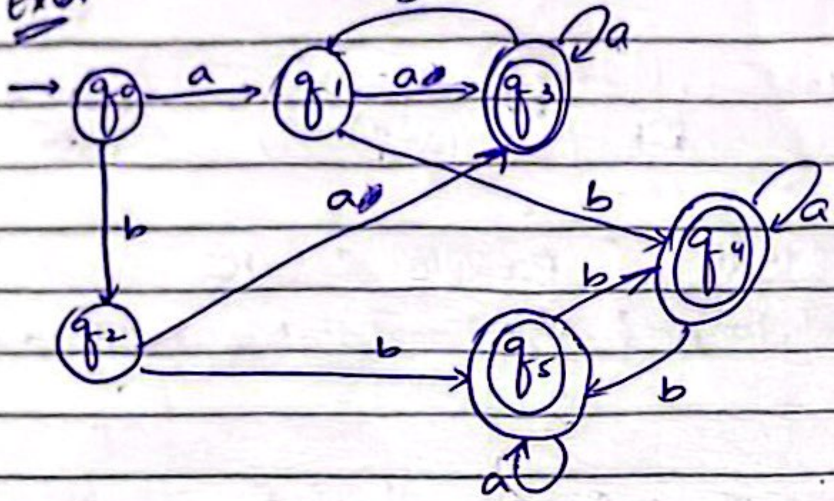
Ex 1

Given



D is unreachable
so remove it

Ex 2



Ex 1 :

{A} {B,C}

{A} {B,C}

Make initial state single and final state single

check krna hai k transition krne k bad same state main hai ya nahi.

Ex 2 :

Non Final

{q0, q1, q2}

{q0} {q1, q2}

{q0} {q1, q2}

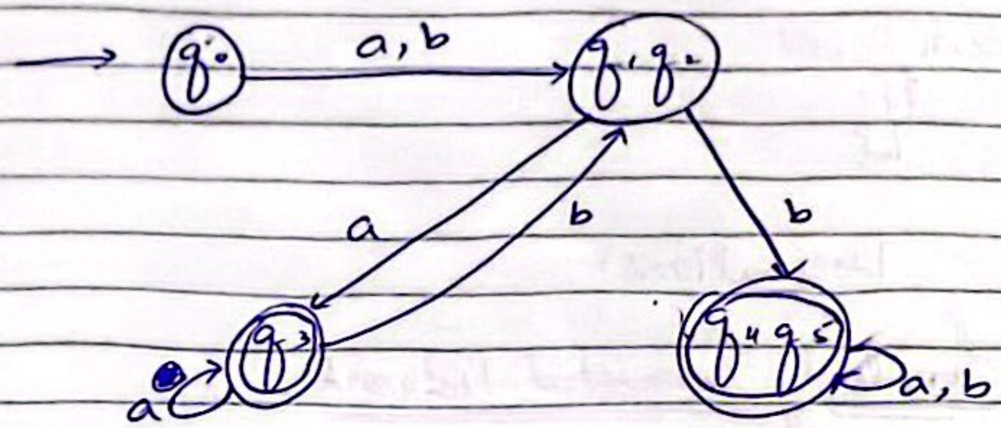
Final

{q3, q4, q5}

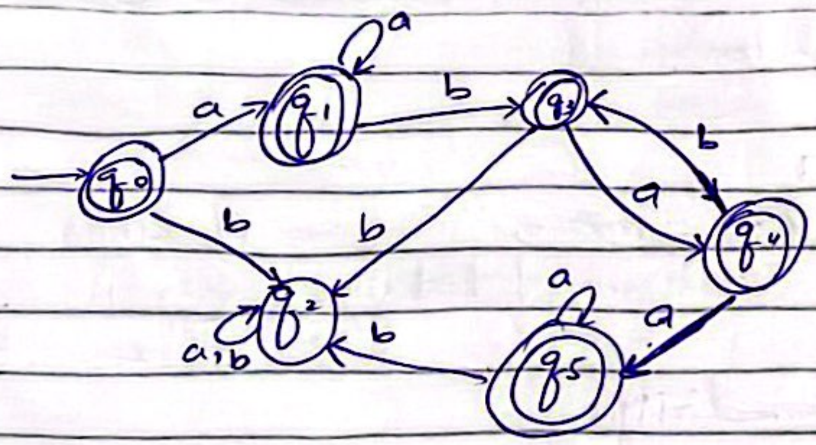
{q3} {q4, q5}

{q3} {q4, q5}

Optimal ans MDFA.



Example 3:



Non Final states	Final states
$\{q_2\}$	$\{q_0, q_1, q_3, q_4, q_5\}$
$\{q_2\}$	$\{q_0\}\{q_3\}\{q_1, q_4\}\{q_5\}$
	$\{q_0\}\{q_3\}\{$

(iv): **Bisection Width:**

Bisection Width of complete binary tree is 1.

TOA 22-02-24 1

<u>Language to Regular Expression:</u>

<u>Symbols</u>

( )

a − z, ε

0 − 9

(A+B)

→ OR

A^+ → Kleen plus

A^* → Kleen star

{∧} ∧/ε

{0} 0

{0, 01} (0+01)

{ε, a, aa, aaa, ...} a^*

{ε, a, b, ab, ba, ...} $(a+b)^*$

$\{10, 01, 111, 1011\}^*$ $(10+01+111+1011)^*$

$a(a+b)^* a$ $\quad$ $(aab$

$\epsilon + \{a, b\}$ $(a(a+b)^* b + b(a+b)^* a)$

: Start with ab :

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

: substring aab :

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

: start and ends with a :

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

: start and ends with different symbols :

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

$|w| = 2$:

$$(a+b)^2 \;/\; (a+b)(a+b) \;/\; (aa+ab+ba+bb)$$

$|w| \geq 3$:

$$(a+b)^3 (a+b)^*$$

$(a+b)^3 \cancel{a} (a+b)^*$

$(a+b)^3 + (a+b)^2 + (a+b)^1 + (a+b)^0$

$(a+b+\epsilon)^3$

$|w| \leq 3$:

$$(a+b)^0 + (a+b)^1 + (a+b)^2 + (a+b)^3 \;/\; (a+b+\epsilon)^3$$

$aa(aa)^*$

$|w|_a = 2$:

$$b^* a \quad b^* a b^*$$

$b^* + b^*ab^* + b^*ab^*ab^*$

: $|w|_a \le 2$:

$\cancel{b^* + b^*ab^* + b^*ab^*}$

$b^* + b^*ab^* + b^*ab^*ab^*$

: string do not contain 01 :

1^*0^* $(10)^*$

a a b b

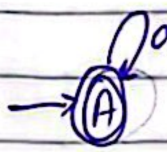
: shortest string that is not in the language

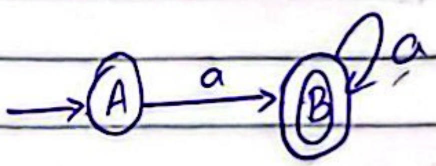
$a^*(ab)^*b^*$:

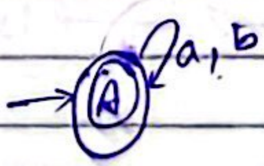
aabb → ba.

Convert R.E to Finite Machine :

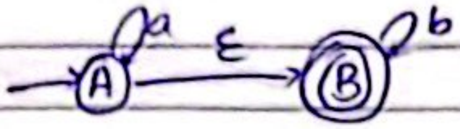
: ε : 

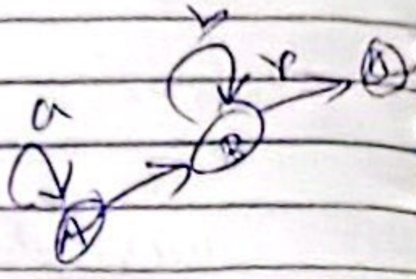
: a^* : 

: a^+ : 

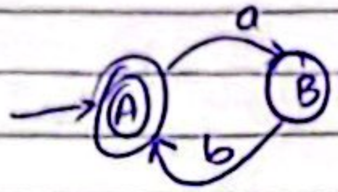
: $(a+b)^*$: 

: a^*b^* :

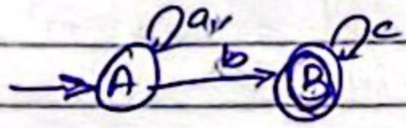




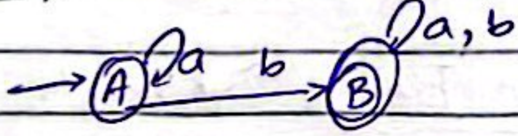
: $(ab)^*$:



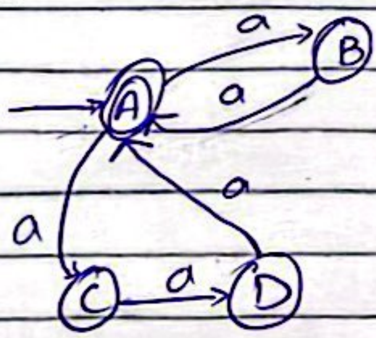
: $a^*b^*c^*$:



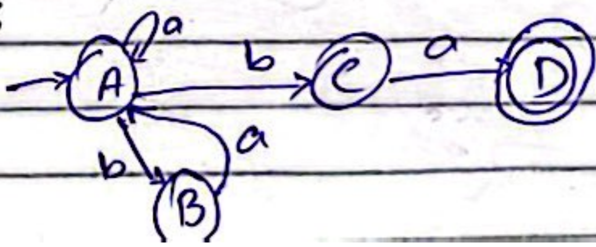
: $a^*b(a+b)^*$:



: $(aa+aaa)^*$:

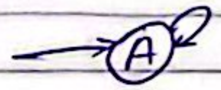


: $(a+ba)^*ba$:

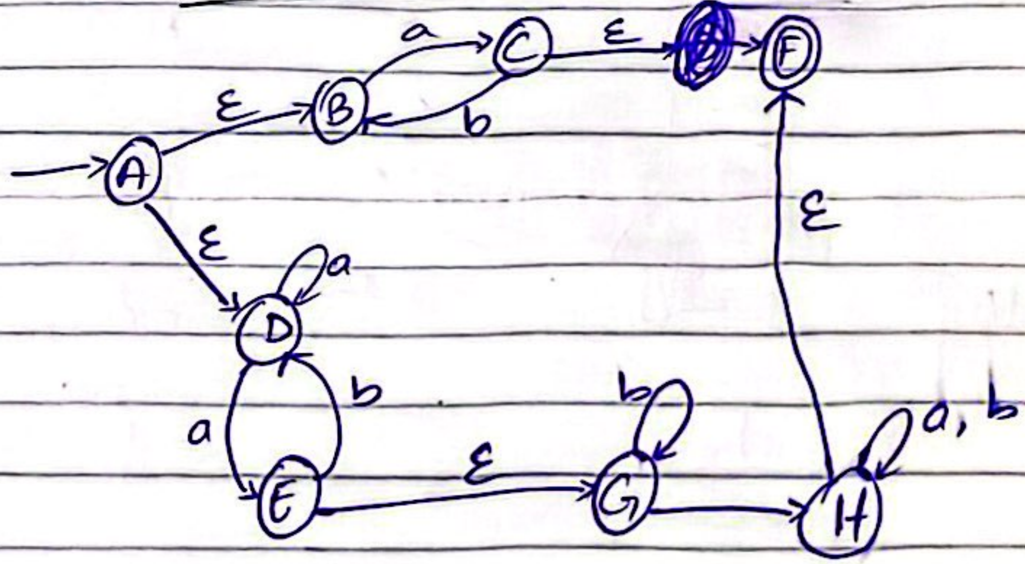


Scanned with CamScanner

$\div$ $(a+ba(a+b))^* a (ba)^* b$:

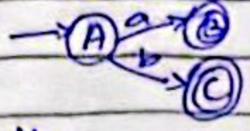
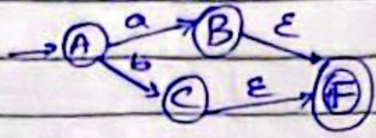


$=$ $(ab)^* + (a+ab)^* b^* (a+b)^*$:



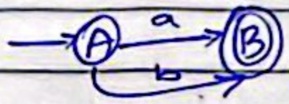
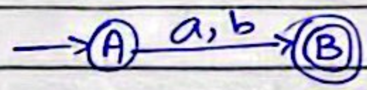
Step 3:

If there is more than one Final state convert all Final state into non Final and make a one new Final state

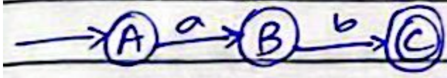
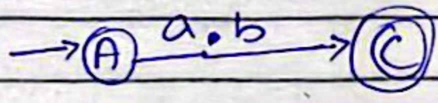



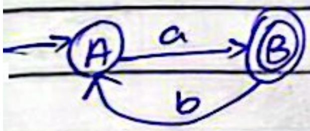
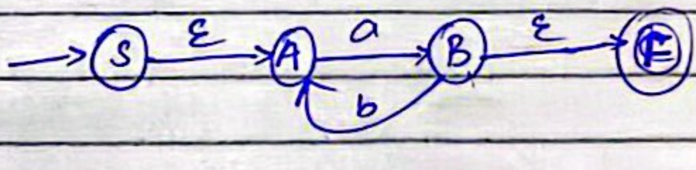
Step 4:

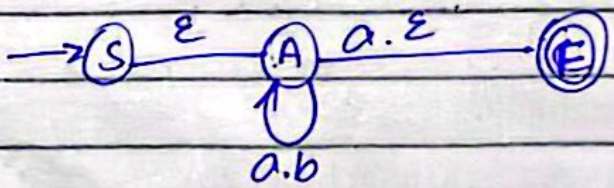
Eliminate all states one by one except initial and Final state

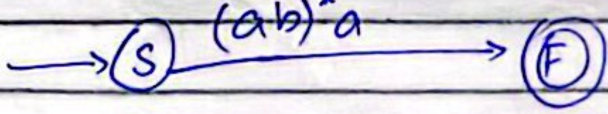



$(a+b)$

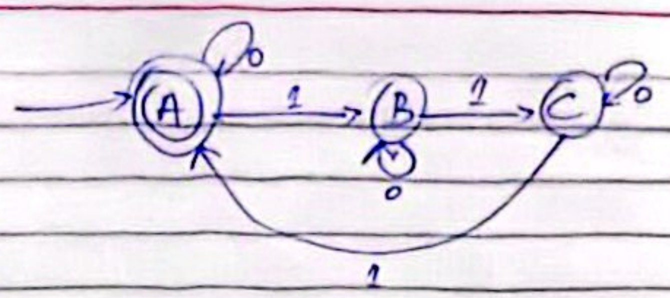



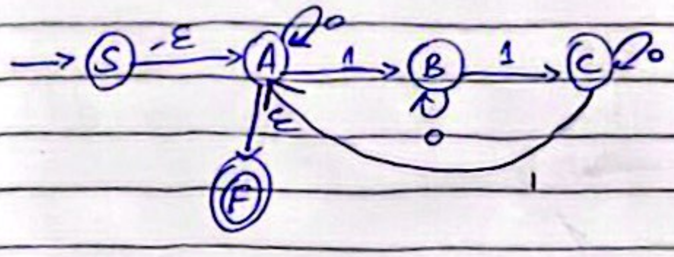


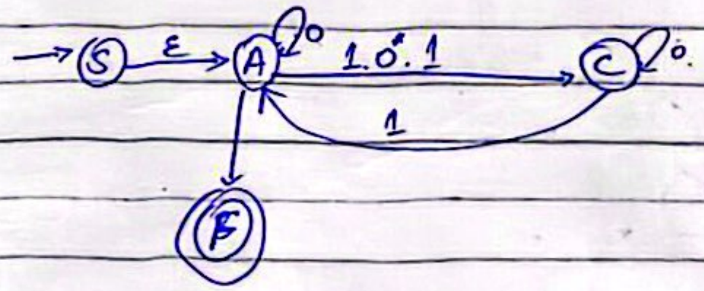


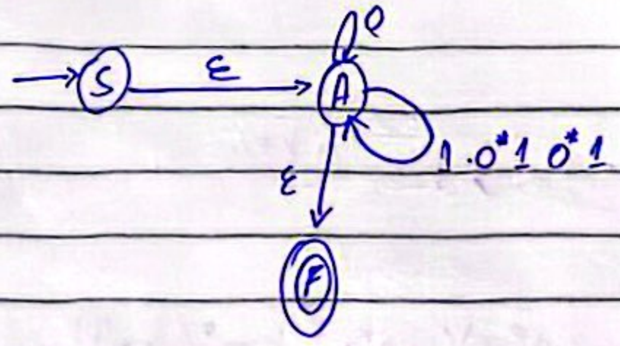
$\varepsilon.(ab)^*a.\varepsilon = \varepsilon.(ab)^*a$

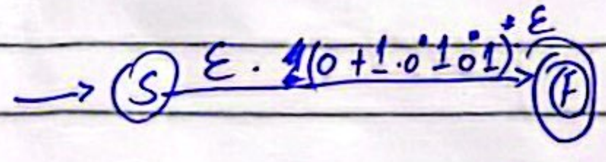
$(ab)^*a$


A
B
C
1
1
0
0
0
1


S
A
B
C
F
ε
ε
1
1
0
0
0
1


S
A
C
F
ε
0
1.0*.1
0.
1


S
A
F
ε
ε
0
1.0*1 0*1


S
F
ε . 1(0+1.0*1 0*1)*. ε

$(0+1.0^*10^*1)^*$.

TOA 5-03-24

STATE ELIMINATION METHOD

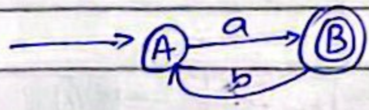
→ Every finite automata has an equivalent Regular Expression.

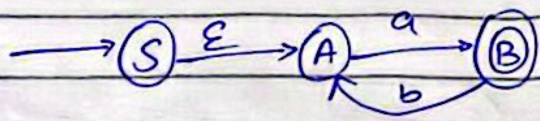
→ The conversion of FM to Regular Expression is done through the State Elimination Method

The state elimination method follow the genera set of Rules.

Step 1 :

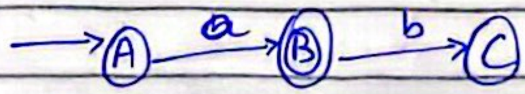
The initial state of FM must not have any incoming edge.

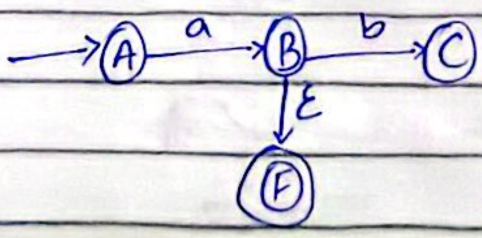




Step 2:

The final state of FM must not have a outgoing edge

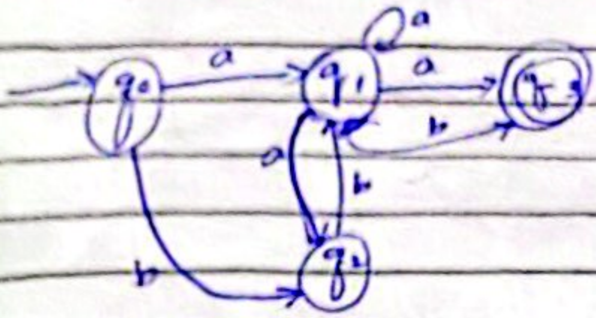


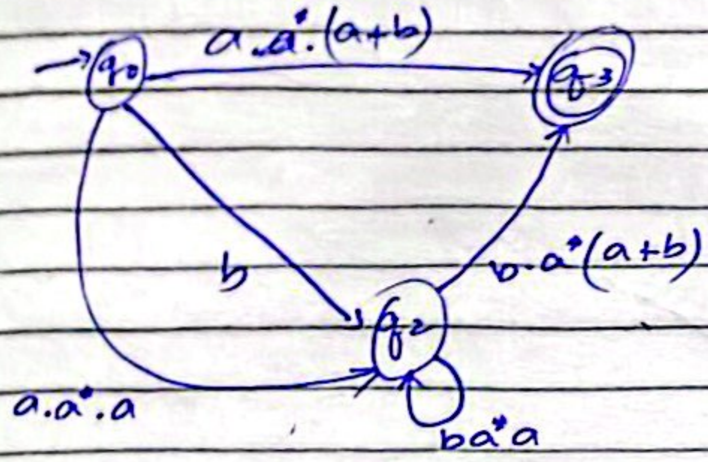


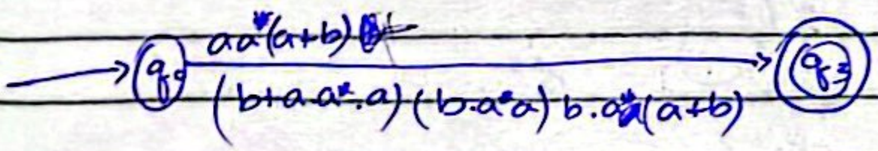
b follow the aaa

baaa

aaab

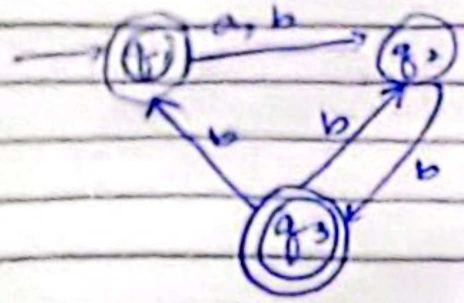




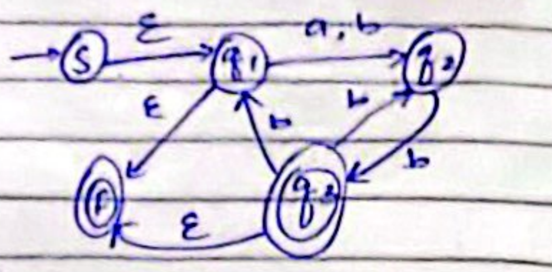


$$aa^*(a+b)+(b+a.a^*a)(b.a^*a)^* ba^*(a+b)$$

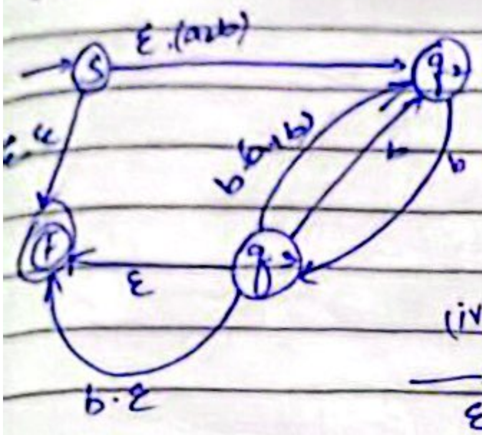
TOA 7-08-24



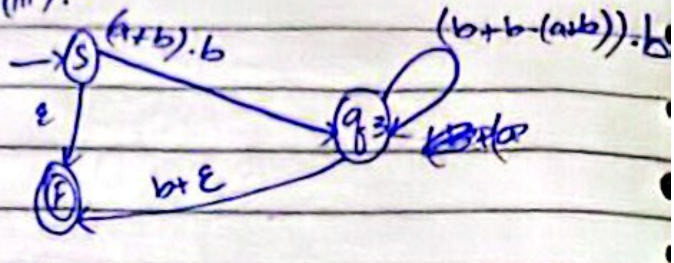
(i)



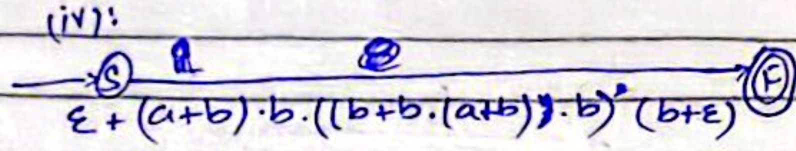
(ii)



(iii):



(iv):

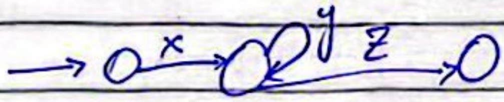


Pumping Lemma :

Pumping Lemma is used to prove that language is not regular.

→ And we cannot use for regular language.

XY'Z



: Regular language hold the Property of Pumping Lemma.

Rules For Pumping Lemma:

→ Divide the string into three parts.

XY'Z

can be Epsilon ↳ can be epsilon.

(ii): $i \geq 0$ for $\forall$ string $\in L$

(iii): $|Y| > 0$ (NOT Epsilon)

(iv): $|XY| \leq P$
↳ Pumping Length.

FM → Pumping Length → No of states.

For irregular language → pumping length → P

Example: irregular language

$0^n, 1^n$ $0^n 1^n$
convert into Pumping length

Divide into $x y^i z$
$= q \; p = q + r$

$x = \varepsilon$	$x = 0^q$	$x = 0^p$	
$y^i = 0^p$	$y^i = 0^r$	$y^i = 0^p$	
$z = 1^p$	$z = 1^p$	$z = \varepsilon$	ye nahi le
$\varepsilon . 0^p . 1^p = 0^p 1^p$	$0^q 0^r 1^p$	$0^p 0^p . \varepsilon$	sakte bcz
$\lvert \varepsilon . 0^p \rvert = \lvert 0^p \rvert$	$0^{q+r} 1^p$	0^{2p}	4th rule does
$\leq P$	$0^p . 1^p$	$2p$	not hold.

xy^iz $\varepsilon(0^p.) 1^p$

@ $i = 0$	$i = 2$
$(0^p)^0 . 1^p$	$0^{2p} 1^p$
1^p	$0^p 0^p . 1^p \notin L$
$1^p \notin L$	

Example 2:

$L = \{ x \mid x \in \Sigma^* \text{ and } x = a^i b^j \text{ where } i \neq j \text{ and } 2i \neq j \}$

$\boxed{a^{p+1} b^p}$

$q = n+1 \quad q+2$

$x = a^{p-s}$

$y^i = a^s$ $\quad$ s is some constant

$z = ab^p$ $\quad$ $s > 0$

$\overset{x}{a^{p-s}} \quad \overset{y^i}{(a^s)^i} \quad \overset{z}{ab^p}$

$a^{p+1} b^p$

$a^p . a . b^p$

$i = 0$

$a^{p-s} ab^p$

$x = a^{p-s}$

$y = a^s$

$z = ab^p$

$s = 1$

$a^{p-1+1} b^p$

$\boxed{a^p b^p}$

$i = j$

$\leq p$

$x = a^{p-s}$

$y = a^s$

$|xy| \leq p$; $|y| > 0$

$a^{p+1} b^p$, $i \neq j$

$+1)-s \; b^{p+3}$

$-s . a^s \boxed{b^p}$

$y \quad z$

$a^{p-s} a^s b^p$

$a^{p-s+s} b^p$

TOA

12-03-24

Grammar CFG

→ Grammar is formally described into 4 tuples

$G = (V, T, S, P)$

V = set of variables and non terminal (Always capital letter)

T = set of terminal (Alphabets)

S = ~~set of~~ Starting point.

P = Production ~~rules~~ rule.

$S \rightarrow AB$

$A \rightarrow a$

$B \rightarrow b$

↳ Production

$H \rightarrow B$

↓ Head of rule

↓ Body of rule

$H \in V$

$B \in \{V \cup T\}^*$

01

0011

000111

0S1 → 0S1 → 0S1 → ε

$L = \{0^P 1^P \mid P \geq 0\}$

ε, 01, 0011

$S \rightarrow \varepsilon \mid 0S1$

ε

01

0S1 → ε

0S1 → 0S1 → ε = 0011

0S1 → 0S1 → 0S1 → ε = 000111

Left Most Derivation (LMD) : In a left most derivation always the left most non-terminal is replaced.

string 00011

$L = \{ w \mid w^R \text{ is palindrome} \}$ | $L = \{ w \mid w \text{ is not palindrome}$

$S \to \varepsilon \mid 0 \mid 1 \mid 0S0 \mid 1S1$

$S \to T \mid 0T1 \mid 1T0$
$0S0 \mid 1S1$.
$T \to \varepsilon \mid 0T \mid 1T$

0010
1101
001
100
01
10

01010

0S0
↓
1S1
↓
1T0
↓
ε

1T
↓
1T0
↓
ε

0S0
|
0T1

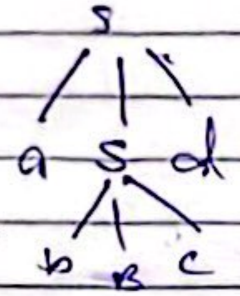
10
01

Ambigious Grammar :

If for the same string there is two LMD and RMD or two parse tree.

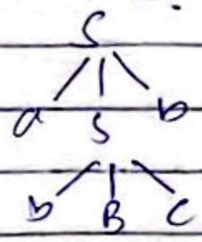
Right Most Derivation (RMD):

In a RMD always the right most ~~derivation~~ non-terminal is replaced.



$S \rightarrow aSd \mid B \mid \lambda$

$B \rightarrow bBc \mid \lambda$



$L = \{a^i b^j c^j d^i,\ i, j \geq 0\}$

ε	0
bcd	1
abbccdd	$i=2, j=2$
aad	$i=1, j=0$
bc	$i=0, j=1$
aadd	$i=2, j=0$

abcd

$S \rightarrow aSd \mid P$; $P \rightarrow bPc \mid \varepsilon$

abcd

aSd
↓
bSc
↓
ε

abcd

aadd

aSd
↓
aSd
↓
ε

aadd

abcd

$S \rightarrow aSd \mid P$

$P \rightarrow bPc \mid \varepsilon$

TOA 19-03-24

CFG (Context Free Grammar) CNF (Chomsky Normal Form)

(i): Removal of Null Production:

In a CFG a non-terminal Symbol 'A' is nullable, if there exist a null production '$A \to \varepsilon$' or there is a derivation that start at 'A' and lead to Epsilon.

ex 1:	ex: 2	example 3
$S \to ABAC \mid \varepsilon$	$S \to AbA$	$S \to aAa \mid bBb \mid$
$A \to aA \mid \varepsilon$	$A \to Aa \mid \varepsilon$	$A \to Ca \mid bCaCb$
$B \to bB \mid \varepsilon$	$\because A \to \varepsilon$	$B \to Cb \mid C$
$C \to c$	$S \to AbA \mid bA \mid Ab$	$C \to ab \mid \varepsilon$
$\because A \to \varepsilon$.	$\mid b$	$\because C \to \varepsilon$
$S \to ABAC \mid \varepsilon \mid BAC \mid ABC \mid BC$	$A \to Aa \mid a$.	$S \to aAa \mid bBb \mid \varepsilon$
$A \to aA \mid a$		$A \to Ca \mid bCaCb \mid a \mid bac$
$B \to bB \mid \varepsilon$		$bCab \mid bab$.
$C \to c$ next page.		$B \to Cb \mid C \mid b \mid \varepsilon$
		$C \to ab$. next

(ii) Removal of Unit Production:

→ Any Production rule of the form $\overset{①}{A} - \overset{①}{B}$ where $A, B \in$ Non terminal Symbol is called unit Production.

$S \to XY$	$S \to 0A \mid 1B \mid C$	$A \Rightarrow$
$X \to a$	$A \to 0S \mid 00$	$A \to$
$Y \to Z \mid b$	$B \to 1 \mid A$	$A -$
$Z \to M$	$C \to 01$	
$M \to N$		
$N \to a$	$S \to 0A \mid 1B \mid 01$	
$S \to XY$	$A \to 0S \mid 00$	
$X \to a$; $M \to a$	$B \to 1 \mid 0S \mid 00$	
$Y \to a \mid b$; $N \to a$	$C \to 01$.	
$Z \to a$		

Scanned with CamScanner

example 2.

$\because B \rightarrow \varepsilon$

$S \rightarrow aAa \mid bBb \mid \varepsilon \mid bb$

$A \rightarrow Ca \mid bCaCb \mid a \mid baCb \mid bCab \mid bab$

$B \rightarrow Cb \mid C \mid b.$

$C \rightarrow ab$

example 2 :

$\because B \rightarrow \varepsilon$

$S \rightarrow ABAC \mid \varepsilon \mid BAC \mid ABC \mid \boxed{BC} \mid AAC \mid AC \mid C$

$A \rightarrow aA \mid a$

$B \rightarrow bB \mid b$

$C \rightarrow C$

Example:

$S \rightarrow aA \mid bB$

$A \rightarrow B$

$B \rightarrow Cb \mid ab \mid C$

$C \rightarrow b$

$S \rightarrow aA \mid bB$

$A \rightarrow Cb \mid ab \mid b$

$B \rightarrow Cb \mid ab \mid b$

$C \rightarrow b$

Scanned with CamScanner

(iii): Removal of Useless Symbol:

→ Non Generating
→ Non reachable

$S \to AB \mid a$

$A \to BC \mid b$

$\times\ B \to aB \mid C$

$\times\ C \to aC \mid B$

terminals dekhne wo likhne pehle

$\{a, b, S, A\}$

phir unki production, wo terminals akele ya unke combinations.

$\{a, b, S, A\}$

$S \to \cancel{AB} \mid a$

$A \to \cancel{BC} \mid b$

: agar agay grammar main bracket main se koi combination nahi araha to usay remove kar dein ge.

$S \to AB$, $A \to b$, B ×

B pehle hi remove hogya tha to ab combination bhi remove karain ge. BC bhi isi tarha.

$S \to a$

$\cancel{A \to b}$

$\boxed{S \to a}$

starting ko dekhen ge ab, arrow k bad wala agar kahin na jaye to bas starting ko rakhen ge.

}

example,

~~S~~

$\{a, b, A, B, S\}$

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

$\{a, b, A, B, S\}$

$S \to AB \mid \cancel{AC}$
$A \to aAb \mid bAa \mid a$
$B \to bbA \mid aaB \mid AB$
× $C \to abCA \mid aDb$
× $D \to bD \mid aC$

$S \to AB$
$A \to aAb \mid bAa \mid a$
$B \to bbA \mid aaB \mid AB$

✓

Scanned with CamScanner

$A \to CX \mid a \mid \varepsilon$

Chomskey Normal Form :

→ Normal form for a CFG.

→ All production rule of the form $A \to BC \mid \varepsilon \mid \varepsilon$ single alphabet

A, B, C are non-terminal and B, C is not a start symbol. $B \to \varepsilon$

→ If $\varepsilon \in L(G)$ then $S \to \varepsilon$ ('S' is starting point)

Following set of rules need to Follow to convert CFG → CNF

(i): Add a new start symbol ($S_0 \to S$) → Not a start symbol
(II): Eliminate all epsilon (null) production ($A \to \varepsilon$)
(iii): Eliminate all unit production ($A \to B \mid C$)
(iv): Eliminate all useless symbol → Non Generating → Non Reachable.
(V): Convert all production of the form.

$A \to BC \mid \varepsilon$

$A \to a$

★ If there is a start symbol on the right side of the CFG than we need to add the new start symbol.

Example:

$S \to BSB \mid B \mid \varepsilon$

$B \to 00 \mid \varepsilon$

→ Add a new start symbol.

$S_0 \to S$

$S \to BSB \mid B \mid \varepsilon$

$B \to 00 \mid \varepsilon$

→ Eliminate epsilon Production.

$\because S \to \varepsilon$

$S_0 \to S \mid \varepsilon$

$S \to BSB \mid B \mid BB$

$B \to 00 \mid \varepsilon$

$\because B \to \varepsilon$

$S_0 \to S \mid \varepsilon$

$S \to BSB \mid B \mid BB \mid SB \mid BS \mid \varepsilon \mid S$

$B \to 00$

$S_0 \to S \mid \varepsilon$

$S \to BSB \mid B \mid BB \mid SB \mid BS \mid \cancel{\varepsilon}$

$B \to 00$

→ Eliminate unit Production.

$S_0 \to S \mid \varepsilon$ $\qquad$ $S_0 \to S$

$\because S \to B$ $\qquad$ $S \to BSB \mid 00 \mid BB \mid SB \mid BS \mid \cancel{\varepsilon}$ $\qquad$ $S \to B$

$B \to 00$ $\qquad$ $S \to S$

$\because S_0 \to S$ $\qquad$ $S_0 \to BSB \mid 00 \mid BB \mid SB \mid BS \mid \varepsilon$

$S \to S$ $\qquad$ $S \to BSB \mid 00 \mid BB \mid SB \mid BS$ $\qquad$ $B \to 00$

$S_0 \rightarrow BSB \mid 00 \mid BB \mid SB \mid BS \mid \varepsilon$

$S \rightarrow BSB \mid 00 \mid BB \mid SB \mid BS$

$B \rightarrow 00$

$A \rightarrow CX \mid a \mid \varepsilon$

Add new Production.

$P \rightarrow BS$

$T \rightarrow 0$

~~00~~ $\rightarrow a$

TT

$S_0 \rightarrow PB \mid TT \mid BB \mid SB \mid BS \mid \varepsilon$

$S \rightarrow PB \mid TT \mid BB \mid SB \mid BS$

$B \rightarrow TT$

$P \rightarrow BS$

$T \rightarrow 0$

$S \rightarrow ASA \rightarrow aB$

$A \rightarrow B \mid S$

$B \rightarrow b \mid \varepsilon$

$C \rightarrow bbC \mid CC$

$D \rightarrow ab \mid aa$

TOA

26-03-24

1|2|3|4
(1,1) (2,4)
(1,2) (3,4)
(1,3) (4,4)

CYK Algorithm:

→ Tells us whether the given string is the part of the Grammar.

Grammar → | CYK | → Yes/No
string →

→ Only applicable for CNF.

(1,4) = 1|2|3|4
(1,1) (2,4) = ∅
(1,2) (3,4) = ∅
(1,3) (4,4) = CB

$S_0 \rightarrow CB|AB|\varepsilon$
$S \rightarrow CB|AB$
$A \rightarrow 0$
$B \rightarrow 1$
$C \rightarrow AS$

string = 0011
1234

	4	3	2	1
1	S_0, S	C	∅	A
2	∅	S_0, S	A	×
3	∅	B	×	×
4	B	×	×	×

(1,3) = 1|2|3
(1,1) (2,3) = AS_0, AS
(1,2) (3,3) = ∅

1|2|3
(1,1) (2,3)

(2,4) = 2|3|4
(2,2) (3,4) = ∅
(2,3) (4,4) = S_0B, SB

Yes
(S_0, S) (B)

AS_0, AS
BS_0, BS

i): If $w = \varepsilon$, then need to check if there is production '$S \rightarrow \varepsilon$' then accept Yes else No.

ii): Construct $n \times n$ table.

iii): Fill the above diagonal one by one.

$T(i, j) = T(i, K) = A$
$\& \; T(K+1, j) = B$

so, if there is a production '$S \rightarrow AB$', then
$T(i, j) = S$

iv): If the starting symbol 'S' is on the $T(1, n)$, it means string generated else No.

- pehla diagonal fill karna string dekh k uski numbering karke.
- dusra diagonal fill karna adjacent combinations ko dekh k upar se niche
- teesra sir ke method se

CNF dekhne k liye Starting symbol right pe na ho. non terminal do hon akathe. Terminal akela ho.

Example:

$S \rightarrow AB \mid BC$

$A \rightarrow BA \mid a$

$B \rightarrow CC \mid b$

$C \rightarrow AB \mid a$

string

baaba
1 2 3 4 5

AB, CB

(1,5) = 1,2 | 3,4,5
= (1,1)(2,5) =
= (1,2)(3,5) =
= (1,3)(4,5) =
(1,4)(5,5) =

(A,C) (A,S)
AA, AS, CA, CS

	5	4	3	2	1
1	S, A, C	∅	∅	S, A	B
2	S, A, C	B	B	A, C	X
3	B	S, C	A, C	X	X
4	A, S	B	X	X	X
5	A, C	X	X	X	X

Yes.

BA, BC
AA, AC, C
AB, CB
BA, B

(2,5) = 2, 3 | 4 | 5
= (2,2)(3,5) = AB, CB
= (2,3)(4,5) = BA, BS
= (2,4)(5,5) = BA, BC

(1,3) = 1 | 2 | 3
= (1,1)(2,3) = BB,
(1,2)(3,3) = SS, SA, AS, AA

(3,5) = 3, 4, 5
(3,3)(4,5) = AA, AS, CS,
(3,4)(5,5) = SA, SC, CA

(1,4) = 1, 2, 3, 4
= (1,1)(2,4) = BB
= (1,2)(3,4) = SS, SC, AS, A
(1,3)(4,4) = ∅

(2,4) = 2, 3, 4
= (2,2)(3,4) = AS, CS, AA, CA
= (2,3)(4,4) = BB

TOA 28-4-24

PDA (Push Down Automata):

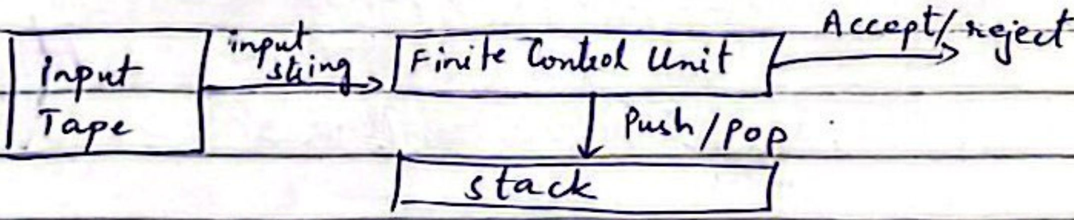
→ Way to implement CFG
→ More powerful than FSM.
→ PDA: FSM + "a" stack".

Three components:

- Input tape.
- Control unit.
- Stack with unlimited memory.

Operations:

Push
Pop.

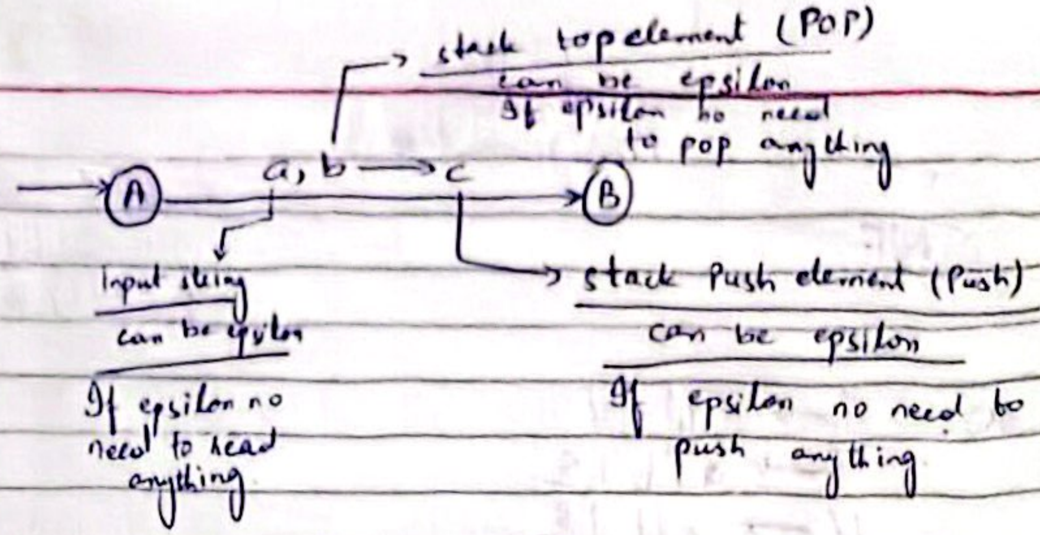


7-tuples

stack top elements

$(Q, \Sigma, \delta, q_0, F, S/\Gamma, I/Z_0)$

stack symbols

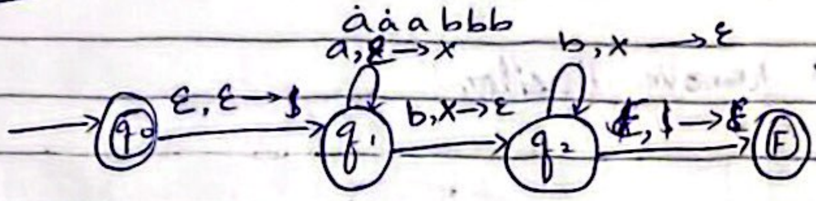


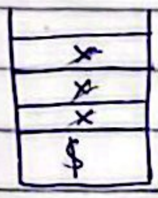
ε, ab, aabb

example 1 : $a^n b^n \quad n \geq 0$

aaabb

aaabbb

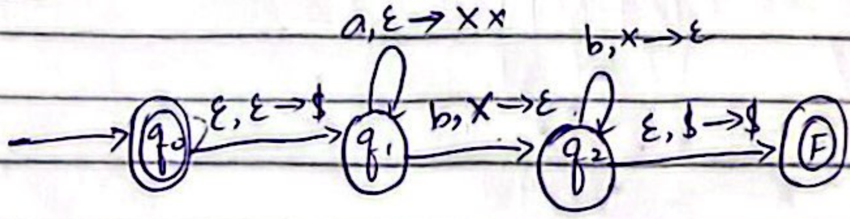




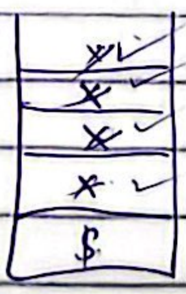
a, ε → x

b, x → ε

example 2 : $a^n b^{2n} \quad n \geq 0$, 0, abb , aabbbb



aabbbb



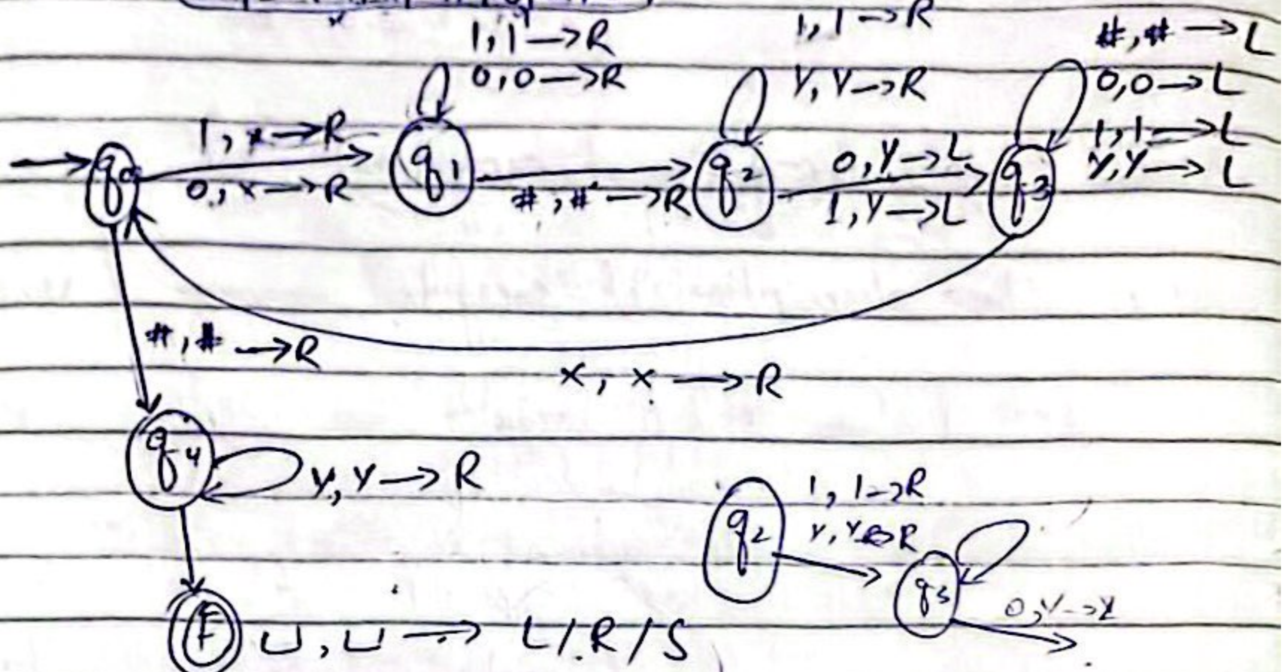
TOA 25-4-24

Single Tape **TURING MACHINE**

$L = \{x\#y \mid x \in \{0,1\}^*, y \in \{0,1\}^*, |x| = |y|\}$

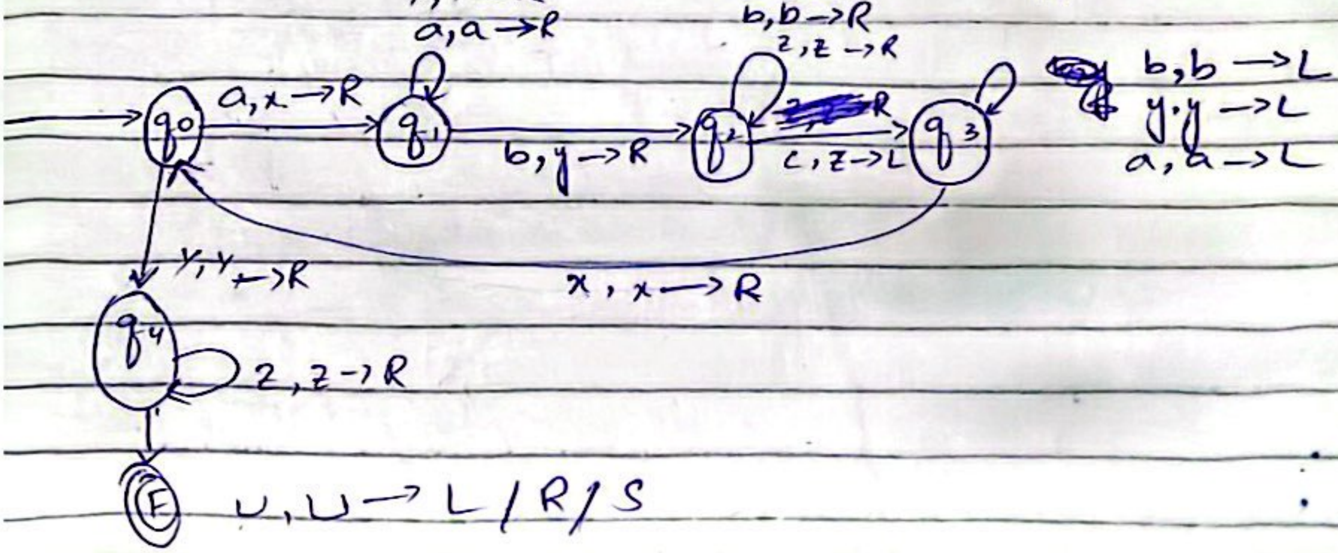
⊔	⊔	1	0	#	1	0	⊔	⊔

⊔⊔ 00 # 10 ⊔⊔



$L = \{a^i b^j c^k$ where $i \geq 1$, $j \geq 1$ and $k = \min\{i, j\}\}$

⊔⊔aabc⊔⊔ ⊔⊔aabbbcc⊔⊔



$A = \{a, b\}$

$a, b \to L$

TOA 7-05-24

$(\Delta\Delta\Delta\Delta)$
$(\Delta\Delta\Delta\Delta)(RRRR)$

$\Delta a b a$

MultiTape Turing Machine

T_1

Δ	a	a	b	Δ	Δ	....

T_2

Δ	a	Δ	.....

T_3

Δ	b	a	Δ	....

T_4

Δ	a	Δ	Δ	....

T_4 will be

Δ	a	a	b	#	a	#	b	a	Δ	Δ	—

$(A, a, b, \Delta)\ (A, a, b, A)\ (R, S, S, R)$

$a^i b^j c^{i+j}$ $(A,$

Δ	a	a	Δ

Δ	b	b	b	Δ

Δ	c	c	c	c	c	Δ

T_1	Δ	a	a	b	Δ	Δ	_

$A = \{a, b\}$

$B = \{a, b\}$

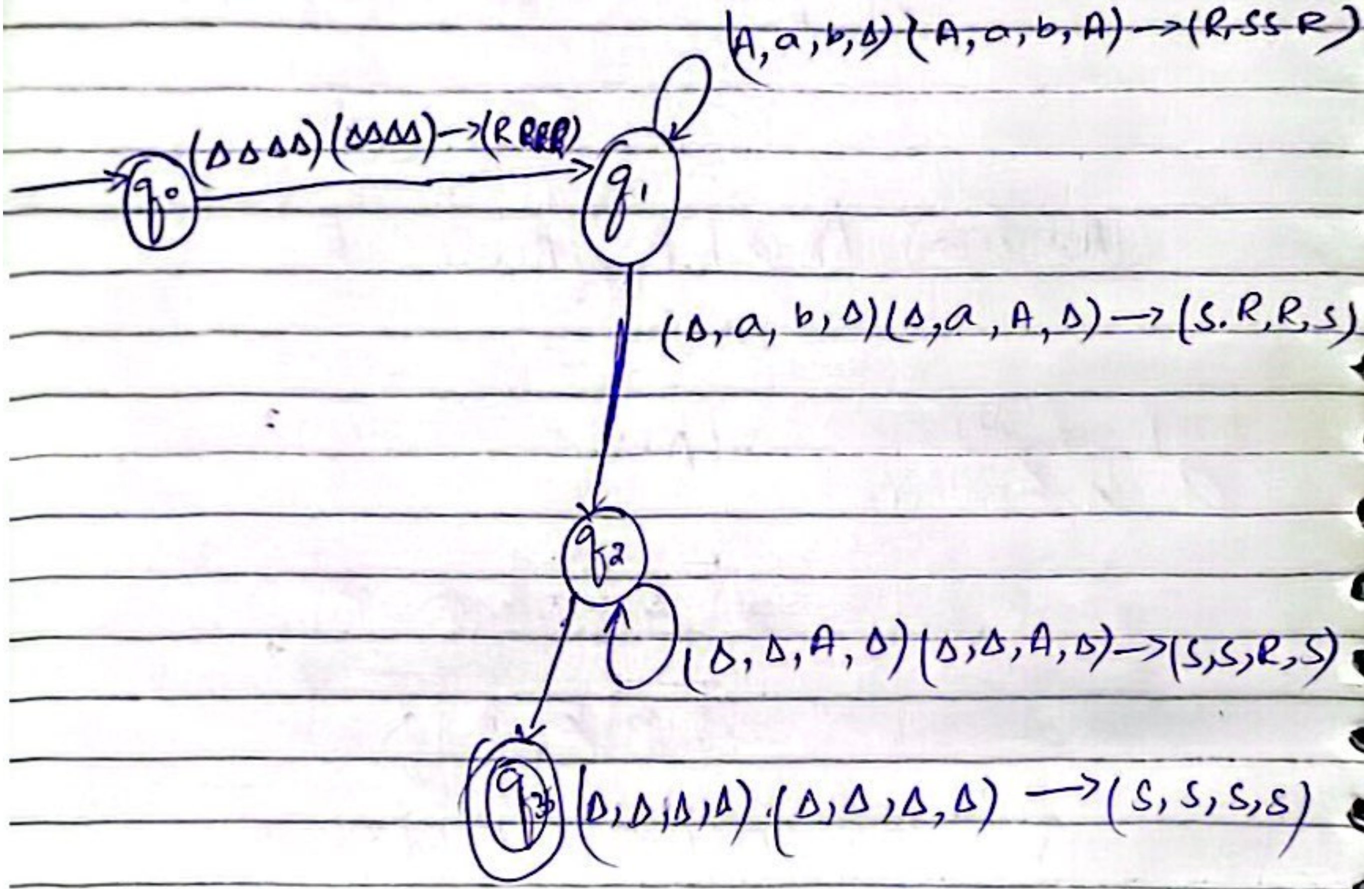
T_2	Δ	a	Δ	Δ	----

T_3	Δ	b	a	Δ	---

T_4	Δ	a	Δ	Δ	----

T_4 will be

Δ	a	a	b	#	a	#	b	a	Δ	Δ	----

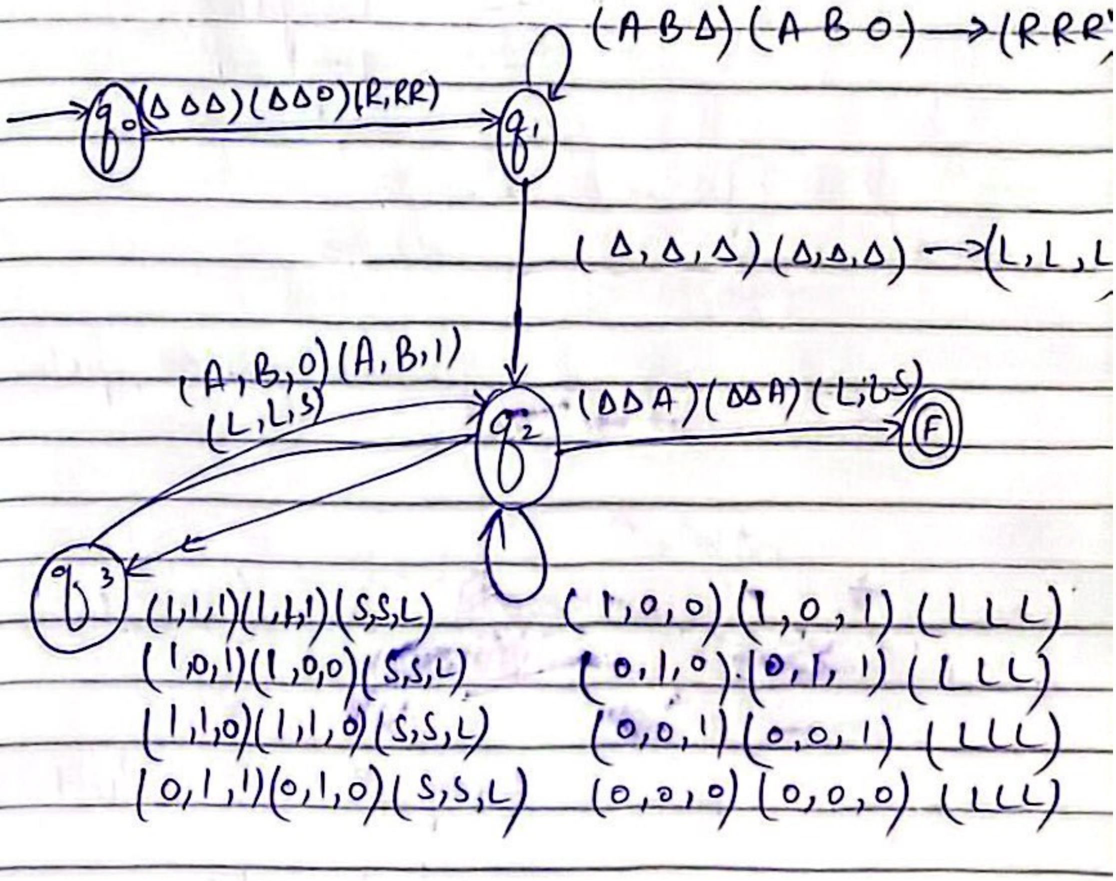


Scanned with CamScanner

binary addition wale sawal.

x	y	z	sum	carry
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

$A = \{0, 1\}$
$B = \{0, 1\}$



Scanned with CamScanner

Pumping Lemma for CFL

- Proof by contradiction
- Assume that A is context free.
- It has a pumping length say P.
- Find a string 's' such that $|s| \geq P$
- Divide the string into 5 parts $vwxyz$
- Prove for some i $vw^ixy^iz \notin A$

xy^iz

$x > 0$ $|wxy| \leq P$

$|wy| > 0$

$v\ w\ x\ y\ z$ → can be epsilon

can be ε

can be epsilon

$(i \geq 0)$

w or y, one can be epsilon at a time.

$a^n b^n c^n$

$a^P b^P c^P$ $P = r+s+t$

$P = r+s+t$

$v = \varepsilon$

$w = a^r$

$x = a^s$

$y = a^t$

$z = b^P c^P$

$|wxy| \leq P$ a^{r+s+t} $|a^P| \leq P$

$\varepsilon (a^r)^i a^s (a^t)^i b^P c^P$

$i = 0$

$a^s b^P c^P \notin A$

$|wxy| \le q/p$

$a^p b^q c^{p \times q}$

$a^i b^j c^{i \times j}$

$i=p \quad i=q$

$a^p b^q c^{p \times q}$

possible q, direct

$v = a^p$

$w^i = b^r$

$x = b^s$

$y^i = b^t$

$z = c^{p \times q}$

$|vy| \ge 0$ ✓

$|wxy| \le p$ or q.

$\cancel{\varepsilon (a^r b^q)^i a^s b^s (a^t b^t)^i}$

$\cancel{\varepsilon (a^p)^i \varepsilon (b^q)^i c^{p \times q}}$

$i=0 \quad c^{p \times q} \notin A$

$a^p (b^r)^i b^s (b^t)^i c^{p \times q}$

$i=0$

$a^p b^s c^{p \times q} \notin A$

$(ab^n)^n (ab^n)^n (ab)^n \qquad (ab)^p (ab)^p (ab)^p$

$v = (ab)^p \qquad p = r+s$

$w^i = ab^r$

$x = \varepsilon$

$y^i = ab^s$

$z = ab^p$

$(ab)^p (ab^r)^i \varepsilon (ab^s)^i (ab)^p$

$i=0 \qquad (ab)^p (ab)^p \notin A.$