

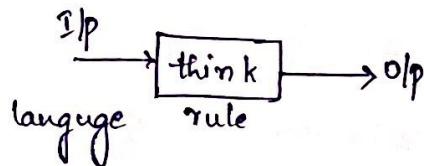
11/6/19

## THEORY OF COMPUTATION

1.0000000000000002

1) Language

2) Grammar



\* Phases of Compiler:-

\* lexical analysis

\* Syntax Analysis

\* Semantic Analysis

\* Intermediate Code generation

\* Code Optimization

\* Code generation

③ Error Reporting, Symbol table & Recovery

\* Lexical Analysis :-

Validity check is done by a language called regular language and the table of first grammar (model) by (finite automata).

→ Syntax Analysis :-

Validity is checked by CFG (Context free grammar)

and grammar for it is push down automata.

It checks by sentence.

→ Semantic Analysis :-

Validity check is done by language called

Context Sensitive language and grammar is Linear bounded automata

Eg:- data type mismatch

12/06/2019

## THEORY OF COMPUTATION

### \* Mathematical Preliminaries & Notations :-

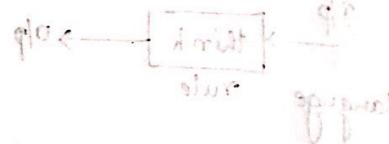
#### 1. Set :-

$$S = \{a, b, c, d\}$$

$$(or) S = \{i, i > 0, i \text{ is even}\}$$

$a \in S \rightarrow \text{belong}$

$a \notin S \rightarrow \text{doesn't belong}$



#### \* Set Operations :-

1. Union

2. Intersection

3. Difference

4. Complementation

5. Size

\* Proper Subset :- The set where  $S_1 \subset S$  which is not equal to actual set.

\* Power Set :-  $S = \{a, b, c\}$  all possible combinations of set are called PS.

$$\Rightarrow P = \{\{a\}, \{b\}, \{c\}, \{ab\}, \{bc\}, \{ca\}, \{abc\}, \emptyset\}$$

#### \* Principle of Inclusion and Exclusion:

$$|A \cup B| = |A| + |B| - |A \cap B|$$

\* Relative Complement - b/w 2 sets

\* Absolute Complement - Universal Set. and Set

#### \* Cartesian Product:

$$S_1 = \{2, 4\}$$

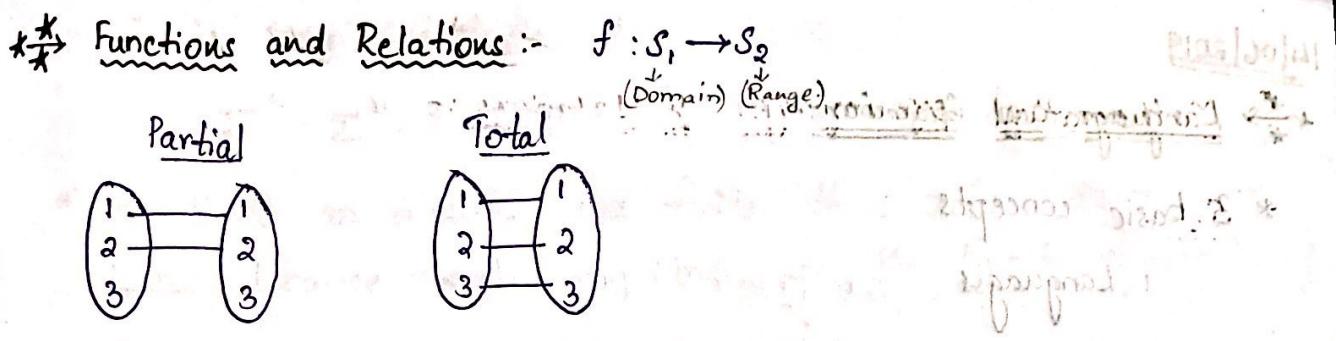
$$S_2 = \{2, 3, 5, 6\}$$

$$S_1 \times S_2 = \{(2, 2), (2, 3), (2, 5), (2, 6), (4, 2), (4, 3), (4, 5)\}$$

#### \* Partition of a Set:

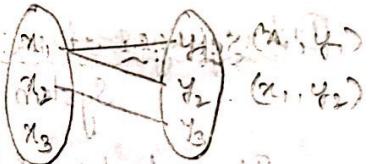
A set is divided into no. of subsets  $\Rightarrow S_1 \cup S_2 \cup \dots \cup S_n = S$

and those subsets should be mutually exclusive and none of the  $S_i$  should be empty.



$\xrightarrow{*}$  Each  $x_i$  can occur at most once as the 1<sup>st</sup> element of the pair.  
If this is not satisfied then we will call it as Relations.

Relations are more general than functions.



$\xrightarrow{*}$  Properties of Relation:-

1. Reflexive

2. Symmetric

3. Anti-Symmetric

4. Transitive

① if  $(a, a) \in R \forall a \in A$

$$A = \{1, 2, 3\}$$

$$R = \{(1, 1), (2, 2), (3, 3)\}$$

② if  $(a, b) \in R$  then  $(b, a) \in R$

③ if  $(a, b) \in R$  then  $(b, a) \notin R$  then  $a = b$

④ if  $(a, b) \in R$  then  $(b, c) \in R \Rightarrow (a, c) \in R$ .

$\xrightarrow{*}$  Two types of Relation:-

1. Partial Order Relation (POSET)

2. Equivalence

① Reflexive, antisymmetric, transitive should be satisfied.

Eg:- The relation  $\leq$  in  $N$   $A = \{2, 3, 4\}$

② Reflexive, symmetric and transitive should be satisfied.

Eg:-  $X = \{1, 2, 3, 4, 5, 6, 7\}$

Relation:  $x - y$  divided by 3.

14/06/2019

## \* Language and Grammar :-

- \* 3 basic concepts

1. Languages

2. Grammar

3. Automata

(1) Languages :- System suitable for expression of our own ideas, facts or concepts.

\* It includes set of simple symbols and rules for their manipulation.

\* Alphabets :- Finite Non-empty set of symbols are called alphabets. It is represented by  $\Sigma$ .

\* String :- Sequence of symbols from alphabet is called strings.

Eg:  $\Sigma = \{a, b\}$  then abab, aaabb are strings on  $\Sigma$ .

\* Elements inside  $\Sigma$  are always lower case letters and name of string is always capital letter.

\* Concatenation of 2 strings :  $w = a_1 a_2 \dots a_n$  &  $v = b_1 b_2 \dots b_m$

$$wv = a_1 a_2 \dots a_n b_1 b_2 \dots b_m$$

$w^R = a_n \dots a_2 a_1$  \* length of string = No. of symbols in string

\*  $\epsilon / \lambda$  = empty string  $|\lambda| = 0$ .

i.e.  $\lambda w = w \lambda = \emptyset w$ .

\* Prefix and Suffix of String :-  $w = \{abbab\}$ ; then prefix =  $\{\lambda, a, ab, abb, abba, abbab\}$ .

Suffix =  $\{b, ba, bab, \dots\}$

\* If  $w$  is a string  $w^n$  stands for that string obtained by repeating  $w$   $n$  times.

\*  $w^0 = \lambda \rightarrow$  empty string.

\* If  $\Sigma$  is an alphabet then  $\Sigma^*$  denotes the set of strings obtained by concatenating zero or more symbols from  $\Sigma$  it includes empty string.

\*  $\Sigma^+$  it doesn't include  $\lambda$ .

$$\Sigma^+ = \Sigma^* - \{\lambda\}$$

\* If  $\Sigma$  is an alphabet and finite then  $\Sigma^*$  and  $\Sigma^+$  are infinite because there is no limit on the length of the string.

⊗ Language is defined generally on a subset of  $\Sigma^*$  or set of strings on an alphabet  $\Sigma$ . A string in  $L$ , be called a sentence of  $L$ .

Eg:- If  $\Sigma = \{a, b\}$  then  $\Sigma^* = \{\lambda, ab, aa, bb, aabb, aaa, \dots\}$

The set  $L = \{a, aa, aaa, aab\}$  is a language on  $\Sigma$ .

(or)  $L = \{a^n b^n : n \geq 0\} \Rightarrow L = \{\lambda, ab, aabb, aaabbb, \dots\} \rightarrow$  infinite.

\* Complement of a language:-  $L = \Sigma^* - L$

\* Reverse of a language:-  $L^R = \{w^R : w \in L\}$

\* Concatenation of two languages:-

Suppose  $L_1, L_2$  are 2 languages. If we concatenate both then  $L_1 L_2 = \{xy : x \in L_1, y \in L_2\}$ .

\*  $L^n \rightarrow L$  concatenated with itself  $n$  times.

\*  $L^0 \rightarrow \{\lambda\}$

\* Properties of language:-

1. Star closure  $= L^* = L^0 U L^1 U L^2 \dots$

2. Additive closure  $= L^+ = L^1 U L^2 U L^3 \dots$

Eg:- If  $L = \{a^n b^n : n \geq 0\}$  then  $L^2 = \{a^n b^n a^m b^m\}$

Let  $L = \{\lambda, a, b\}$ .

$L^* = \{aabb\}$ ,  $L = \{ab\}$

$L^2 = \{abaabb, aabbab\}$

$\therefore L^2 = \{a^n b^n a^m b^m\}$  such that  $n \geq 0, m \geq 0$ . Not  $m$  &  $n$  are unrelated ( $\neq$ )

## ② Grammar:-

Grammar For any language tell us that whether it is formed correctly or not.

Notation:  $\langle \text{Sentence} \rangle \rightarrow \langle \text{noun-phrase} \rangle \langle \text{predicate} \rangle$

$\langle \text{noun-phrase} \rangle \rightarrow \langle \text{article} \rangle \langle \text{noun} \rangle$

$\langle \text{predicate} \rangle \rightarrow \langle \text{Verb} \rangle$

18/06/2019

\* Grammar is defined as a quadruple  $G = (V, T, S, P)$ .

$V$  = Set of finite & Variable (Non-Terminals)

$T$  = Finite set of Terminal Symbols.

$S$  = Starting Variable

$P$  = Finite set of productions.

\* All terminal should {be lower case} and Non-Terminals can be capital letters.

Eg:- Consider the grammar  $G = (\{S\}, \{a, b\}, \{S\}, P)$ . Suppose production given as  $S \rightarrow aSb$ ,  $S \rightarrow \lambda$

Sol: ①  $S \xrightarrow{\textcircled{1}} aSb$

$\xrightarrow{\textcircled{2}} a\lambda b$

$a\lambda$

$aa\lambda b$

$aaa\lambda bb$

$a^n b^n$

$\Rightarrow (ab)^n$

① ②

2 Method :-  $S \rightarrow aB | \lambda$

$B \rightarrow Sb$

$G = (\{S, B\}, \{a, b\}, S, P)$

$S \xrightarrow{\textcircled{1}} aB$

$\xrightarrow{\textcircled{2}} aSb$

$\xrightarrow{\textcircled{3}} a\lambda B b$

$\xrightarrow{\textcircled{4}} a\lambda S bb$

$\xrightarrow{\textcircled{5}} aa\lambda bb$

- \* If  $w_1 \Rightarrow w_2 \Rightarrow w_3 \Rightarrow \dots \Rightarrow w_n$ . Then we say that  $w_1 \xrightarrow{*} w_n$  (derives)
- and we can generally denote as  $w_1 \xrightarrow{*} w_n$ . By applying m number of prod.
- \* (\*) indicates that the no. of steps can be taken to derive  $w_n$  from  $w$
- By applying production rule in diff order a given grammar can normally generate many strings.
- Let  $G = (V, T, S, P)$ , then the set  $L(G) = \{w \in T^* : S \xrightarrow{*} w\}$  is the language generated by  $G$ .

### Problems:-

1. Find the Grammar that generates  $L = \{a^n b^{n+1} : n \geq 0\}$ . Find set of all productions.

$$\text{Sol: } G = (\{S, a, b\}, \{a, b\}, S, P)$$

b, abb, aabbb,

$S \rightarrow aSbb$  } production.

$S \rightarrow \epsilon$

$$\begin{aligned} ① &\Rightarrow Sb = b \\ ② &\Rightarrow Sb = aSbb = abb \end{aligned}$$

2. Find grammar for  $\Sigma = \{a, b\}$  that generates,

(a) All strings with exactly 1  $a \rightarrow S \rightarrow Sb | bs | a$ .

(b) All strings with atleast 1  $a$ .

(c) All strings with no more than 3  $a \rightarrow S \rightarrow Sb | bs | a$

3. Let  $\Sigma = \{a, b\}$  find the grammars that generate the language

$$L = \{a^n b^m : n \geq 0, m > n\}$$

Ans:-

a b, abb---, ab,

$$S = Sb | \lambda$$

$$S = aSb$$

$$S = b$$

$$\begin{array}{l} ① \\ \Rightarrow Sb = b \checkmark \end{array}$$

$$\begin{array}{l} ② \\ \Rightarrow asb \xrightarrow{②} asbb \xrightarrow{②} abb \checkmark \end{array}$$

$$\begin{array}{l} ③ \\ \Rightarrow asb \xrightarrow{③} asbb \xrightarrow{③} abb \dots \end{array}$$

19/06/19

Q. What language does the grammar with these productions generate.

$$① S \rightarrow Aa$$

$$② A \rightarrow B$$

$$③ B \rightarrow Aa$$

Ba

Aaa

Baa

$L = \emptyset$

Q. Find grammar for (i)  $L = \{a^n b^n; n \geq 0\}$

$$(ii) L = \{a^{n+2} b^n; n \geq 1\}$$

$$(iii) L = \{a^n b^{n-3}; n \geq 3\}$$

Q. Construct a grammar where  $\Sigma = \{0, 1, c\}$  which accepts all strings of palindrome. use c as delimiter.

Q. Find the grammar where  $\Sigma = \{\alpha\}$  given language

$$(i) L = \{w : |w| \bmod 3 = 0\}$$

$$(ii) L = \{w : |w| \bmod 3 > 0\}$$

Q. What is the language generated by grammar  $S \rightarrow aSa \mid bSb \mid ab$

Q. Consider the grammar  $G = (\{A, S\}, \{a, b\}, S, P)$  with production

$$S \rightarrow aAb \mid \lambda$$

$$A \rightarrow aAb \mid \lambda$$

Answers:-

4A.  $S \rightarrow Aa$

$A \rightarrow B$

$B \rightarrow Aa$

This string never ends so,  $L = \emptyset$

5A. (i)  $L = \{a^n b^n; n \geq 0\}$

$\text{OK}^* = \emptyset, ab^*, aabb^*, aabb^*$

$$S \rightarrow aSb \mid \lambda$$

$$S \rightarrow \underline{Sb}$$

$$S \rightarrow \underline{ab}$$

asb

aasbb

aasbb

aasbbb

aasbbb

aasbbb

aabb

&lt;

(i)  $s \rightarrow asbb | \lambda$

(ii)

21/06/19

### ③ $\xrightarrow{*}$ Automata :-

It's an abstract model of a digital computer which includes input file, control unit, output and storage.

\* At any given time the control unit is in some internal state and ifp mechanism scanning a particular symbol on the ifp file. The internal state of control unit at the next time step is determined by transition function.

The transition fun produce the next state in terms of current state, current ifp symbol and info currently in temp storage.

#### 2 Types of Automata.

1. Deterministic :- It is one in which each move is uniquely determined by current configuration.

2. Non-Deterministic : It may have several possible view so, we can only predict set of possible actions.

### $\xrightarrow{*}$ Models:-

Automata can be designed in 2 ways.

1. acceptor

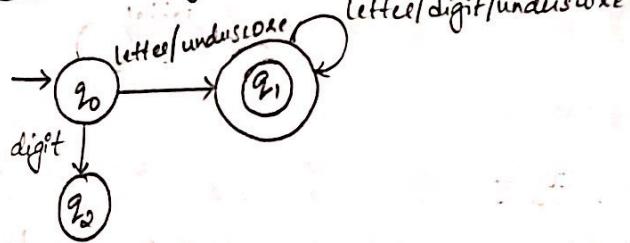
2. transducer

1. Acceptor: An automata whose o/p response is limited to a simple yes or No, is called as acceptor i.e. acceptor either accepts the given string or rejects it.

2. Transducer: It is a more general automaton capable of producing strings of o/p symbols as o/p.

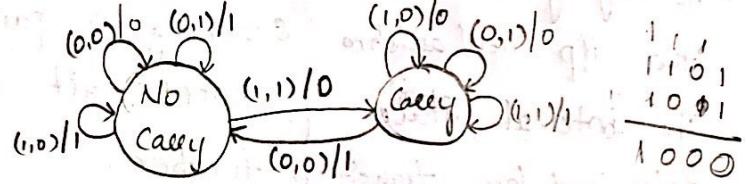
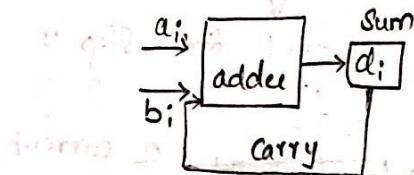
Eg:- 1. Acceptor:  $\langle id \rangle \rightarrow \langle letter \rangle \langle rest \rangle \mid \langle underscore \rangle \langle rest \rangle$   
 $\langle rest \rangle \rightarrow \langle letter \rangle \langle rest \rangle \mid \langle digit \rangle \langle rest \rangle \mid \langle underscore \rangle \langle rest \rangle$   
 $\langle letter \rangle \rightarrow a \mid b \mid \dots \mid z \mid A \mid B \mid \dots \mid Z$   
 $\langle digit \rangle \rightarrow 0 \mid 1 \mid \dots \mid 9 \mid \langle underscore \rangle \rightarrow _$

## \* Automata diagram

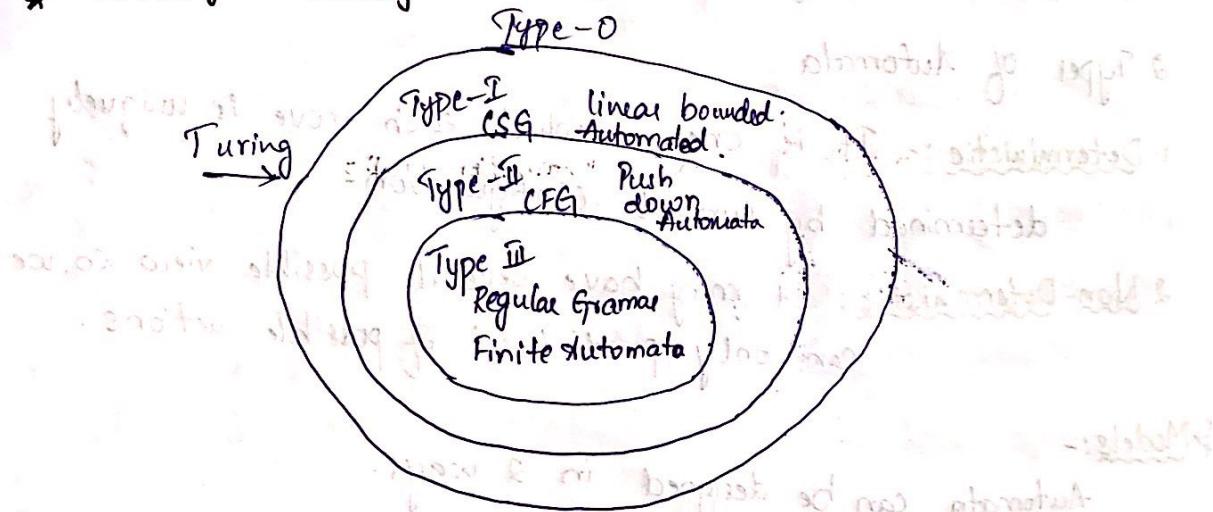


2. Transducer :- binary adder

	0	1	2	3	4	5	6	7	8	9
0	0	1	2	3	4	5	6	7	8	9
1	1	0	1	2	3	4	5	6	7	8
2	2	1	0	1	2	3	4	5	6	7
3	3	2	1	0	1	2	3	4	5	6
4	4	3	2	1	0	1	2	3	4	5
5	5	4	3	2	1	0	1	2	3	4
6	6	5	4	3	2	1	0	1	2	3
7	7	6	5	4	3	2	1	0	1	2
8	8	7	6	5	4	3	2	1	0	1
9	9	8	7	6	5	4	3	2	1	0



$\xrightarrow{*}$  Chomsky hierarchy :-



## \*<sup>★</sup> Regular Grammar (Type - III) :-

$S \rightarrow Aa$

(or)

$S \rightarrow \cap A$

$$S \rightarrow \epsilon | \lambda | a$$

\* '8' should not appear on the ~~written~~<sup>right</sup> side of the rule.

\* Finite Automata :- (or) Finite Acceptors.

It doesn't include any temp storage. Only Acceptors is limited since temp storage is not available, it can process only limited input.

\* Two types of FA:-

1. Deterministic FA (DFA)

2. Non-Deterministic FA (NFA)

1. Deterministic Finite Automata:-

DFA is defined by quintuple (5) where

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

$Q$  = Finite Set of internal States.

$\Sigma$  = Finite set of symbols called Input Alphabet.

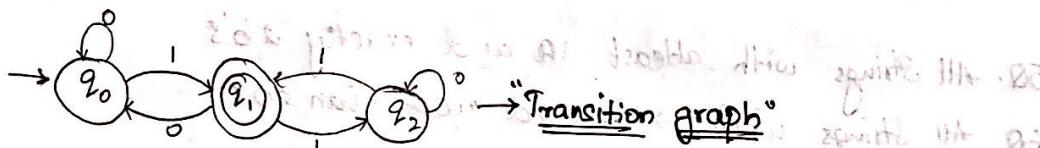
$\delta$  = Transition function of type total function.

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

$q_0$  = Initial State.  $q_0 \in Q$ .

$F$  = Set of Final States.  $F \subseteq Q$ .

Eg:-



$$Q = q_0, q_1, q_2$$

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

$$\delta F = q_1$$

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

$$\delta(q_0, 1) = q_1$$

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

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

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

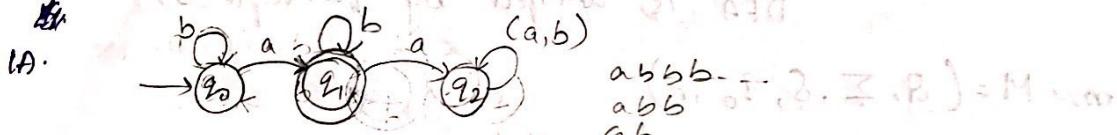
$$\delta(q_2, 1) = q_1$$

0110, 100, 110, 111, 1101

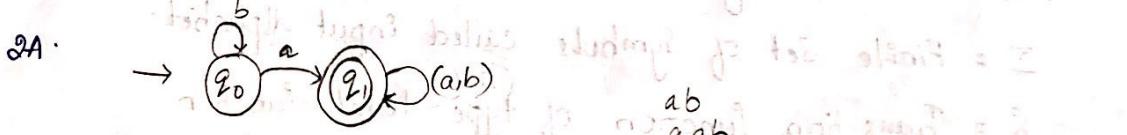
\* This automata doesn't accept the string ~~starts~~ ends with '0'.

\* $\rightarrow$  Problems:-

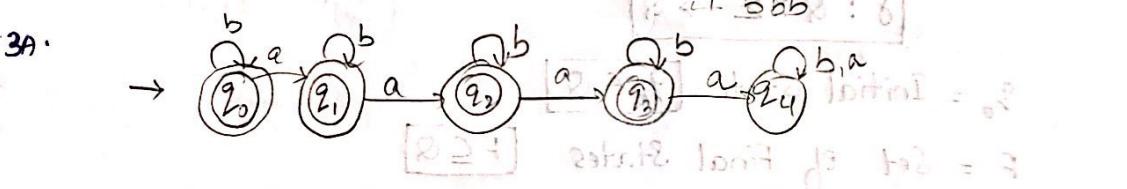
1. Construct DFA which accepts all strings with exactly 1a
2. All strings with atleast 1a.
3. All strings with no more than 3a's
4. All strings starting with 01



$$abbb - \{a, b\}^4 = 16 - 1 = 15$$



$$abb - \{a, b\}^3 = 8 - 1 = 7$$



$$abb - \{a, b\}^3 = 8 - 1 = 7$$

5Q. All strings with atleast 1a and exactly 2b's

6Q. All strings with 2a's and more than 2b's

20619

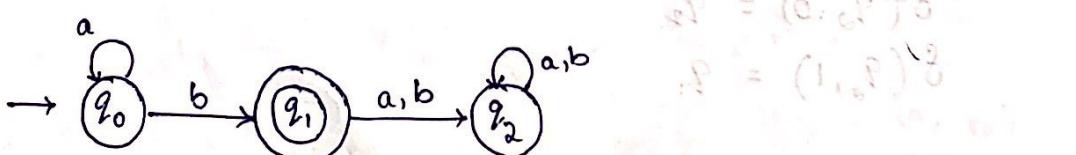
\* $\rightarrow$  ① Extended Transition Function :-

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

$$\delta^*(q_0, ab) = q_2 \quad \delta^*(q_0, ab) = \delta(\delta^*(q_0, a))(b)$$

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

\* $\rightarrow$  ② Transition table :-



$Q_i$	a	b
$q_0$	$q_0$	$q_1$
$q_1$	$q_2$	$q_2$
$q_2$	$q_2$	$q_2$

\*<sup>3</sup> Languages and DFA's:-

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

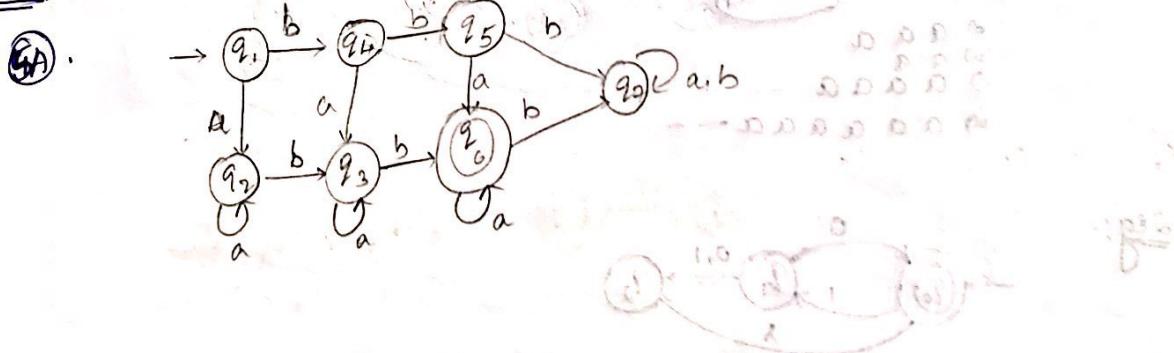
$$L(M) = \{ w \in \Sigma^* : \delta^*(q_0, w) \in F \}$$

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

\* Questions

- 7Q. Construct DFA that accepts all strings which accept 2a's & more than one b's.
- 8Q. All strings start containing 2 consecutive zero followed by 2 consecutive one's.
- (9Q.) All strings containing even no. of 0's and any num of 1's.
- (10Q.) All strings containing 00 but not triple zero.
- (11Q.) A strings with any no. of a's & b's.
- \* 12Q. All strings in which the left most symbol differs from right most one.  $\Sigma = \{0, 1\}$ .
- \* 13Q. Left most 2 symbols and rightmost 2 symbols are identical
- (14Q.)  $L = \{ w : |w| \bmod 3 = 0 \}$

Ans:-



## \* Non-Deterministic Finite Automata (NFA) :-

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

$\delta$  = Transition function of type total fun

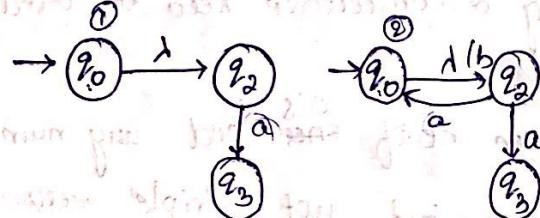
$$\delta : Q \times (\Sigma \cup \{\lambda\}) \rightarrow 2^Q$$

$$(1) \quad \delta(q_1, a) = (q_0, q_2)$$

$$(2) \quad \delta(q_1, \lambda) = q_3$$

(3)  $\delta(q_1, a)$  may be empty.

Eg:-



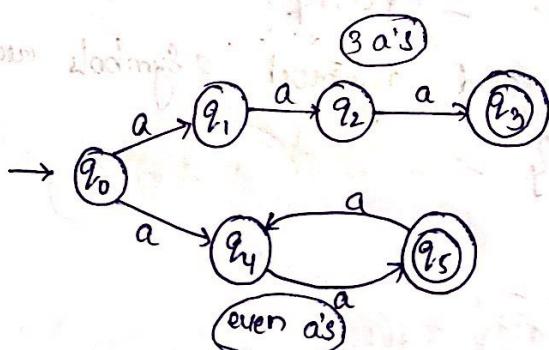
→ Extended transition:

$$(q_0, a) = \{q_3\}$$

$$(q_0, a) = \{q_0, q_2, q_3\}$$

(q0, a) = {Null} for b.

Ex:-



⇒ a a a.

⇒ a a

⇒ a a a a - - -

⇒ a a a a a a - - -

Eg:-



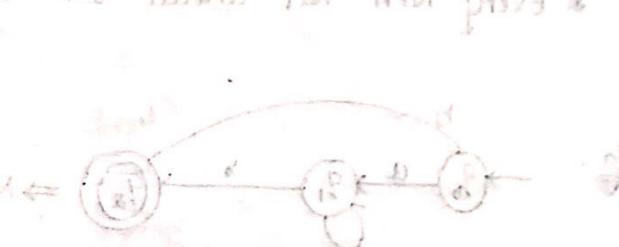
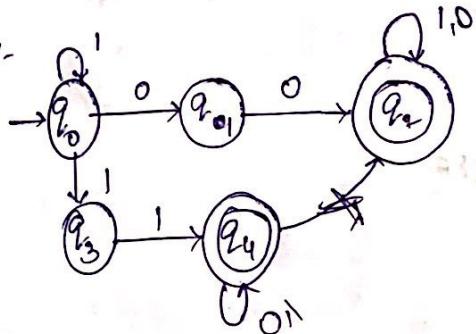
101010 - - -

$$L = \{(10)^n ; n \geq 0\}.$$

## \*Problems

- 1Q. Design an NFA that accepts a binary string which have atleast 1 pair 00 , (Or) 1 pair 11.

Ans :-



$$\text{Q1} \leftarrow \{0,1\}^* = (\{0,1\})^{\omega}$$

- 2Q. Design NFA , that ends with 00 (Or) 11.

- 3Q. construct NFA with 4 states where  $L = \{a^n : n \geq 0\} \cup \{ba^n : n \geq 0\}$  (Or)

$$\text{Q3} \leftarrow \{a\}^* = (\{a\})^{\omega}$$

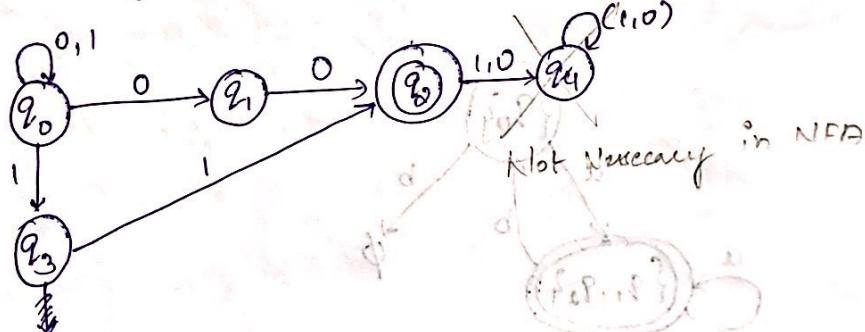
$$\text{Q3} \leftarrow \{b\}^* = (\{b\})^{\omega}$$

$$\text{Q3} \leftarrow \{a,b\}^* = (\{a,b\})^{\omega}$$

- 4Q.  $L = \{abaan : n \geq 0\} \cup \{aba^n : n \geq 0\}$

- 5Q.  $L = \{ab, abc\}^*$

2Ans:-



$$\text{Q2} \leftarrow \{a\}^* = (\{a\})^{\omega}$$

$$\text{Q2} \leftarrow \{b\}^* = (\{b\})^{\omega}$$

$$\text{Q2} \leftarrow \{a,b\}^* = (\{a,b\})^{\omega}$$

3Ans:-

- $\{a^n / b^n a : n \geq 0\} \cup \{a^n b / b^n a : n \geq 0\} \cup \{a^n b^n : n \geq 0\}$

- $a^n / b^n a : n \geq 0$

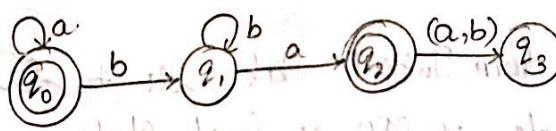
- $a^n b / b^n a : n \geq 0$

- $a^n b^n : n \geq 0$

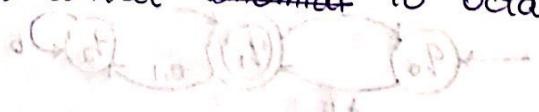
- $a^n / b^n a : n \geq 0$

- $a^n b / b^n a : n \geq 0$

- $a^n b^n : n \geq 0$



- Q. Define a transducer to convert binary to octal.



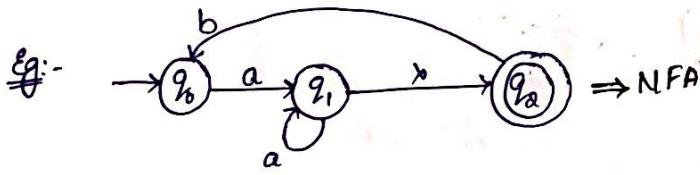
$$\{0,1\}^* = (\{0,1\})^{\omega}$$

$$\{0,1\}^* = (\{0,1\})^{\omega}$$

26/06/2019

\*  $\xrightarrow{*}$  Conversion of NFA to DFA :-

\* Every NFA has atleast one DFA.



$$\delta_N(q_0, a) = \{q_1, q_2\} \rightarrow ①$$

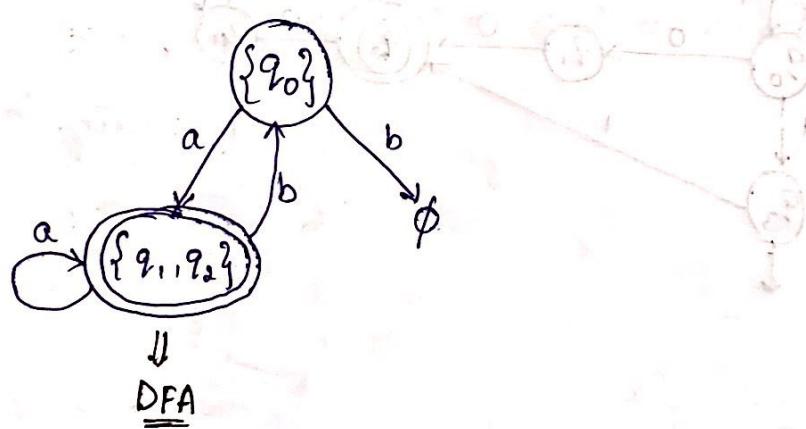
$$\delta_N(q_0, b) = \{\emptyset\} \rightarrow ②$$

$$\delta_N(q_1, a) = \{q_2\} \rightarrow ③$$

$$\delta_N(q_1, b) = \{q_0\} \rightarrow ④$$

$$\delta_N(q_2, a) = \{\emptyset\} \rightarrow ⑤$$

$$\delta_N(q_2, b) = \{q_0\} \rightarrow ⑥$$

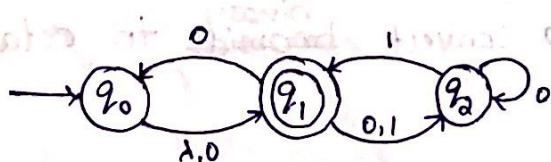


$$* \delta_N(\{q_1, q_2\}, a) = \delta_N(q_1, a) \cup \delta_N(q_2, a) = \{q_2\} \cup \emptyset = \{q_2\}$$

$$* \delta_N(\{q_1, q_2\}, b) = \delta_N(q_1, b) \cup \delta_N(q_2, b) = \{q_0\}.$$

Note:- Any  $\lambda$  transition from Initial state leads to final state then mark initial state of DFA as final state.

Q)



$$\delta_N(q_0, 0) = \{q_1, q_2, q_3\}$$

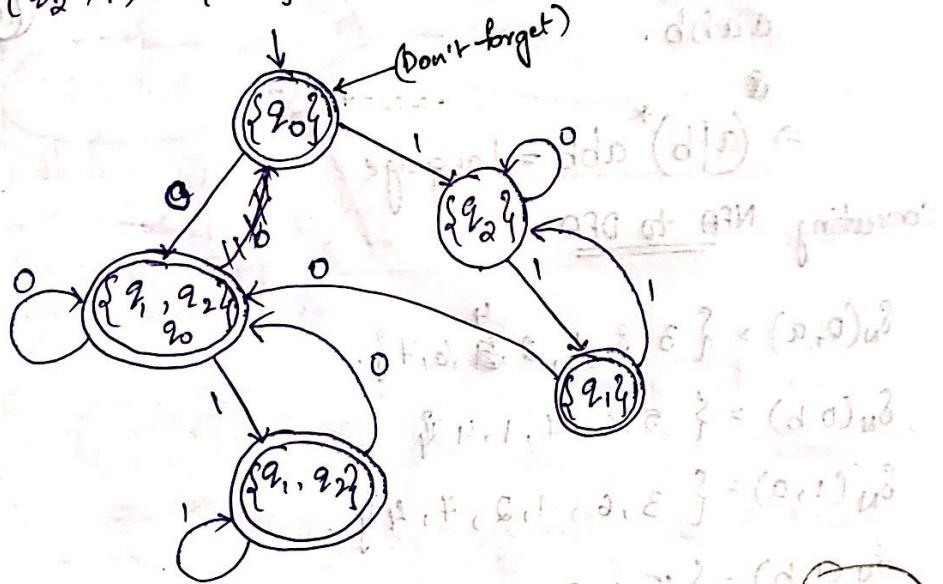
$$\delta_N(q_0, 1) = \{q_2, q_3\}$$

$$\delta_N(q_1, 0) = \{ q_{0_N}^{q_1}, q_2 \}$$

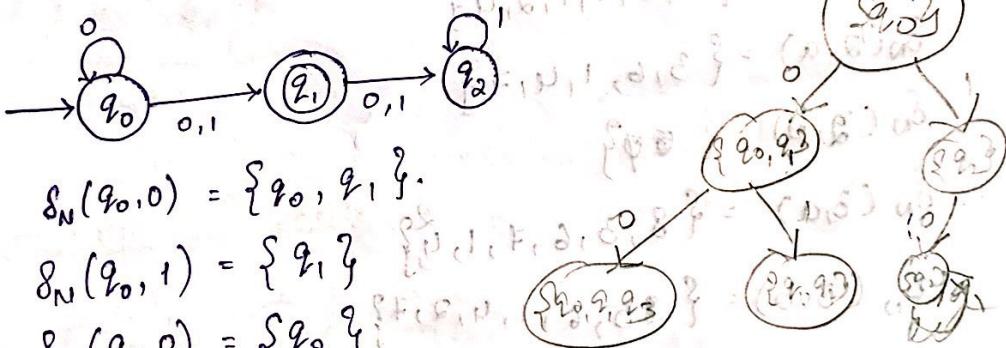
$$\delta_N(q_1, 1) = \{q_2\} q_{12}$$

$$\delta_N(q_2, 0) = \{q_2\}$$

$$\delta_N(q_2, 1) = \{q_1\}$$



38



$$g_N(q_0, 0) = \{q_0, q_1\}.$$

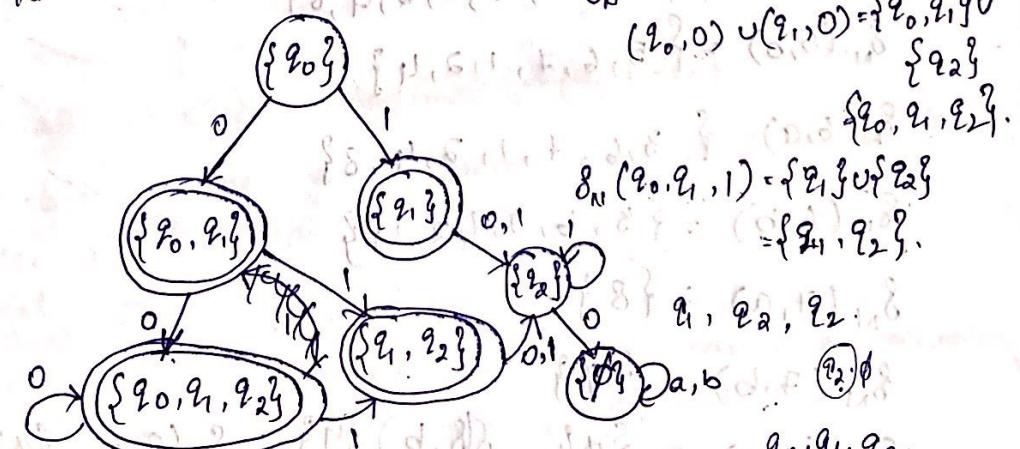
$$g_{11}(q_{n+1}) = \{q_1\}$$

$$\delta_{\alpha}(q_1, 0) = \{q_2\}$$

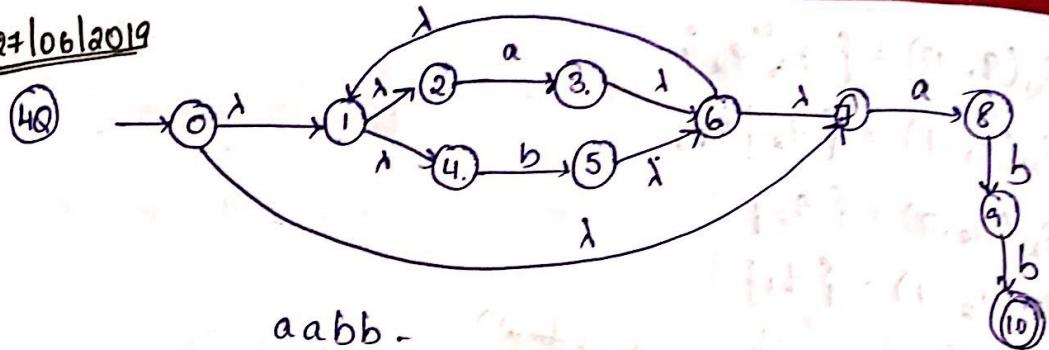
$$\delta_1(q_1, 0) = \{\emptyset\}$$

$$g_1(q_1, 1) = \{q_2\}$$

$$g(q_1, 1) = \{q_2\}.$$



27/06/2019



aabb -

$$\Rightarrow (a|b)^* abb = \text{Language}$$

Converting NFA to DFA :-

$$\delta_N(0, a) = \{ 3, 8, 1, 2, 4, 6, 7 \}$$

$$\delta_N(0, b) = \{ 5, 6, 7, 1, 4, 2 \}$$

$$\delta_N(1, a) = \{ 3, 6, 1, 2, 4, 7 \}$$

$$\delta_N(1, b) = \{ 5, 6, 7, 1, 2, 4 \}$$

$$\delta_N(2, a) = \{ 3, 6, 1, 4, 7 \}$$

$$\delta_N(2, b) = \{ \emptyset \}$$

$$\delta_N(3, a) = \{ 8, 3, 6, 7, 1, 4 \}$$

$$\delta_N(3, b) = \{ 5, 6, 1, 4, 2, 7 \}$$

$$\delta_N(4, a) = \{ \emptyset \}$$

$$\delta_N(4, b) = \{ 5, 6, 7, 1, 2, 4 \}$$

$$\delta_N(5, a) = \{ 3, 6, 7, 1, 2, 4, 8 \}$$

$$\delta_N(5, b) = \{ 5, 6, 7, 1, 2, 4 \}$$

$$\delta_N(6, a) = \{ 3, 6, 7, 1, 2, 4, 8 \}$$

$$\delta_N(6, b) = \{ 5, 6, 7, 1, 2, 4 \}$$

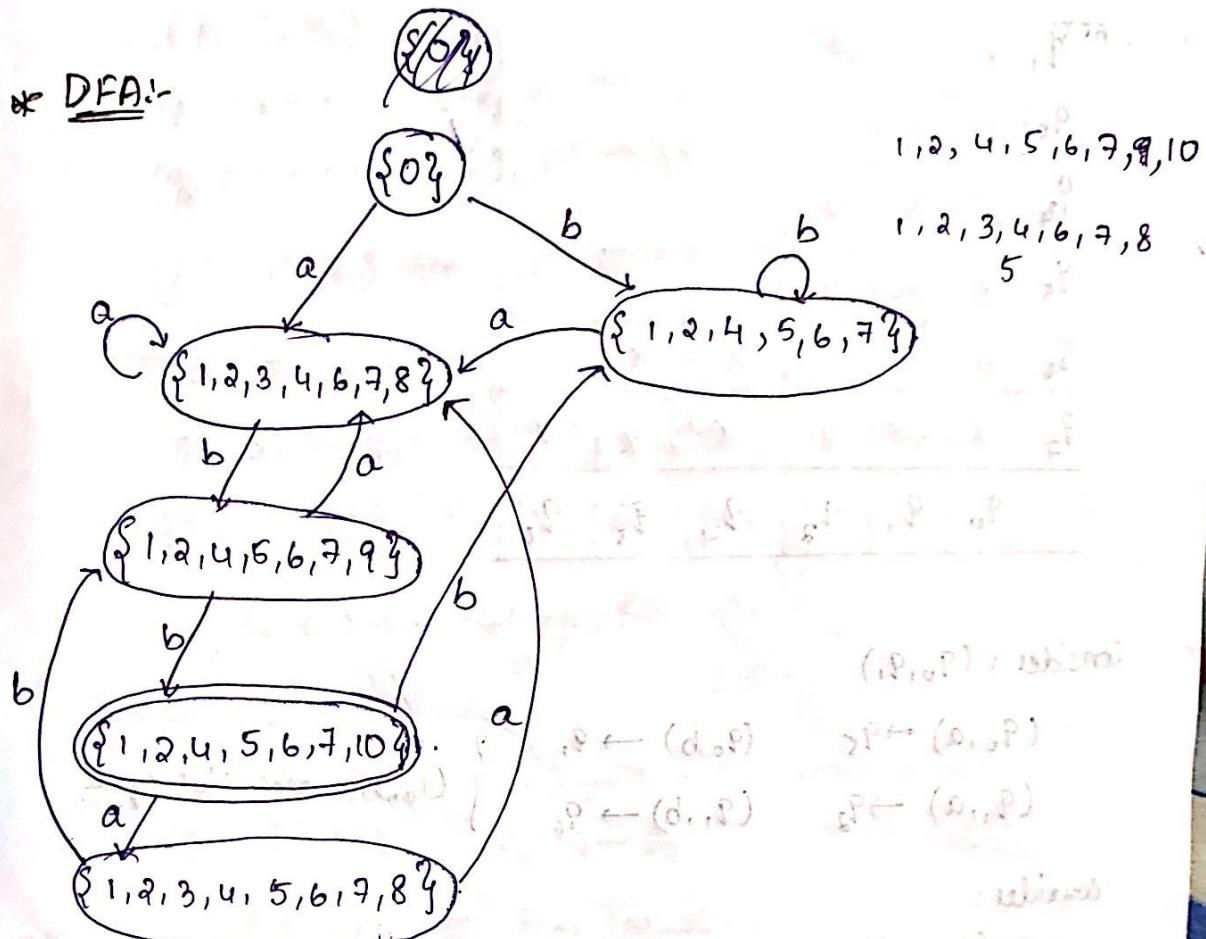
$$\delta_N(7, a) = \{ 8 \}$$

$$\delta_N(7, b) = \{ \emptyset \}$$

$$\delta_N(8, a) = \{ \emptyset \} \quad \delta_N(8, b) = \{ 9 \}; \quad \delta_N(9, a) = \{ \emptyset \}$$

$$\delta_N(9, b) = \{ 10 \}$$

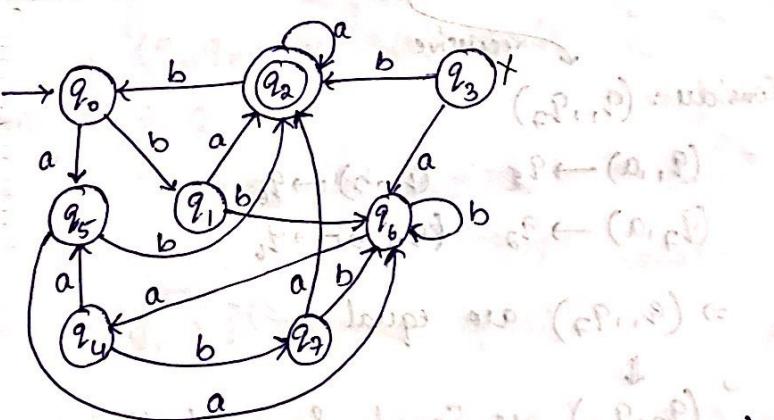
\* DFA:-



28/06/19

\* \* Minimization of DFA:-

Eg:-



1. Remove all unreachable states/node.  $\Rightarrow (q_3 \text{ Node})$ .

2. Draw table x-axis, y-axis where x-axis = all but not last step  
y-axis = all but not first.

3. All final state  $(a, w) \in F$  with non-final put (X).

4.  $(a, w) \in F$        $(a, w) \notin F$       if equivalent do recursive  
Indistinguishable      Distinguishable      or put X

equivalent

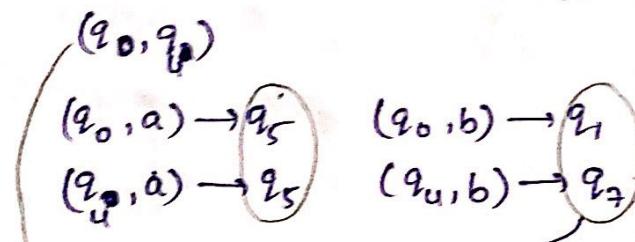
Not equivalent

$q_1$	$\times$					
$q_3$	$\times$	$\times$				
$q_4$	$\checkmark$	$\times$	$\times$			
$q_5$	$\times$	$\times$	$\times$	$\times$		
$q_6$	$\times$	$\times$	$\times$	$\times$	$\times$	
$q_7$	$\times$	$\checkmark$	$\times$	$\times$	$\times$	$\times$
$q_0$	$q_1$	$q_2$	$q_4$	$q_5$	$q_6$	

Consider :  $(q_0, q_1)$

$$\begin{array}{l} (q_0, a) \rightarrow q_5 \\ (q_0, b) \rightarrow q_1 \end{array} \quad \left. \begin{array}{l} (q_1, a) \rightarrow q_3 \\ (q_1, b) \rightarrow q_6 \end{array} \right\} (q_0, q_1) \text{ are Not equal.}$$

Consider :



Consider :  $(q_1, q_7)$

$$\begin{array}{l} (q_1, a) \rightarrow q_5 \\ (q_1, b) \rightarrow q_6 \end{array} \quad \begin{array}{l} (q_7, a) \rightarrow q_2 \\ (q_7, b) \rightarrow q_6 \end{array}$$

$\Rightarrow (q_1, q_7)$  are equal. ( $\checkmark$ )

$\downarrow$   
 $(q_0, q_4)$  are Equal so, put ( $\checkmark$ )

Consider :  $(q_0, q_5)$

$$\begin{array}{l} (q_0, a) \rightarrow q_5 \\ (q_0, b) \rightarrow q_1 \end{array} \quad \begin{array}{l} (q_5, a) \rightarrow q_6 \\ (q_5, b) \rightarrow q_2 \end{array}$$

$\therefore (q_0, q_5)$  are Not equal.

Consider:  $(q_0, q_6)$

$$\begin{array}{ll} (q_0, a) \rightarrow q_5 & (q_0, b) \rightarrow q_1 \\ (q_6, a) \rightarrow q_4 & (q_6, b) \rightarrow q_6 \end{array}$$

$\therefore (q_0, q_6)$  are Not equal.

Consider:  $(q_0, q_7)$

$$\begin{array}{ll} (q_0, a) \rightarrow q_5 & (q_0, b) \rightarrow q_1 \\ (q_7, a) \rightarrow q_2 & (q_7, b) \rightarrow q_6 \end{array}$$

$\therefore (q_0, q_7)$  are Not equal.

Consider:  $(q_1, q_4)$

$$\begin{array}{ll} (q_1, a) \rightarrow q_2 & (q_1, b) \rightarrow q_6 \\ (q_4, a) \rightarrow q_5 & (q_4, b) \rightarrow q_9 \end{array}$$

$\therefore (q_1, q_4)$  are Not equal.

Consider:  $(q_1, q_5)$

$$\begin{array}{ll} (q_1, a) \rightarrow q_2 & (q_1, b) \rightarrow q_6 \\ (q_5, a) \rightarrow q_6 & (q_5, b) \rightarrow q_2 \end{array}$$

$\therefore (q_1, q_5)$  are Not equal.

Consider:  $(q_1, q_6)$

$$\begin{array}{ll} (q_1, a) \rightarrow q_2 & (q_1, b) \rightarrow q_6 \\ (q_6, a) \rightarrow q_4 & (q_6, b) \rightarrow q_6 \end{array}$$

$\therefore (q_1, q_6)$  are Not equal.

Consider:  $(q_2, q_4)$

$$\begin{array}{ll} (q_2, a) \rightarrow q_3 & (q_2, b) \rightarrow q_0 \\ (q_4, a) \rightarrow q_5 & (q_4, b) \rightarrow q_7 \end{array}$$

$\therefore (q_2, q_4)$  are Not equal.

Consider:  $(q_1, q_7)$

$$\begin{array}{ll} (q_1, a) \rightarrow q_2 & (q_1, b) \rightarrow q_6 \\ (q_7, a) \rightarrow q_2 & (q_7, b) \rightarrow q_6 \end{array}$$

$\therefore (q_1, q_7)$  is Equal.

$(q_1, q_7)$  is Equal

consider:  $(q_4, q_5)$

$$\begin{array}{ll} (q_4, a) \rightarrow q_5 & (q_4, b) \rightarrow q_7 \\ (q_5, a) \rightarrow q_6 & (q_5, b) \rightarrow q_8 \end{array}$$

$\therefore (q_4, q_5)$  are Not equal.

consider:  $(q_4, q_6)$

$$\begin{array}{ll} (q_4, a) \rightarrow q_5 & (q_4, b) \rightarrow q_7 \\ (q_6, a) \rightarrow q_4 & (q_6, b) \rightarrow q_6 \end{array}$$

$\therefore (q_4, q_6)$  are Not equal.

consider:  $(q_4, q_7)$

$$\begin{array}{ll} (q_4, a) \rightarrow q_5 & (q_4, b) \rightarrow q_7 \\ (q_7, a) \rightarrow q_2 & (q_7, b) \rightarrow q_6 \end{array}$$

$\therefore (q_4, q_7)$  are not equal.

consider:  $(q_5, q_6)$

$$\begin{array}{ll} (q_5, a) \rightarrow q_6 & (q_5, b) \rightarrow q_2 \\ (q_6, a) \rightarrow q_4 & (q_6, b) \rightarrow q_6 \end{array}$$

$\therefore (q_5, q_6)$  are not equal.

consider:  $(q_5, q_7)$

$$\begin{array}{ll} (q_5, a) \rightarrow q_6 & (q_5, b) \rightarrow q_2 \\ (q_7, a) \rightarrow q_3 & (q_7, b) \rightarrow q_6 \end{array}$$

$\therefore (q_5, q_7)$  are not equal.

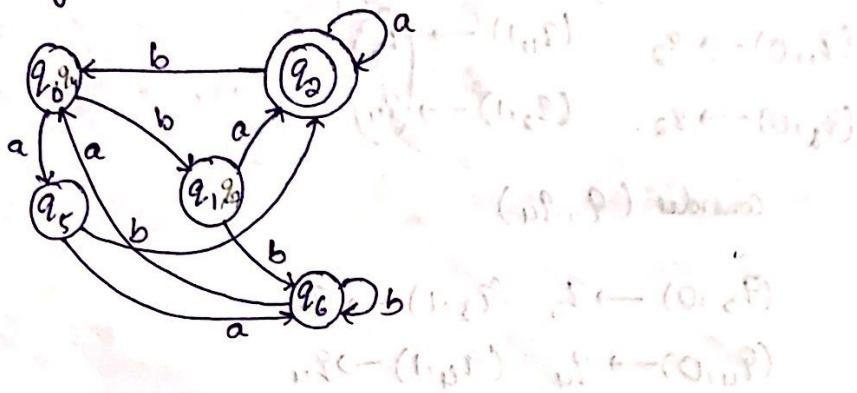
consider:  $(q_6, q_7)$

$$\begin{array}{ll} (q_6, a) \rightarrow q_4 & (q_6, b) \rightarrow q_6 \\ (q_7, a) \rightarrow q_2 & (q_7, b) \rightarrow q_6 \end{array}$$

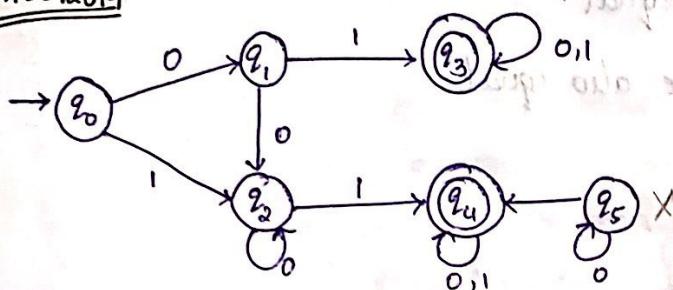
$$(q_2, q_4) \rightarrow X$$

$\therefore (q_6, q_7)$  are Not equal.

Redrawing :-



29/06/2019



$q_1$	$x$			
$q_2$	$x$	$\checkmark$		
$q_3$	$x$	$x$	$x$	
$q_4$	$x$	$x$	$x$	$\checkmark$
	$q_0$	$q_1$	$q_2$	$q_3$

Note:- If both states are final we shouldn't mark as ( $x$ ), we should workout and find.

Consider :-  $(q_0, q_1)$

$$(q_0, 0) \rightarrow q_1$$

$$(q_0, 1) \rightarrow q_3$$

$(q_1, q_2)$   $\therefore q_0, (q_0, q_1)$  are Not equal.

$$(q_1, 0) \rightarrow /$$

$$(q_1, 1) \rightarrow /$$

$$(q_0, 0) \rightarrow /$$

$$(q_0, 1) \rightarrow /$$

Consider :-  $(q_0, q_2)$

$$(q_0, 0) \rightarrow q_1$$

$$(q_2, 0) \rightarrow q_2$$

$$(q_0, 1) \rightarrow q_2$$

$$(q_0, 1) \rightarrow q_4$$

$(q_0, q_2)$  are Not equal.

Consider :  $(q_1, q_2)$

$$\begin{array}{ll} (q_1, 0) \rightarrow q_2 & (q_1, 1) \rightarrow q_3 \\ (q_2, 0) \rightarrow q_2 & (q_2, 1) \rightarrow q_4 \end{array}$$

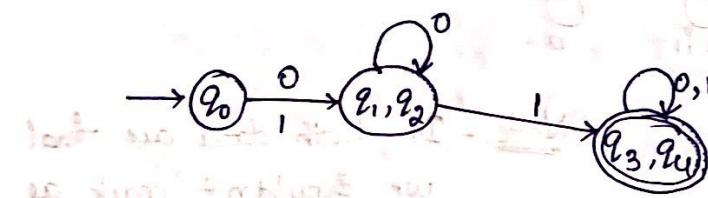
Consider  $(q_3, q_4)$

$$\begin{array}{ll} (q_3, 0) \rightarrow q_3 & (q_3, 1) \rightarrow q_3 \\ (q_4, 0) \rightarrow q_4 & (q_4, 1) \rightarrow q_4 \end{array}$$

$\therefore (q_3, q_4)$  is equal.

$\Rightarrow$  Hence,  $(q_1, q_2)$  are also equal.

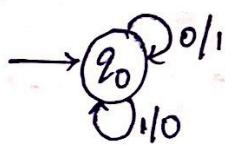
$\Rightarrow$  Drawing output:-



### Transducer Problems:-

Q consider finite state transducers to compute its complement.

Ans.

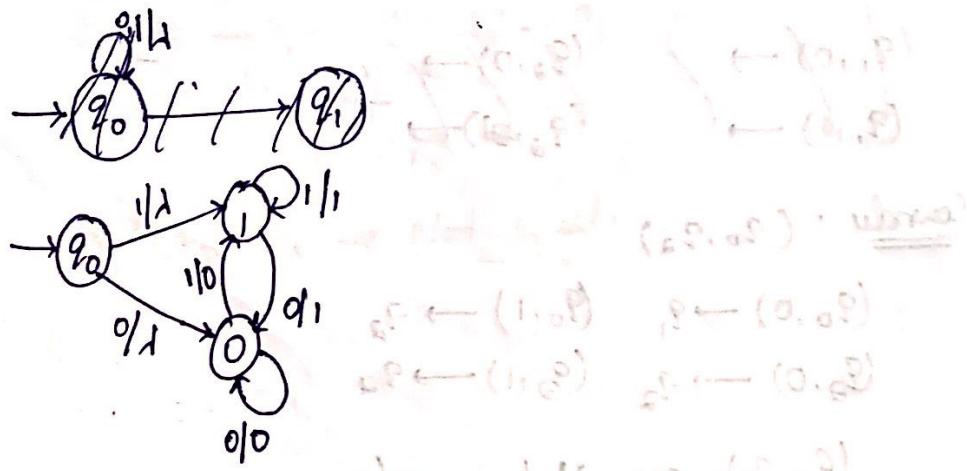


Q Design a unit delay transducer for inputs 0 & 1

Eg.: If I/p: 1011

O/p: 1101

Ans



③ Design a transducer for the input which reads the binary form which is divisible by 3. If the num is divisible by 3 then ans is 0. otherwise output is 1.

④  $L = \{ a^i b^j / i, j > 0 \}$ . Design a transducer if the strings are accepted the result is 1 else the result is 0.

⑤ Design a transducer to convert binary to octal.

### \* DFA:- (Acceptor)

Q. A man travels with wolf, Goat and cabbage hand he wants to cross a river from east to west. A row boat is available only large enough for man along with any one of position. Wolf eats goat if left alone together. Goats eat & cabbage if left alone together. How can man cross without loss.

Hint: 4 moves can be encoded as 4 symbols

1. Man crosses with wolf as (W)

2. " " " Goat as (G)

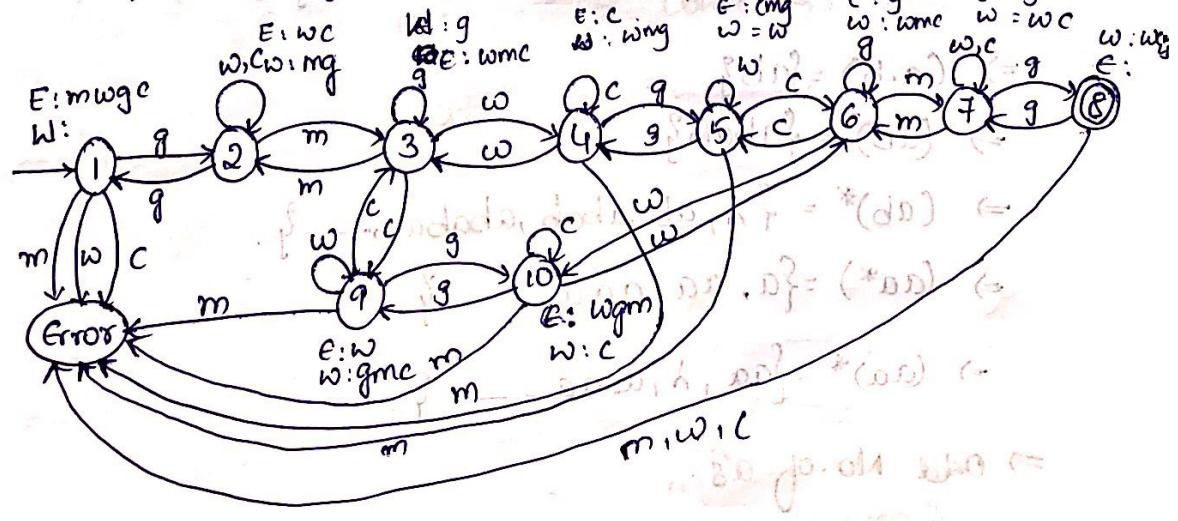
3. " " " Cabbage as (C)

4. " " " Nothing as (N).

$$\Sigma = \{ W, G, C, N \}.$$

Then a sequence of move is a string such as Anwgcng

Sol:-



4/07/2019

## \* $\rightarrow$ Regular Expression:-

Describing Regular language through a notation called regular expression.

→ The notation involves a combination of strings of symbols from some alphabets sigma, parenthesis [ ( ) ],  $\oplus$ ,  $\cap$ ,  $\circ$ .

\* There are Two types of Regular Expressions:-

1. Primitive

2. Composite

1. Primitive:-  $\emptyset$ ,  $\lambda$  and  $a \in \Sigma$

2. Composite:-  $r_1, r_2, r_1 + r_2, r_1 \cdot r_2, r_1^*$  (i.e.,  $r_1 \cdot r_1 \cdot r_1 \dots$ )

(\*) Important:-

\*  $\rightarrow$  (,  $\cup$ +) Same [all are same which gives same output].

(a, b)

(a  $\cup$  b)

(a + b)

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

$\Rightarrow (aa, ab, ba, bb)$

$(a+b)^* = \{\lambda, a, b, ab, abb, \dots\}$

$\Rightarrow (a \cdot b) = \{ab\}$

$\Rightarrow (ab)^2 = \{babab\}$

$\Rightarrow (ab)^* = \{\lambda, ab, abab, ababab, \dots\}$

$\Rightarrow (aa)^* = \{a, aa, aaa, \dots\}$

$\Rightarrow (aa)^* = \{aa, \lambda, aaaa, \dots\}$

$\Rightarrow$  Odd No. of a's.

1Q. Odd No. of a's Construct regular Exp for language which accepts only a/b

Ans:-

$$r = a+b$$

$L(r) = \{w : w^2 = (ab)^k, k \text{ is odd}\}$

2Q. Regular Expression for language ab alone

$$r = a \cdot b$$

3Q. R.E for language which ends with one over the set {0,1}

Ans:-

$$r = (0+1)^* \cdot 1$$

(or)

$$r = (0^* \cdot 1^*)^* \cdot 1$$

4Q. Given  $r = (a+b)^* (a+b b)$

Ans

↓  
all comb of a&b

$\Rightarrow L(r) = \{ \text{all strings with ending with } a \text{ or } bb \}$ .

5Q. Given  $r = (aa)^* (bb^*)^* \cdot b$

Ans

↓      ↓  
aa → all    bb → all

$L(r) = \{ \text{Strings with even no. of a's and odd b's} \}$

$L(r) = \{ a^{2n} b^{2m+1} ; n \geq 0, m \geq 0 \}$ .

6Q.  $\Sigma = \{0,1\}$  give a RE such that  $L(r) = \{ w \in \Sigma^* \text{ where } w \text{ has atleast one pair of consecutive zeros.} \}$

Ans:-

$$r = [(00)^* + (0,1)^*]^+$$

0|00  
0 0 0 | 0 1 0 0 0 |

7Q. Find regular Exp for  $L(r) = \{ a^n b^m, n > 3, m \text{ is even} \}$ .

Ans:-

$$r =$$

(ii)  $L(r) = \{ a^n b^m, (m+n) \text{ is even} \}$

(iii)  $L(r) = \{ a^n b^m, n \geq 4, m \leq 3 \}$ .

(iv)  $L(r) = \{ w \in \{0,1\}^* \text{ where } w \text{ has no pair of consecutive zeros}\}$

Q. Give re for  $L(r) = \{ w : |w| \bmod 3 = 0\}$

5/07/19

\*  $\rightarrow$  Chomsky Hierarchy:-

Type 0, 1, 2, 3 [Grammae]

\*  $\rightarrow$  Type 0 (or) Recursively Renewable Grammae:-

$$\alpha \rightarrow \beta$$

$$\alpha \in (VUT)^+$$

$$\beta \in (V \cup U T)^*$$

Eg.:  $S \rightarrow AcaB$

$$Bc \rightarrow acB$$

$$CB \rightarrow DB$$

$$aD \rightarrow Db$$

\*  $\rightarrow$  Type 1 Grammae:-

Context Sensitive Grammae.

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

A  $\rightarrow$  Non-terminal

$$\alpha, \beta, \gamma \in (\Gamma UV)^*$$

$\gamma \rightarrow$  can't be empty

$S \rightarrow e$   $\rightarrow$  Is possible only if S doesn't appear on the right hand side of any value.

Eg.:  $AB \rightarrow AbBC$

$$A \rightarrow bCA$$

$$B \rightarrow b$$

### 3. Type-2 Grammae:-

Context free Grammae.

$$A \rightarrow \gamma$$

$A \rightarrow$  Non-terminals

$$\gamma \in (\Sigma \cup V)^*$$

### 4. Type-3 Grammae:-

Regular Grammae.

$$S \rightarrow Aa \text{ (or) } S \rightarrow aA$$

$$S \rightarrow \Sigma / \lambda$$

$$S \rightarrow a$$

\* Right side either one /  $\lambda$  / combined with  $a$  should be there.

→ This is done by finite automata.

$$S \rightarrow \Sigma / A$$

$S \rightarrow aSb|bSa \rightarrow$  middle linear.  $\rightarrow$  only for LG not for R.G

\* Right side either one /  $|$  combined with  $\varnothing$  should be there.

$\rightarrow$  This is done by finite automata.

17/7/19

\* \* Relationship b/w Regular Expression and Regular Language:-

$$RL \leftrightarrow RE$$

Def: Language is Regular if it is accepted by 'DFA'.

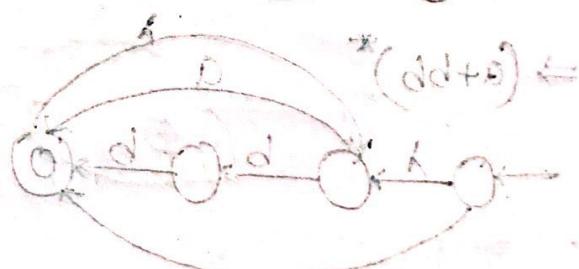
Because of the equivalence of NFA's and DFA's a language is accepted by NFA is also called as Regular language.

Q. How to construct NFA from a Regular Expression?

Sol: \* Rule :- (Thompson's Rule)

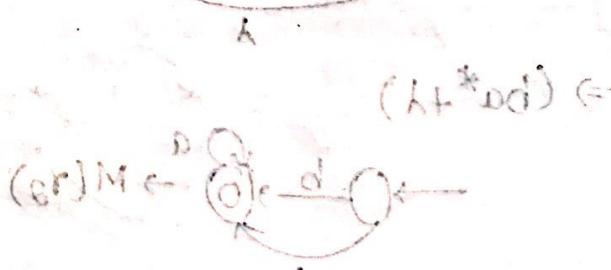
① NFA accepts  $\emptyset$  (empty)

Eg:  $\rightarrow q_0 \xrightarrow{\quad} q_1$

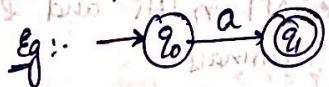


② NFA accepts  $\{\lambda\}$

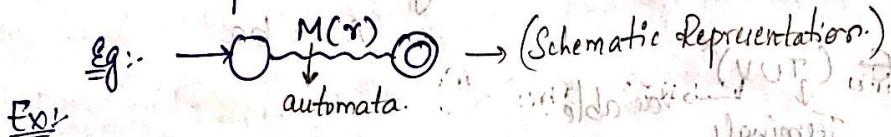
Eg:  $\rightarrow q_0 \xrightarrow{\lambda} q_1$



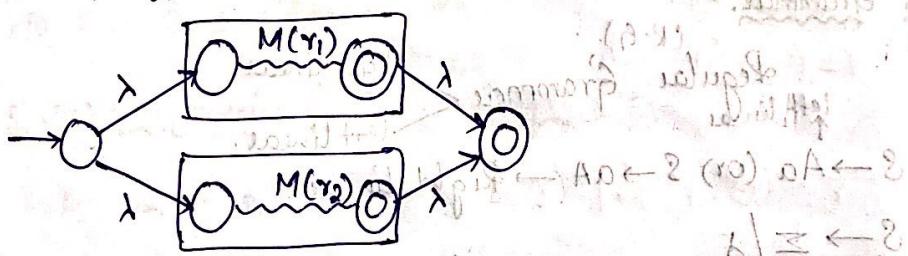
③ NFA accepts any alphabet  $\Rightarrow \{a\}$



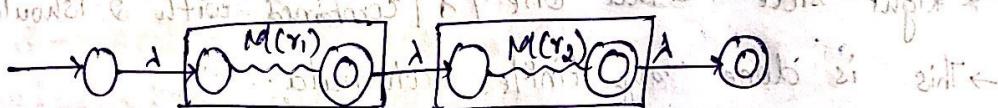
④ Schematic representation of an NFA accepting  $L(r)$  (Regular language).



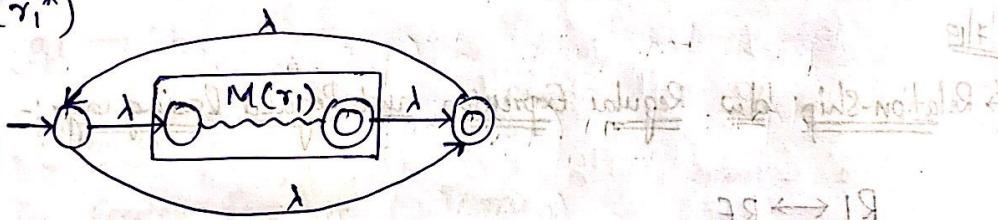
Ex: ②  $\Rightarrow L(r_1 + r_2)$



③  $L(r_1 r_2)$



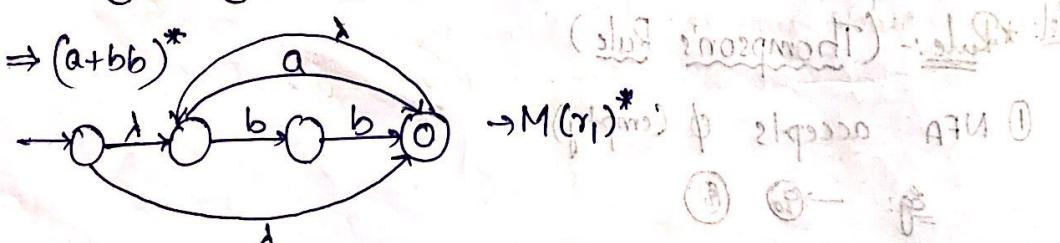
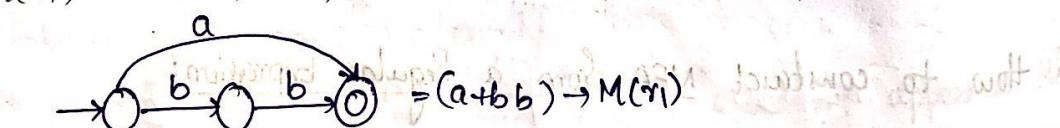
④  $L(r_1^*)$



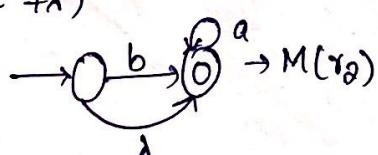
\*Problems:-

IQ. Find an NFA that accepts  $L(r)$  where  $r = (a+bb)^*(ba^*+\lambda)$

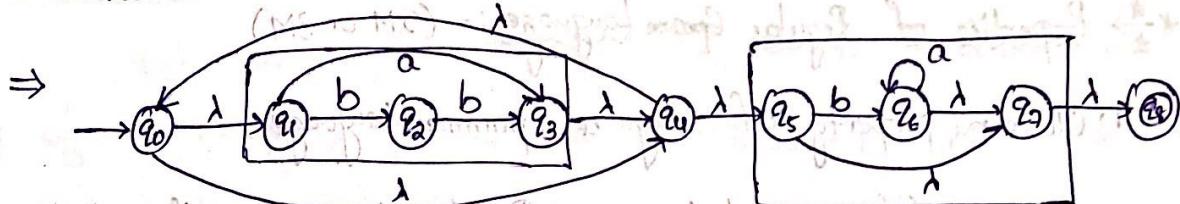
Sol:  $L(r_1) = abb, bba, \lambda, aabb, abbb \dots$



$\Rightarrow (ba^* + \lambda)$



Concatenate  $M(r_1)^* \cdot M(r_2)$ .

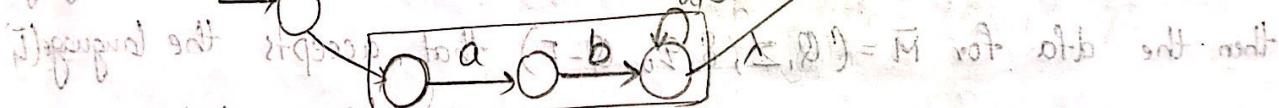
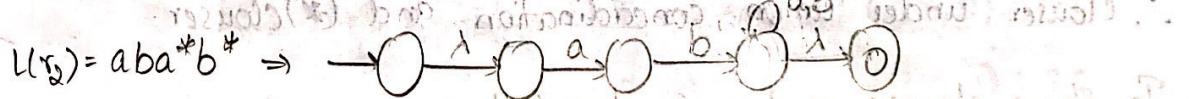


20. Find an NFA for  $L(aa^* + aba^*b^*)$ .

- (ii)  $L((a+b)^* b(a+bb)^*)$
- (iii)  $L(ab^*aa + bba^*ab)$

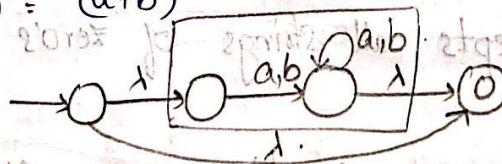
Sol: (i)  $L(aa^* + aba^*b^*)$ .

$\{a, aa, aaa, \dots, ab, aba, abab, abaab, ababb, \dots, aab, aababa\}$

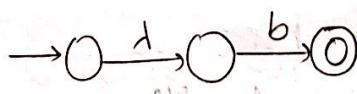


(ii)  $L((a+b)^*.b(a+bb)^*)$ .

$$L(r_1) = (a+b)^*$$



$$L(r_2) = b(a+bb)^*$$



19/7/19

\*  $\rightarrow$  Properties of Regular Gram Language :- (5M or 3M)

(1) Closer property of Regular Grammar/Language:-

If  $L_1$  and  $L_2$  are Regular languages then  $L_1 \cup L_2$ ,  $L_1 \cap L_2$ ,  $L_1 \cdot L_2$ ,  $L_1^*$  will also produce regular language if it is closure under the operations.

\* Proof:-

If  $L_1$  and  $L_2$  are regular then there exists  $r_1$  and  $r_2$  such that  $L_1 = L(r_1)$  and  $L_2 = L(r_2)$ .

By def:-  $r_1 + r_2$ ,  $r_1 \cdot r_2$ ,  $r_1^*$  are also regular expression

$\Downarrow$   
 $\therefore L(r_1 + r_2)$ ,  $L(r_1 \cdot r_2)$ ,  $L(r_1)^*$  are also regular languages.

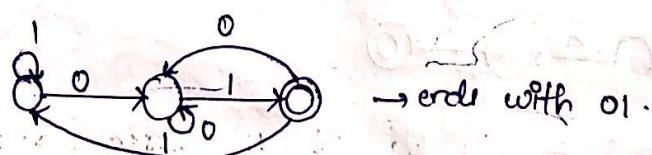
$\therefore$  Closer under union, concatenation and (\*) closer.

\*  $\rightarrow$  To show closer under complementation:-

Let  $M = (Q, \Sigma, \delta, q_0, F)$  be dfa that accepts the lang then the dfa for  $\bar{M} = (Q, \Sigma, \delta, q_0, Q - F)$  that accepts the lang where the transitions are  $\delta^*(q_0, w)$  is a final state we.  $\delta^*(q_0, w) \in Q - F$  and  $w \in \Sigma$ .

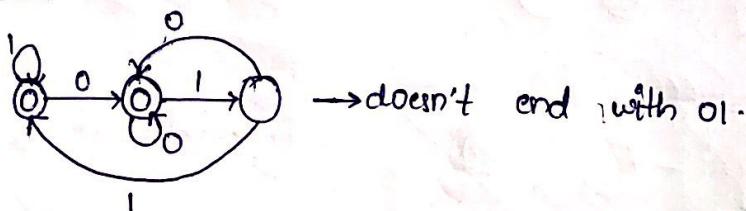
Eg:- Design a dfa that accepts all strings of zero's and ones doesn't end with 01.

Sol:- Construct dfa that accepts all strings ends with 01 and do complement of it.



→ ends with 01.

Complementation:-



→ doesn't end with 01.

## \* Clouser Under Intersection:-

Let  $L_1 = L(M_1)$  and  $L_2 = L(M_2)$  where  $M_1 = (Q, \Sigma, \delta_1, q_0, F_1)$

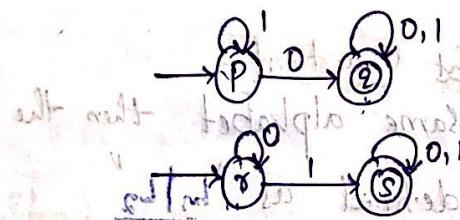
and  $M_2 = (P, \Sigma, \delta_2, p_0, F_2)$  are dfa's [as  $\Sigma$  is same for  $M_1, M_2$  we can do it]

Construct from  $M_1$  and  $M_2$  a combined automaton.

$\hat{M} = (\hat{Q}, \Sigma, \hat{\delta}, (q_0, p_0), \hat{F})$ , where  $\hat{Q} = P \times Q \rightarrow (q_i, p_j)$

$\hat{F} = (q_i, p_j)$  where  $q_i \in F_1$  &  $p_j \in F_2$ ;  $\hat{\delta} = ((q_i, p_j), a) = (q_k, p_l)$

Eg:- Consider dfa's

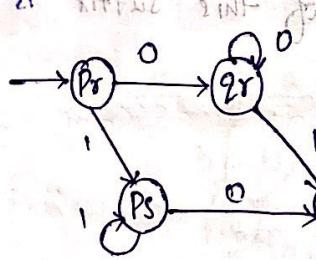


$$\hat{Q} = \{p_{qr}, p_{rs}, p_{qs}\}; \hat{\delta} = (p_{qr}, 0) : q_r \mid (q_r, 0) : q_r$$

$$(p_{rs}, 1) : p_s \mid (q_r, 1) : q_s$$

$$\rightarrow (p_{rs}, 0) : q_s \mid (q_s, 0) : q_s$$

$$(p_{qs}, 1) : p_s \mid (q_s, 1) : q_s$$



[only if (q\_s) are both final states  
make it as final state]

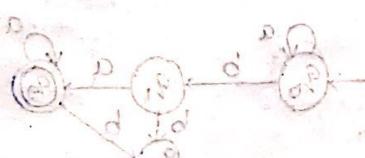
(\*) Other Way:-  $\bar{L}_1, \bar{L}_2$ , odd, odd, odd, odd, odd, odd, odd }

$$\bar{L}_1 \cup \bar{L}_2$$

$$\Rightarrow \boxed{\bar{L}_1 \cup \bar{L}_2 = L_1 \cap \bar{L}_2}$$

\* Show that the family of RL is closed under difference :-

$$\Rightarrow \boxed{L_1 - L_2 = L_1 \cap \bar{L}_2}$$



23/7/19

- \*  $\rightarrow$  Regular Language closed under Reverse: (GM) 1.3.1.1.1.1.
- In the transition graph make the initial vertex as final vertex and reverse the direction of all the edges:  $(L^R)$
- V.V. Imp
- \*  $\rightarrow$  Regular Language closed under Homomorphism: (GM) 1.3.1.1.1.1.
- (It is easy if you learn it on your own. 😊)

Right Quotient:  
 $\rightarrow$  Regular Language closed under Right Quotient:  
Let  $L_1$  and  $L_2$  be languages on same alphabet then the right quotient of  $L_1$  with  $L_2$  is defined as  $L_1 / L_2$

$$L_1 / L_2 = \{x \mid xy \in L_1 \text{ for some } y \in L_2\}$$

- $\rightarrow$  To form the right quotient of  $L_1$  with  $L_2$ , take all strings in  $L_1$  that have a suffix belonging to  $L_2$ .
- $\rightarrow$  Every such string after removal of this suffix is called as  $L_1 / L_2$ .

Eg.: let  $L_1 = L(a^*baa^*)$

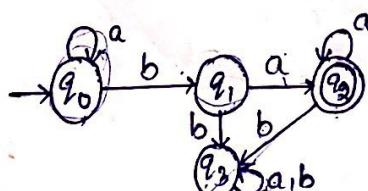
$$L_2 = L(ab^*) \text{ find } L_1 / L_2$$

$$L_1 = \{ba, ab, abaa, aaba, aabaa, \dots\}$$

$$L_2 = \{a, ab, abbb, abb, abbb, \dots\}$$

$$L_1 / L_2 = \{b, ab, aba, \dots\}$$

Step 1: draw dfa for  $L_1$



$$[c \cap a, b = c - b] \leftarrow$$

Step 2:- Find from each state collect the languages and do intersection with  $L_2$ .

$$L(M_0) \cap L_2 = \emptyset \quad [\text{as there is no intersection with } L_2]$$

$$L(M_1) \cap L_2 = \{a\} \neq \emptyset$$

$$L(M_2) \cap L_2 = \{a\} \neq \emptyset$$

$$L(M_3) \cap L_2 = \emptyset$$

$$L(M_0) = \{aba, aaba, abaa, \dots\}$$

$$L(M_1) = \{a, aa, aaa, \dots\}$$

$$L(M_2) = \{a, \dots\}$$

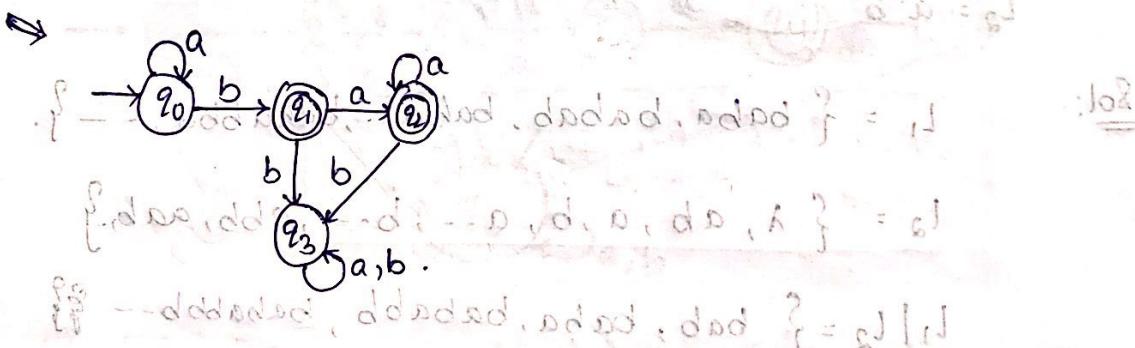
$$L(M_3) = X.$$

Step 3:- If  $L(M_i) \cap L_2 = \emptyset \Rightarrow$  then don't change anything.

$L(M_i) \cap L_2 \neq \emptyset \Rightarrow$  if it's not final state then make it as final state.

" "  $\Rightarrow$  if it's final leave (don't change).

$L(M_i) \cap L_2 = \emptyset \Rightarrow$  if  $M_i$  is final state then make it as non-final state.



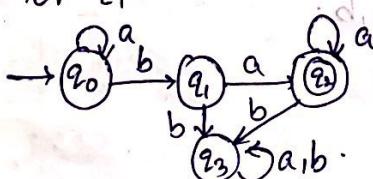
Q. Let  $L_1 = L(a^*baa^*)$ ,  $L_2 = L(aba^*)$  find  $L_1 \cap L_2$ .

Sol.:  $L_1 = \{ba, aba, abaa, aabaa, \dots\}$

$$L_2 = \{ab, aba, abaa, \dots\}$$

$$L_1 \cap L_2 = \{a, \dots\}$$

Step 1: dfa for  $L_1$



Step 2:-  $L(M_0) = \{ aba, aaba, abaa, \dots \}$

$$L(M_1) = \{ a, aa, \dots \}$$

$$L(M_2) = \{ a, aa, \dots \} \quad \emptyset = L(M)$$

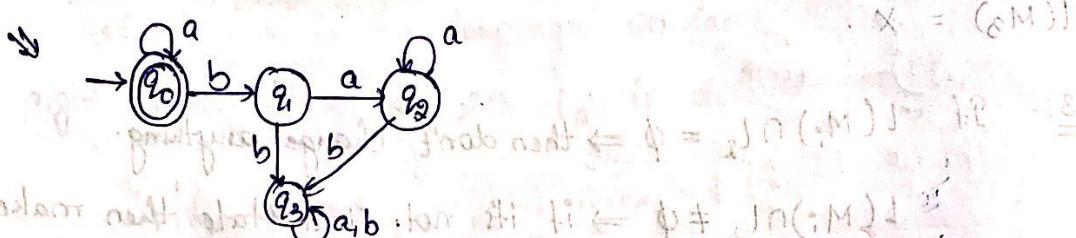
$$L(M_3) = X$$

$$L(M_0) \cap L_0 = \{ aba, aaba, \dots \} \neq \emptyset$$

$$L(M_1) \cap L_2 = \{ a, aa, \dots \} \neq \emptyset = \emptyset$$

$$L(M_2) \cap L_2 = \{ a, \emptyset \}$$

$$L(M_3) \cap L_2 = X$$



Q.  $L = babab^*a^*$  find  $L_1 \cap L_2 = \emptyset$

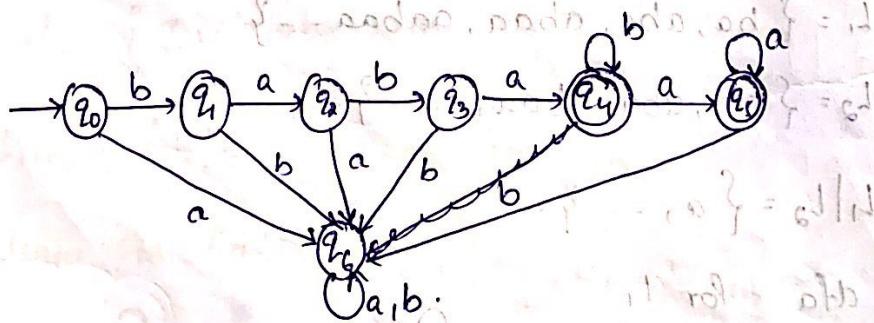
$$L_2 = a^*b^*$$

Sol:  $L_1 = \{ baba, babab, babaa, bababba, \dots \}$

$$L_2 = \{ \lambda, ab, a, b, a\dots, b\dots, abb, aab, \dots \}$$

$$L_1 \cap L_2 = \{ bab, baba, bababb, bababbb, \dots \}$$

①  $\Rightarrow$  DFA for  $L_1$ .



$$\textcircled{2} \quad L(M_0) = \{babab^*, babab^{*(a+d)}\}, \quad \text{Ans}$$

$$L(M_1) = \{babab^*a^*\}$$

$$L(M_2) = \{bab^*a^*\}$$

$$L(M_3) = \{ab^*a^*\}$$

$$L(M_4) = \{b^*, a^*, b^*a^*\}$$

$$L(M_5) = \{a^*\}$$

$$L(M_6) = \emptyset$$

$$L(M_0) \cap L_2 = \emptyset$$

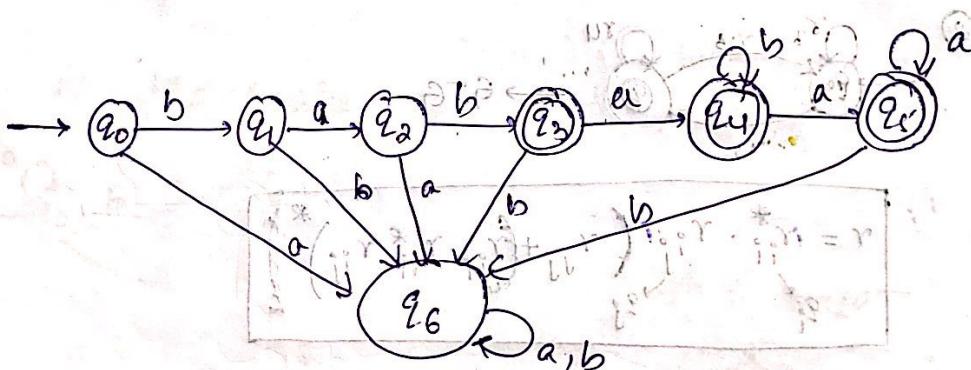
$$L(M_1) \cap L_2 = \emptyset$$

$$L(M_2) \cap L_2 = \emptyset$$

$$L(M_3) \cap L_2 \neq \emptyset$$

$$L(M_4) \cap L_2 \neq \emptyset$$

$$L(M_5) \cap L_2 \neq \emptyset$$



24/7/19 \*Homomorphism:-

Suppose ~~both~~  $\Sigma$  &  $\Gamma$  are alphabets then a function

$h: \Sigma \rightarrow \Gamma^*$  is called "Homomorphism".

→ Homomorphism is a substitution in which a single letter is replaced with a string.

→ If  $w = a_1, a_2, \dots, a_n$  then  $h(w) = h(a_1)h(a_2)\dots h(a_n)$

Q: Let  $\Sigma = \{a, b\}$  and  $\Gamma = \{a, b, c\}$  and  $h$  is defined as

$h(a) = ab$ ,  $h(b) = bbc$ , then find  $r = (a + b^*)(aa)^*$

$$\text{Sol: } r = (ab + bba^*)^*(abab)^*$$

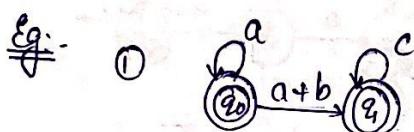
just substitute  $a(a)$  in place of  $a$  and  $b(b)$  in place of  $b$ .

### \* Regular Expression For Regular Languages

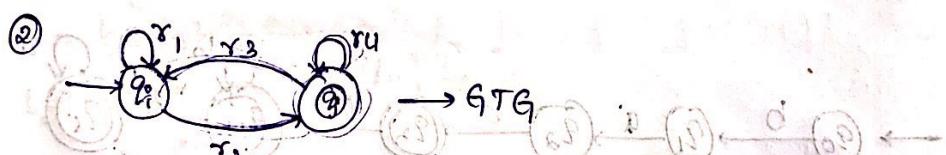
There are several ways to find the regular expression from regular languages.

#### 1. GTG (Generalized Transition Graph)

\* GTG: It is a transition graph whose edges are labeled with Regular Expressions.



$$RE = a^* (a+b) c^* \quad a^* + [a^* + (a+b) + c^*]$$

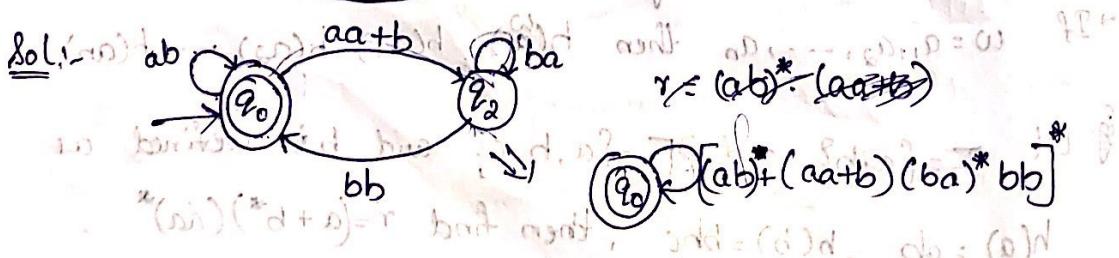
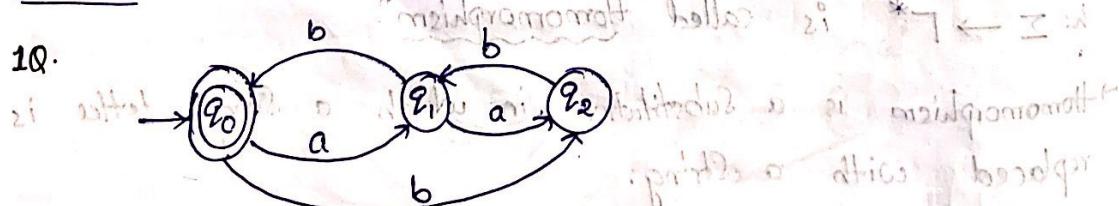


$$r = r_{ii}^* \cdot r_{ij} \left( r_{jj} + (r_{ji} \cdot r_{ii}^* \cdot r_{ij})^* \right)$$

$$r = r_1^* r_2 (r_4 + r_3 r_1^* r_2)^*$$

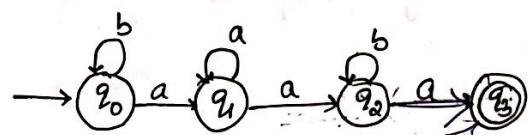
#### Problems:-

10.

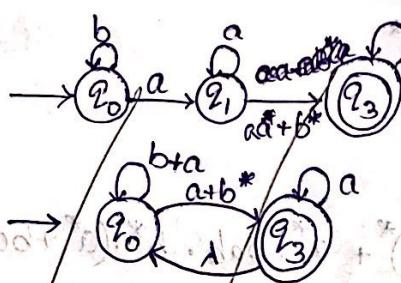


26/7/19

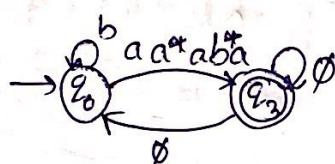
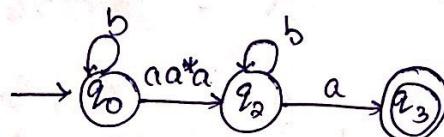
28 Find RE for FA-



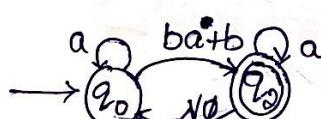
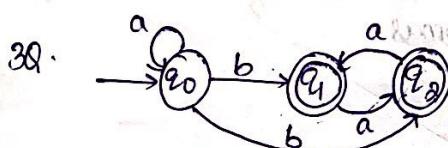
A.M. :-



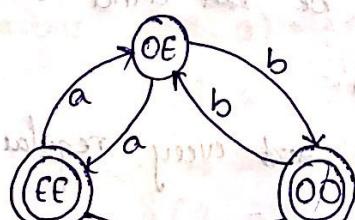
$$\Rightarrow RE \neq (b+a) (a+b^*) [a + (b+a)^* (a+b^*)]^*$$



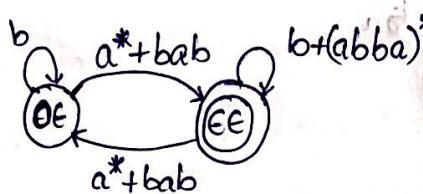
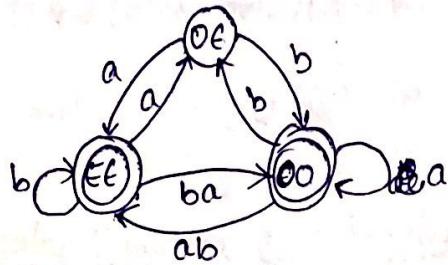
$$RE = b^* a a^* a b^* a$$



$$\Rightarrow a^*(ba^* + b)[a + a^*ba^*b]^* \Rightarrow a^*(b+ba^*)(aa)^* \Rightarrow aba^*.$$



Any



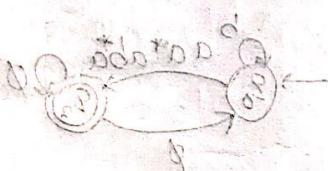
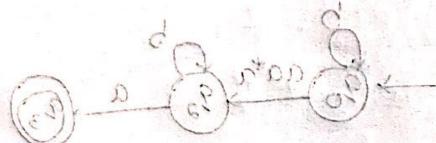
$$RE = b^* \cdot (a^* + bab) [b + (abba)^* + a^* + bab \cdot b^* \cdot (a^* + bab)]^*$$

31/7/19

\*  $\rightarrow$  Regular Grammar:-

↓  
left

$S \rightarrow$  terminal Non-terminal  
 $A \rightarrow$  terminal



Eg:-  $S \rightarrow a(A) \rightarrow$  right side (NT)  
 $A \rightarrow \lambda \mid b \mid aa$

\* This is called Right linear Grammar.

$S \rightarrow (A)a$

$A \rightarrow b \mid S$

\* \* \* This is left linear Grammar.

\* \* Combination of both is called "Linear Grammar".

\* Note:- The Regular Grammae can only be both either left or right linear Grammae but not both.

\* Not Every linear is regular but ~~most~~ every regular is linear grammar.

\*  $\rightarrow$  Linear Grammae:- It is a grammar in which almost one variable can occur on the right side of production without restriction on the position of the variable.

→ Right Linear Grammar :-

A grammar  $G = (V, T, S, P)$  is said to be right linear if all the prod of the form  $A \rightarrow xB$  where  $A, B \in V$  and  $x \in T^*$

→ Left LG :-

$$A \rightarrow Bx$$

$$B \rightarrow x$$

$$A \rightarrow xB$$

$$B \rightarrow x$$

→ Ex :-  $0S \rightarrow abS/a \rightarrow$  Right LG

$$\textcircled{2} \quad S \rightarrow S, ab$$

$$S_1 \rightarrow S, ab | S_2 \quad \left. \begin{array}{l} \\ \end{array} \right\} \rightarrow \text{Left LG.}$$

$$S_2 \rightarrow a$$

$$\textcircled{3} \quad S \rightarrow A$$

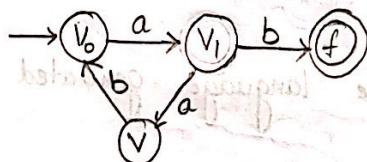
$$A \rightarrow aB | \lambda \quad \left. \begin{array}{l} \\ \end{array} \right\} \rightarrow \text{linear Grammar}$$

$$B \rightarrow Ab \quad \left. \begin{array}{l} \\ \end{array} \right\} \rightarrow \text{Not regular Grammar.}$$

Problem :-

① Construct a finite automata that accepts the language generated by the grammar  $V_0 \rightarrow aV_1$ ,  $V_1 \rightarrow abV_0 | b$

Sol.:



$$\Rightarrow (aab)^*ab.$$

② Construct a right linear grammar for the given language  $L(aab^*)a$

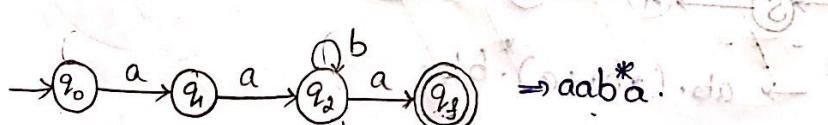
Transitions  $\delta(q_0, a) = \{q_1\}$  and convert Right LG to Left LG?

$$\delta(q_1, a) = \{q_2\}$$

$$\delta(q_2, b) = \{q_3\}$$

$$\delta(q_3, a) = \{q_f\}$$

Sol.:

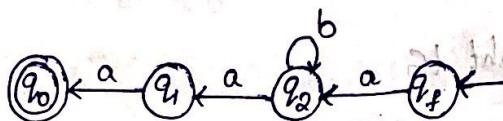


$$\Rightarrow aab^*a.$$

$\text{RLG} :-$   
 $q_0 \rightarrow a q_1$   
 $q_1 \rightarrow a q_2$   
 $q_2 \rightarrow b q_3$   
 $q_3 \rightarrow a q_f$   
 $q_f \rightarrow \lambda$

\* Converting to left:-

① Find reverse to the transition graph.



②

$$q_f \rightarrow a q_2$$

$$q_3 \rightarrow b q_2$$

$$q_0 \rightarrow a q_1$$

$$q_1 \rightarrow a$$

③ Reverse the above grammars + production.

$$q_f \rightarrow q_2 a$$

$$q_3 \rightarrow b q_2$$

$$q_0 \rightarrow q_1 a$$

$$q_1 \rightarrow a$$

(\*) This is the left linear grammar

③ Construct a <sup>automata</sup> dfa that accepts the language generated by <sup>th</sup> grammars

$$S \rightarrow abA$$

$$A \rightarrow baB$$

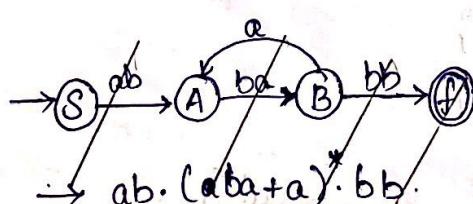
$$B \rightarrow aA \mid bb$$

④ Convert the given RLG to LLG. (above grammar)

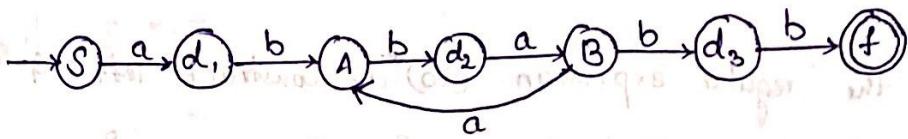
⑤ Find a regular grammar that generates the language  $L(a^*(ab))$

Answers:-

Ans :-



$$ab \cdot (abab + a)^* \cdot bb$$

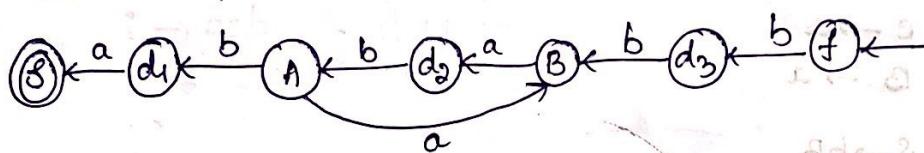


$$R_{DL} = ab \cdot (baa)^+ \cdot bb$$

$f^*(d_2) \text{ does not have } ab \text{ as a suffix}$

Ans:-  $S \rightarrow abA$   
 $A \rightarrow baB$   
 $B \rightarrow aa \mid bb$

Reverse RLG



$$\begin{cases} f \rightarrow Bbb \\ B \rightarrow Aba \\ A \rightarrow Ba \mid Sab \end{cases}$$

$$\begin{cases} f \rightarrow d_3 b \\ d_3 \rightarrow Bab \\ B \rightarrow da \\ da \rightarrow Ab \\ A \rightarrow Ba \mid dabb \end{cases}$$

$$\begin{cases} da \rightarrow Sa \\ Sa \rightarrow \lambda \end{cases}$$

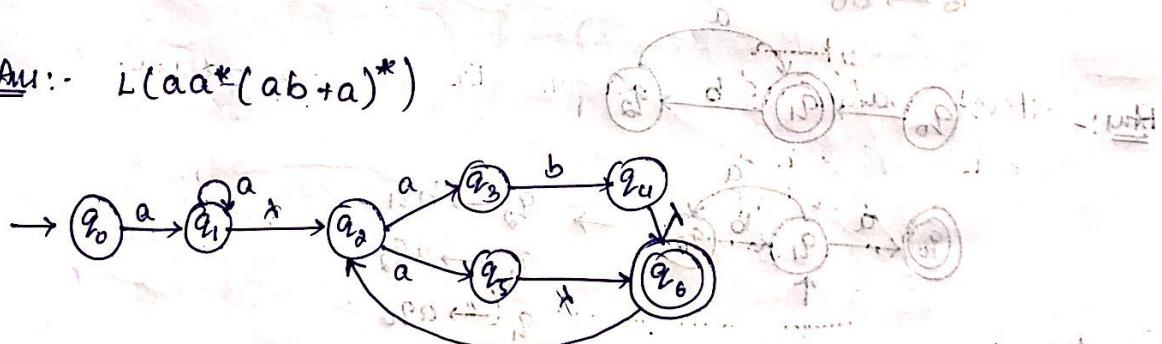
$$\Rightarrow f \rightarrow Bbb$$

$$B \rightarrow Aba$$

$$A \rightarrow Ba \mid Sab$$

$$S \rightarrow \lambda$$

Ans:-  $L(aa^*(ab+a)^*)$



$$q_0 \rightarrow aa,$$

$$q_1 \rightarrow aq_2,$$

$$q_1 \rightarrow q_2$$

$$q_2 \rightarrow aq_3$$

$$q_3 \rightarrow bq_4$$

$$q_4 \rightarrow q_5$$

$$q_5 \rightarrow aq_6$$

$$q_6 \rightarrow q_7 ; q_7 \rightarrow q_8$$

$$ld \leftarrow \varnothing$$

$$ld \leftarrow \{ \varnothing \}$$

$$ld \leftarrow \{ \varnothing \}$$

$$ld \leftarrow \varnothing$$

$$ld \mid ld \leftarrow \varnothing \quad (r)$$

$$ld \leftarrow \varnothing \quad (l)$$

$$ld \mid ld \leftarrow \varnothing \quad (l)$$

- 28/19
- Consider the regular expression  $(ab)^*a$ ; Construct left L.G.
  - Construct RLG for the language  $\{a^n b^m \mid n \geq 2, m > 3\}$ .
  - Construct LLG for the  $L = \{aab(ab)^*\}$
  - Convert the given right linear to left linear

(1)  $S \rightarrow bB$

$B \rightarrow bc / aB$

$c \rightarrow a$

$B \rightarrow a$



(2)  $S \rightarrow bB$

$B \rightarrow bc / aB / b$

$c \rightarrow a$

(3)  $S \rightarrow OA$

$A \rightarrow 1OA / d$

$b / ab / dd \rightarrow A$

5. Construct DFA that accepts the language.

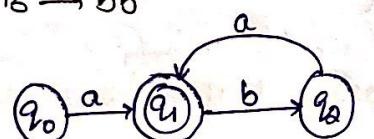
$S \rightarrow aba$

$A \rightarrow baB$

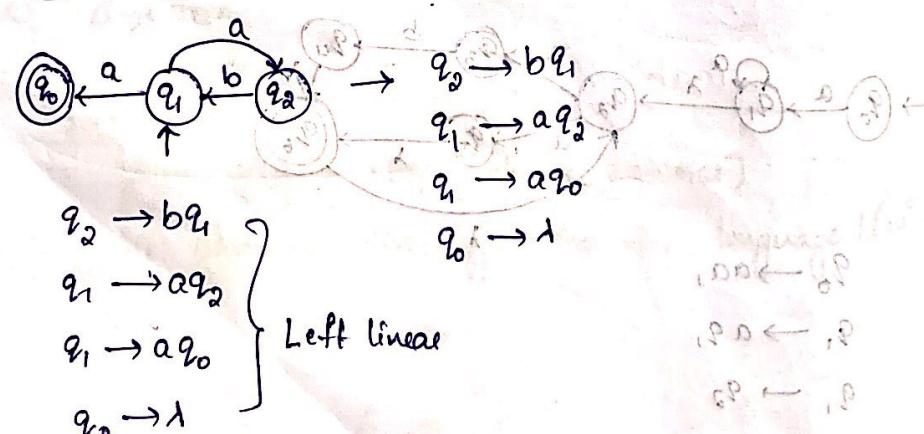
$B \rightarrow aA$

$B \rightarrow bb$

Ans:-



$$\begin{aligned} ddg &\leftarrow b \\ dDA &\leftarrow d \\ dae &\leftarrow A \\ h &\leftarrow S \end{aligned}$$



Left linear

(or)  $S \rightarrow a / abS$

(or)  $S \rightarrow S_1 a$

$S_1 \rightarrow S_1 ab / \lambda$

Ques 1:-  $\{a^n b^m \mid n \geq 2, m \geq 3\}$

aabb

$S \rightarrow aaA$

$A \rightarrow aA \mid bbbB$

$B \rightarrow bB \mid \lambda$

Ans:-  $\{aab(ab)^*\}$

$S \rightarrow aab \mid Sab$

$S \rightarrow aabA$

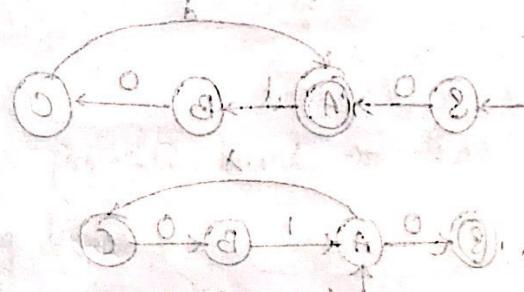
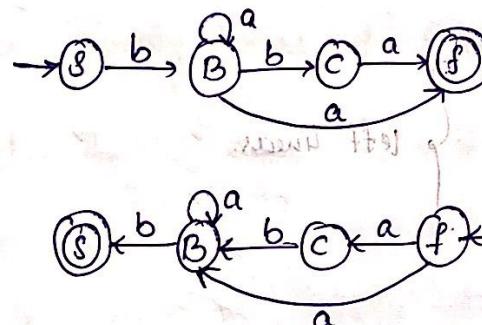
$A \rightarrow abA$

Ans:- (i)  $S \rightarrow bB$

$B \rightarrow bC \mid aB$

$C \rightarrow a$

$B \rightarrow a$ .



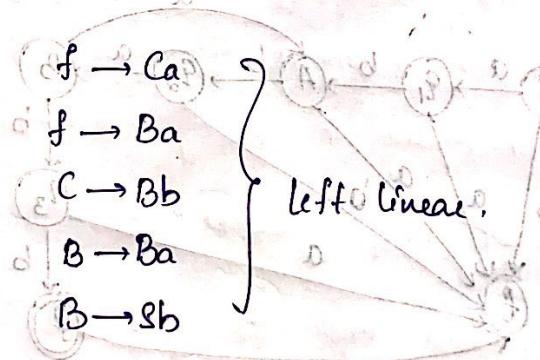
$f \rightarrow ac$

$f \rightarrow aB$

$C \rightarrow bB$

$B \rightarrow aB$

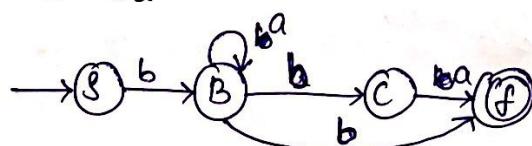
$B \rightarrow bS$

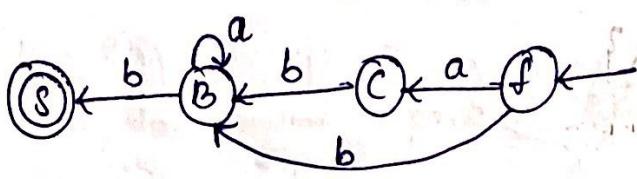


(ii)  $S \rightarrow bB$

$B \rightarrow bc \mid aB \mid b$

$C \rightarrow a$





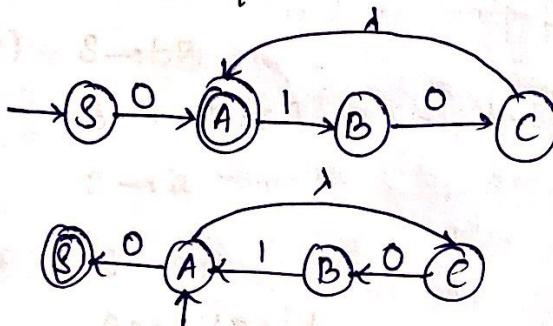
$f \rightarrow ac$   
 $f \rightarrow bB$   
 $C \rightarrow bB$   
 $B \rightarrow ab$   
 $B \rightarrow bs$

$f \rightarrow Ca$   
 $f \rightarrow Bb$   
 $C \rightarrow Bb$   
 $B \rightarrow Ba$   
 $B \rightarrow Sb$

left linear

(3)  $S \rightarrow 0A$

$A \rightarrow 10A / \lambda$

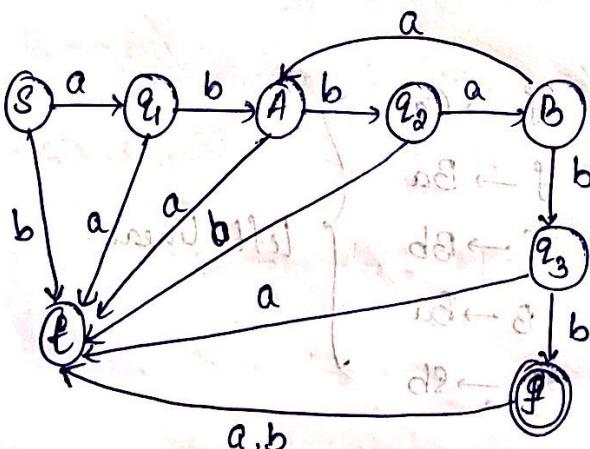


$A \rightarrow 0S$   
 $A \rightarrow C$   
 $C \rightarrow 0B$   
 $B \rightarrow 1A$

$A \rightarrow S0$   
 $A \rightarrow C$   
 $C \rightarrow B0$   
 $B \rightarrow AI$

left linear.

5Am :-



6/8/19

\* Identify Non-Regular Languages:-  
(Can't draw FA, dfa)

Eg:  $L = \{a^n b^n; n \geq 0\}$

$\{a, ab, aabb, \dots\} \rightarrow$  we can't draw the dfa bcoz it can't store the  $n$  value  $\rightarrow$  Non-regular

\*  $L = \{a^n b^m, n \geq 0, m \geq 0\} \rightarrow$  This can have a dfa as there is no relation b/w  $n$  &  $m$  so, it is regular grammar

\* Pumping Lemma:- (Game)

It uses pigeon hole principle based on that it will use a game called pumping lemma.

① Use pumping lemma to show that  $L = \{a^n b^n, n \geq 0\}$  is Non-regular

Step 1:- The opponent is asked to pick a m value.  $\Rightarrow m=3$ .

Step 2:- Now choose a string  $w$  from  $L$  in such a way its length is equal or greater than  $m$ . ( $|w| \geq m$ )  $= \{aabbb\}$

Step 3:- choose the decomposition.  $w_i = xyz$

where  $|xy| \leq m$  &  $|y| \geq 1$

a/a/bb  
x/y/z

Step 4:- Try to pick the value of  $i$  in such a way that pumped string  $w_i$  is not in  $L$ .

$i=0; w=abb$  (doesn't belong to given language)

$\therefore$  It is not a RG as there is atleast one  $i$  where  $w_i$  is not the given string in language.

Q. S.T.  $L = \{(ab)^n a^k : n > k, k \geq 0\}$ .

$$L = \{ab, ababa, \dots\}.$$

Sol.: Step 1:  $m = 2$ .

Step 2:  $w = \{abab\}$

Step 3:  $w_0 = \underline{a}y\underline{zb}$  or  $w_0 = \underline{a}\underline{y}\underline{z}b$

where  $|ay| \leq m$  &  $|y| \geq 1$

$$\begin{array}{c} abab \\ \cancel{a} \cancel{y} \cancel{b} \\ ab \\ \cancel{a} \cancel{y} \cancel{b} \end{array}$$

Step 4:  $i=0 \Rightarrow aab \notin L$ .

Hence it is proved.

Q.  $L = \{a^n b^k c^{n+k} : n \geq 0, k \geq 0\}$ . P.T. if it is not regular using pumping

3Q.  $L = \{a^n : n \text{ is a perfect square}\}$

4Q. S.T.  $L = \{w.w^R : w \in \Sigma^*\}$ . P.T.  $\Sigma = \{a, b\}$

\*  $\rightarrow$  Context free Grammar - (We can't draw dfa).

A Grammar  $G = (V, T, S, P)$  is said to be context free if all productions in  $P$  of the form  $A \rightarrow x$ ;  $A \in V$ ,  $x \in V^*$

\* The language CFG generates CFL.

(\*) Regular Grammars and LG are context free, but a context free grammar is not necessarily linear or regular grammar.

1Q. Construct a CFG where the string starts and ends with same symbol. The symbols are  $\Sigma = \{a, b\}$

Sol.:  $S \rightarrow aAa \mid bBb \mid A$

$A \rightarrow aA \mid Aa \mid BA \mid Ab$

$B \rightarrow \dots$

Q. Find the context free grammar for (i)  $L = \{a^n b^m; n \leq m+3\}$

(ii)  $L = \{a^n b^m; 2n \leq m \leq 3n\}$

(iii)  $L = \{a^n b^m c^k; k = n+m, m, n \geq 0\}$

7/8/19

$\xrightarrow{*}$  Left Most and Right Most derivation:-

Eg: Grammar  $G = \{(\{A, B, S\}), \{a, b\}, S, P\}$ .

$$S \rightarrow AB$$

$$A \rightarrow aaA$$

$$A \rightarrow \lambda$$

$$B \rightarrow Bb$$

$$B \rightarrow \lambda$$

$$\text{LM} \quad S \rightarrow A-B$$

$$\downarrow$$

$$aaA-B$$

$$\downarrow$$

$$aa\lambda-B$$

$$\downarrow$$

$$Bb$$

$$\downarrow$$

$$\lambda b$$

$$RM \quad S \Rightarrow AB$$

$$\downarrow$$

$$A-Bb$$

$$\downarrow$$

$$aaA$$

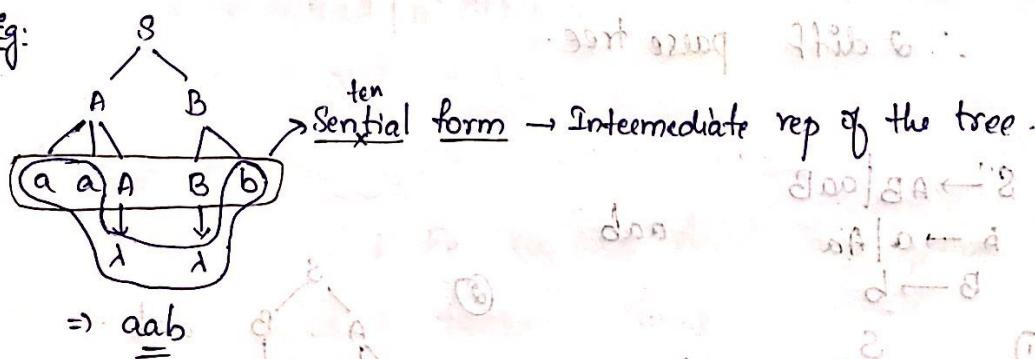
$$\downarrow$$

$$\lambda$$

$\xrightarrow{*}$  Derivation Tree:-

The another way of showing derivations, independent of the order in which productions are used is called a derivation tree or parse tree.

Eg:



\* \* Yield of the tree:-  
The string of symbols obtained by reading the leaf's of the tree from left to right, vomiting 1's is said to be yield of the tree.

\*<sup>\*</sup>→ Ambiguous Grammar :-

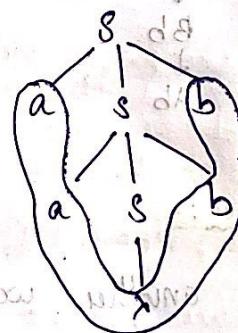
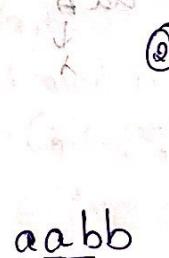
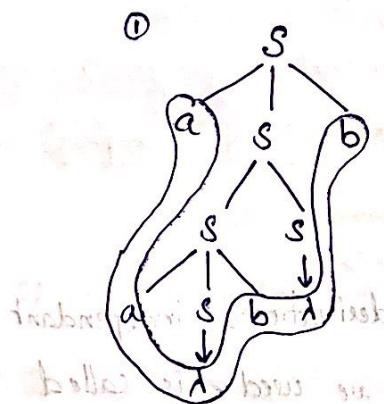
For a string if you can draw 2 diff parse tree  
then such a grammar is called as ambiguous grammar.

def 2 → For a string if more than one LM def or RM def are available then such a grammar is called 'AG'.

Eg: Check whether the given grammar is ambiguous or not

Q: S → asb | ss | x

Sol: aabb

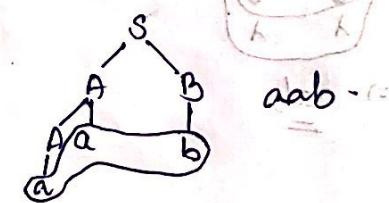
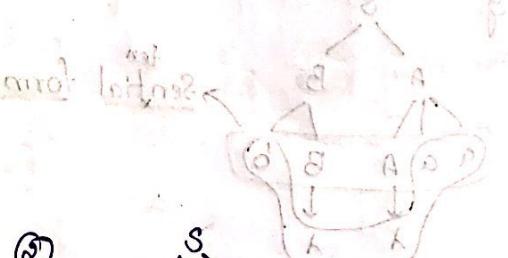
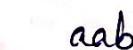
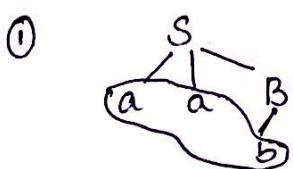


$\therefore$  2 diff parse tree.

$$(20) \quad S \rightarrow AB \mid aaB$$

$$A \rightarrow a | A_a$$

$$B \rightarrow b$$



## \* Simplification of CFG :-

CFG simplification can be done through ~~2~~ <sup>normal</sup> forms

(1) Chomsky Normal form

(2) Greibach Normal form.

To apply the above we have to first simplify CFG by following steps

① Removing  $\lambda$  production

② Removing useless production

③ Removing unit production

If Those 3 are done through a concept called "Usefull Substitution Rule".

① Removing Useless Production:-

Eg: when even a production of grammar that can never take part in any derivation is called as useless production.

$$\text{Eg: } S \rightarrow aSb | \lambda | A$$

$$A \rightarrow aA$$



$$S \rightarrow aSb | \lambda .$$

②  $S \rightarrow abAB | ba$

$$A \rightarrow aaa$$

$$B \rightarrow aA | bb$$

$$\Rightarrow S \rightarrow abAaA | abAbb | ba$$

$$A \rightarrow aaa$$

③ Removing  $\lambda$  Productions:

① It can't be reached from start symbol

② It can't derive a terminal string.

Ex: ①  $S \rightarrow aS \mid AB$

$A \rightarrow bA$

$B \rightarrow AA$

$L = \{\emptyset\}$

②  $S \rightarrow AaB \mid aaB$

$A \rightarrow \lambda$

$B \rightarrow bbA \mid \lambda$

Sol: remove  $A \rightarrow \lambda \rightarrow \textcircled{1}$

③  $S \rightarrow aB \mid aaB$

$B \rightarrow bb \mid \lambda$

④  $S \rightarrow a \mid aa \mid aB \mid aaB$

$B \rightarrow bb$

27/8/19

## \* \* The CYK Algorithm:

Author: J.Cocke D.Younger T.Kasami (CYK); TC:  $O(n^3)$

→ This is called Dynamic Programming (or) table-filling algorithm.

⊗ To apply CYK → first the grammar should be in CNF. If not  
Simplify grammar into CNF.

Eg.: CNF

$$S \rightarrow AB \mid BC$$

$$A \rightarrow BA \mid a$$

$$B \rightarrow CC \mid ab$$

$$C \rightarrow AB \mid a$$

$$w \rightarrow baaba$$

Is baaba in L(G)? prove it using CYK algo?

Sol.:

$\{S, A, C\}$ $x_{15}$				
$\emptyset$ $x_{14}$	$\{S, A, C\}$ $x_{25}$			
$\emptyset$ $x_{13}$	$\{B\}$ $x_{24}$	$\{B\}$ $x_{25}$		
$\{S, A\}$ $x_{12}$	$\{B\}$ $x_{23}$	$\{S, C\}$ $x_{24}$	$\{S, A\}$ $x_{25}$	
$\{B\}$ $x_{11}$	$\{A, C\}$ $x_{22}$	$\{A, C\}$ $x_{33}$	$\{B\}$ $x_{44}$	$\{A, C\}$ $x_{55}$

b a a b a

⊗ (Used to fill table rows no)  
Formula:  $U(x_i, x_{i+1}, j), (x_i, x_{i+1}, x_{i+2}, j) \dots (x_i, x_{j-1}, x_j)$

Now 2<sup>nd</sup> row

$$\Rightarrow x_{12} \rightarrow (\underline{x_{11}}, \underline{x_{22}})$$

$$(\{B\}, \{A, C\})$$

$\Rightarrow \{BA, BC\} \rightarrow$  if any prod is there keep the value else

$\downarrow$   
A

$\downarrow$   
S

$$\textcircled{i} \quad x_{23} \rightarrow (x_{22}, x_{33}), (x_{22}, x_{13}), (x_{22}, x_{11})$$

$$(\{A, C\}, \{A, C\}), (\{A, C\}, \{B, C\}), (\{A, C\}, \{B, C\})$$

$$\Rightarrow \{\overline{AA}, \overline{AC}, \overline{CA}, \overline{CC}\}$$

$$\downarrow \quad \downarrow \quad \downarrow \quad \downarrow$$

$$\emptyset \quad \emptyset \quad \emptyset \quad \{B\}$$

$$\Rightarrow x_{34} \rightarrow (x_{33}, x_{44}), (x_{33}, x_{34}), (x_{33}, x_{24})$$

$$(\{B\}, \{A, C\}), (\{B\}, \{B, C\}), (\{B\}, \{A, C\})$$

$$\{\overline{AB}, \overline{CB}\}$$

$$\downarrow \quad \downarrow$$

$$\emptyset \quad \emptyset$$

$$\Rightarrow x_{45} \rightarrow (x_{44}, x_{55}), (x_{44}, x_{45}), (x_{44}, x_{35})$$

$$(\{B\}, \{A, C\}), (\{BA, BC\}), (\{AC\}, \{AC\}), (\{A, C\}, \{B\})$$

$$(\{A\}, \{B\}), (\{B\}, \{B\}), (\{A\}, \{B\})$$

$$\downarrow \quad \downarrow$$

$$\{A\} \quad \{B\}$$

$$\textcircled{x} \quad x_{13} \rightarrow (x_{11}, x_{23}), (x_{12}, x_{33})$$

$$\text{Ansatz } \boxed{m_1 m_2} (\{B\}, \{B\}), (\{SA\}, \{A, C\})$$

$$\{\overline{BB}, \overline{SA}, \overline{SC}, \overline{AA}, \overline{AC}\}$$

$$\downarrow \quad \downarrow \quad \downarrow \quad \downarrow \quad \downarrow$$

$$\emptyset \quad \emptyset \quad \emptyset \quad \emptyset \quad \emptyset$$

$$x_{24} \rightarrow (x_{22}, x_{34}), (x_{23}, x_{44})$$

$$(\{A, C\}, \{S, C\}), (\{B\}, \{B\})$$

$$\{\overline{AS}, \overline{AC}, \overline{CS}, \overline{CC}, \overline{BB}\}$$

$$\downarrow \quad \downarrow \quad \downarrow \quad \downarrow \quad \downarrow$$

$$\emptyset \quad \emptyset \quad \emptyset \quad \emptyset \quad \emptyset$$

$$x_{35} \rightarrow (x_{33}, x_{45}), (x_{34}, x_{55})$$

$$(\{A, C\}, \{S, A\}), (\{S, C\}, \{A, C\})$$

$$\{\overline{AS}, \overline{AA}, \overline{CS}, \overline{CA}, \overline{SA}, \overline{SC}, \overline{CA}, \overline{CC}\}$$

$$\downarrow \quad \downarrow \quad \downarrow \quad \downarrow \quad \downarrow \quad \downarrow \quad \downarrow \quad \downarrow$$

$$\emptyset \quad \emptyset \quad \emptyset \quad \emptyset \quad \emptyset \quad \emptyset \quad \emptyset \quad B$$

$x_{14} \rightarrow (x_{11}, x_{24}), (x_{12}, x_{34}), (x_{13}, x_{44})$

$(\{B^4, \{B^4\}), (\{S, A^3\}, \{S, C^3\}), (\{\phi^3, \{B^4\})$

$\{BB, SS, SC, AS, AC, \phi\}$

$x_{25} \rightarrow (x_{22}, x_{35}), (x_{23}, x_{45}), (x_{24}, x_{55})$

$(\{A, C^3\}, \{B^3\}), (\{S, A^3\}, \{S, A^3\}), (\{B^3, \{A^3\}\})$

$\{AB, CB, BS, BA, BA, BC\}$

$\{S, C\}$

$\rightarrow x_{15} \rightarrow (x_{11}, x_{25}), (x_{12}, x_{35}), (x_{13}, x_{45}), (x_{14}, x_{55})$

$(\{B^3, \{A^3\}\}, \{B^3, \{A^3\}\}), (\{\phi^3, \{S, A^3\}\}), (\{\phi^3, \{A, C^3\}\})$

$\{BS, BA, BC, SB, AB\}$

$\phi A S \emptyset S, C$

☹ If the starting symbol ( $S$ ) should be there in  $\boxed{x_{15}}$  then  
 the given language will be there in the given grammar.

28/8/19

\* Push down Automata:-

↓  
CFG

$L = \{a^n b^n ; n \geq 0\}$

aaabbbb



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

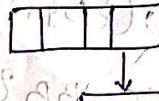
$\delta : Q \times (\Sigma \cup \{\lambda\}) \times \Gamma \rightarrow Z \in \Gamma$

$\Rightarrow \delta : Q \times \Gamma^* \rightarrow Z$

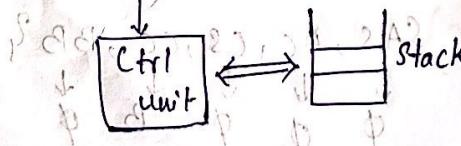
( $p^R, \{p^R\}), (p^R, \{p^R\}) \leftarrow p^R$

input file

$\{p^R, \{p^R\}, \{p^R, \{p^R\}\}, \{p^R, \{p^R, \{p^R\}\}\}\}$



Ctrl unit



Stack

( $a^R, \{a^R\}), (a^R, \{a^R\}) \leftarrow a^R$

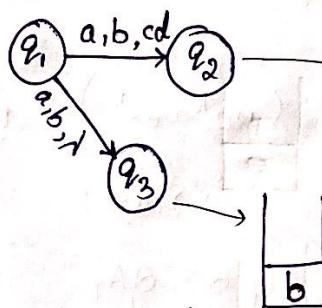
$(a^R, \{a^R\}), (a^R, \{a^R\}) \leftarrow a^R$

Q. Suppose the set of transition rules of an NPDA contains

$$\delta(q_1, a, b) = \{(q_2, cd), (q_3, \lambda)\}$$

↓  
(Current State)  
↓  
(Input symbol)  
↓  
(top of the stack)

Sol:-



$$0.5 : \{d, \lambda\} = \Sigma : \{a\}$$



(E1, Q1)  $\rightarrow$  (E2, Q2)

Deterministic

For same input we have diff transition so, it is not possible in PDA.

Q. Consider an NPDA

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

$$\Sigma = \{a, b\} \rightarrow \text{IP}$$

$$\Gamma = \{0, 1\} \rightarrow \text{Stack}$$

$$Z = 0$$

$$F = \{q_3\}$$

→ initial state is  $q_0$

Transitions:  $(q_0, a, 0) = (q_1, 10)$

$$\delta(q_0, a, 0) = \{(q_1, 10), (q_3, \lambda)\}$$

$$\delta(q_0, \lambda, 0) = \{(q_3, \lambda)\}$$

$$\delta(q_1, a, 1) = \{(q_2, 11)\}$$

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

$$\delta(q_2, b, 1) = \{(q_2, \lambda)\}$$

$$\delta(q_2, \lambda, 0) = \{(q_3, \lambda)\}$$

Sol:-

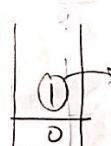
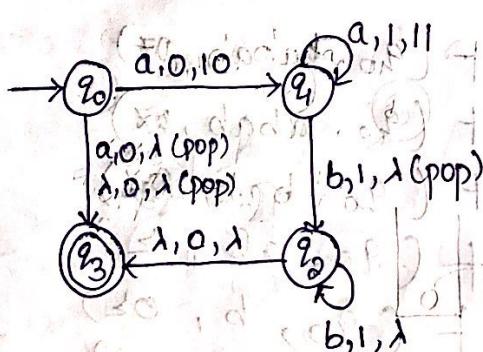


abb → won't be generated

aabb → can be generated

30/8/19

Q. Construct an NPDFA for language  $L = \{w \in \{a,b\}^*: n_a(w) = n_b(w)\}$

$$L = \{aabbb, abab, bbaa, babaa\}$$

Sol:  $\Gamma = \{0,1\} : \Sigma = \{a,b\} ; z = 0$

$$\delta(q_0, a, z) = (q_0, 0z)$$

$$\delta(q_0, b, z) = (q_0, 1z)$$

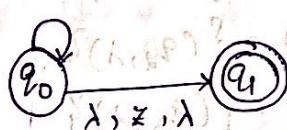
$$\delta(q_0, a, 0) = (q_0, 00)$$

$$\delta(q_0, b, 1) = (q_0, 11)$$

$$\delta(q_0, a, 1) = (q_0, \lambda)$$

$$\delta(q_0, b, 0) = (q_0, \lambda)$$

$$\delta(q_0, \lambda, z) = (q_1, \lambda)$$



tracing:- ababab  
 $(q_0, ababab, z) \xrightarrow{\quad} (q_0, babab, 0z)$   
 $\xrightarrow{\quad} (q_0, abab, 0z)$   
 $\xrightarrow{\quad} (q_0, bab, 0z)$   
 $\xrightarrow{\quad} (q_0, ab, 1z)$   
 $\xrightarrow{\quad} (q_0, b, 0z)$   
 $\xrightarrow{\quad} (q_0, \lambda, z)$   
 $\xrightarrow{\quad} (q_f, \lambda, z)$

Q.  $L = \{w \in \{a,b\}^*: w \in \{a,b\}^*\}$

Processing the string back, the

Sol: baacaab

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

$$\delta(q_0, b, b) = (q_0, 1z)$$

$$\delta(q_0, a, 0) = (q_1, 00)$$

$$\delta(q_0, b, 1) = (q_0, \lambda)$$

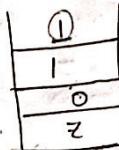
$$\delta(q_0, a, 1) = (q_0, 00)$$

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

$$\delta(q_0, c, 0) = (q_0, \lambda)$$

$$\delta(q_0, c, 1) = (q_0, \lambda)$$

$$\delta(q_0, c, \lambda) = (q_f, \lambda)$$



$\overline{abb} \overline{Cbb} \overline{a}$

3/8/89

Problems:-

① NPDA for  $L = \{a^n b^{2n} \mid n \geq 1\}$

② NPDA for  $L = \{a^n b^m c^{m+n}, m \geq 0 \mid n \geq 0\}$

③ NPDA for  $L = \{a^n b^m : n \leq m \leq 3n\}$

④ " "  $L = \{a^3 b^n c^n : n \geq 0\}$

Sol:-

$$L = \{a^n b^{2n} \mid n \geq 1\}$$

$$\delta(q_0, a, z) = (q_1, 0z)$$

$$\delta(q_0, a, 0) = (q_0, 00)$$

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

$$\times [\delta(q_1, b, 1) = (q_1, \lambda)] \times$$

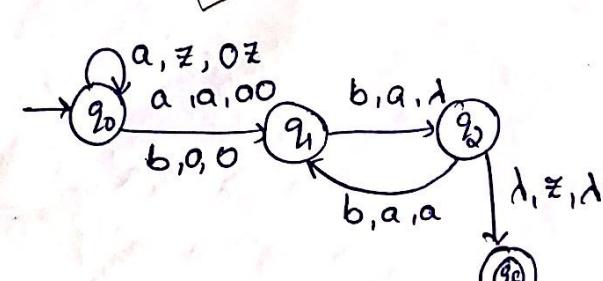
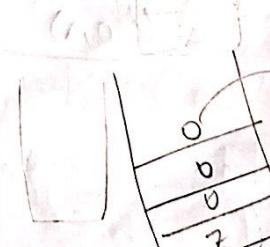
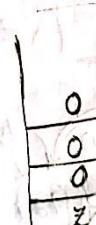
$$\delta(q_1, \lambda, 0) = (q_f, \lambda)$$

$$\delta(q_0, \lambda, 0) = (q_0, 00)$$

$$\delta(q_1, b, 0) = (q_2, \lambda)$$

$$\delta(q_2, b, a) = (q_1, 0)$$

$$\delta(q_2, \lambda, z) = (q_f, \lambda)$$



Ques tracing:-

$$aabbbb = \vdash (q_0, (aabbbb, z))$$

$$\vdash (q_0, abbbb, 0z)$$

$$\vdash (q_0, bbbb, 00z)$$

$$\vdash (q_1, bbb, 00z)$$

$$\vdash (q_2, bb, 0z)$$

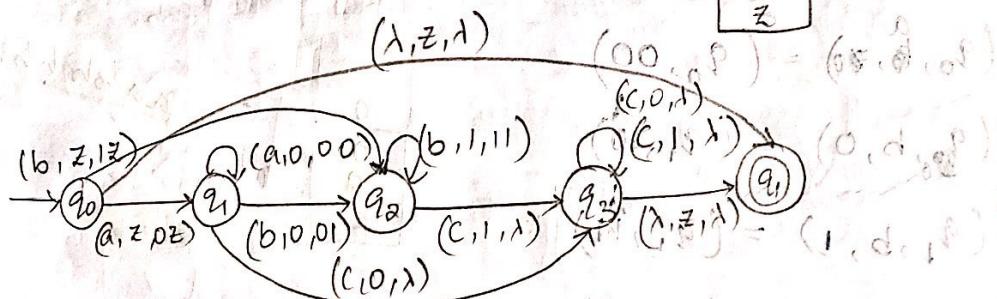
$$\vdash (q_1, b, 0z)$$

$$\vdash (q_2, \lambda, z)$$

$$\vdash (q_f, \lambda, z)$$

(Q)  $L = \{a^n b^m c^{m+n} \mid n, m \geq 0\}$

$m, n = 0 \rightarrow \{\lambda, abcc, aabccc \dots\}$



Ans  $L = \{a^n b^m : n \leq m \leq 3n\} \rightarrow NPDA$

$$\delta(q_0, a, z) = \{(q_1, 1z), (q_1, 11z), (q_1, 111z)\}$$

$$\delta(q_1, a, 1) = \{(q_1, 1z), (q_1, 11z), (q_1, 111z)\}$$

$$\delta(q_1, b, 1) = \{(q_1, 1z)\} \xrightarrow{\text{previous content}}$$

$$\delta(q_1, d, z) = \{(q_f, z)\}$$

\* Pushing more than one to stack.

$$L = \{a^n b^n : n \geq 0\}$$

$$\delta(q_0, \lambda, z) \rightarrow \{(q_f, z)\}$$

$$\delta(q_0, a, z) = \{(q_1, 1z)\}$$

$$\delta(q_1, a, 1) = \{(q_1, 11z)\}$$

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

$$\delta(q_2, a, z) = \{(q_f, z)\}$$

$$\delta(q_2, b, 1) = \{(q_2, 1)\}$$

aabb

1
1
z.

N.V. V. Purushotham  
NPDA:- (Def)

①  $\delta(q_0, a, 0)$  } can only happen in NPDA  
 $\delta(q_0, \lambda, 0)$  }

② same input & multiple transitions can happen.

- Ⓐ  $(q_0, a^3 b^2 c^2, z) \vdash (q_1, a^2 b^2 c^2, z)$   
 $\vdash (q_2, a b^2 c^2, z)$   
 $\vdash (q_3, b^2 c^2, z)$   
 $\vdash (q_4, b c^2, 0z)$   
 $\vdash (q_{41}, c^2, 00z)$   
 $\vdash (q_{45}, c10z)$   
 $\vdash (q_5, \lambda, z)$

## \* Left Recursive Grammae:

A production of grammar is said to have left recursion if the left most variable (of its RHS) is same as variable of LHS.

A Grammar containing a production having left recursion is called as Left Recursive Grammae.

### \* Eliminating Left Recursion:

$$A \rightarrow A\alpha / \beta$$

$$\Downarrow$$

$$A \rightarrow \beta A'$$

$$A' \rightarrow \alpha A' / \lambda$$

Ex:  $S \rightarrow A$

$$A \rightarrow Aa / Ac / Ab / aB / aCc$$

$$B \rightarrow bBc / f$$

Sol:

$$S \rightarrow A$$

$$A \rightarrow aBA' / acA'$$

$$A' \rightarrow dA' / eA' / \lambda$$

$$B \rightarrow bBc / f$$

⑨

$$A \rightarrow Ba / Aa / c$$

$$B \rightarrow Bb / \boxed{Ab} / d$$

↑ Substitute

Sol:

$$A \rightarrow \boxed{BaA' / CA'}$$

$$A' \rightarrow aA' / \lambda$$

$$B \rightarrow Bb / \boxed{BaA'b / CA'b / d}$$

$$\Rightarrow A \rightarrow BaA' / CA'$$

$$A' \rightarrow aA' / \lambda$$

$$B \rightarrow CA'bB' / dB'$$

$$B' \rightarrow bB' / aA'bB' / \lambda$$

## $\rightarrow$ PDA for CFG

The Grammar is  $S \rightarrow AA/a$   
 $A \rightarrow SA/b$

$S \rightarrow AA/a/SAA/bA$   
 $A \rightarrow AAA/aA/b$

$S \rightarrow AA/a$

$A \rightarrow aAA'/bA'$

$A' \rightarrow AAA'/f$

$$\{(\epsilon, \epsilon)\}^* = (\epsilon, \epsilon)^* \quad \textcircled{1}$$

$$\{(\epsilon, \epsilon)\}, \{(\epsilon, \epsilon)\}^* = (\epsilon, \epsilon)^* \quad \textcircled{2}$$

$$\{(\epsilon, \epsilon)\} = (\epsilon, \epsilon)^* \quad \textcircled{3}$$

$$\{(\epsilon, \epsilon)\} = (\epsilon, \epsilon)^* \quad \textcircled{4}$$

$$\{(\epsilon, \epsilon)\} = (\epsilon, \epsilon)^* \quad \textcircled{5}$$

$$(\epsilon, \epsilon, \epsilon) \vdash (\epsilon, \epsilon, \epsilon) \quad \textcircled{6}$$

$$(\epsilon A, \epsilon, \epsilon) \vdash$$

$$(\epsilon A, \epsilon, \epsilon) \vdash$$

$$(\epsilon B, \epsilon, \epsilon) \vdash$$

$$(\epsilon, \epsilon, \epsilon) \vdash$$

6/9/19

Page No. 40

~~\* Steps for designing PDA for CFL:-~~

Step 1:- Put Start symbol on the stack.

Step 2: To stimulate application of a production  $A \rightarrow a\alpha$ , the variable 'A' on the top of the stack and 'a' is the input symbol.

Step 3:- The variable on the top of the stack is removed and replaced by the variable string  $\alpha$ .

Eg:- Construct a PDA that accepts the language generated by grammar with productions.

$$S \rightarrow aSbb/a$$

↓  
Transform into GNF

$$S \rightarrow aSA/a$$

$$A \rightarrow bB$$

$$B \rightarrow bB$$

$$B \rightarrow b$$

$$\textcircled{1} \quad \delta(q_0, \lambda, z) = \{(q_1, S, z)\}$$

$$\delta(q_1, a, S) = \{(q_1, SA)\}, \{(q_1, \lambda)\}$$

$$\delta(q_1, b, A) = \{(q_1, B)\}$$

$$\delta(q_1, b, B) = \{(q_1, \lambda)\}$$

$$\delta(q_1, \lambda, z) = \{(q_2, \lambda)\}$$

~~Input  $\Rightarrow aabb \rightarrow (q_0, aabb, z)$~~

$$f(q_1, aabb, Sz)$$

$$f(q_1, abb, SAz)$$

$$f(q_1, bb, Az)$$

$$f(q_1, b, Bz)$$

$$f(q_1, \lambda, z) \xrightarrow{} f(q_2, \lambda) \checkmark$$

$\textcircled{Q} \quad S \rightarrow aA,$   
 $A \rightarrow aABC/bB/a$   
 $B \rightarrow b$   
 $C \rightarrow c.$

$$\underline{\text{Sol:}} \quad \delta(q_0, \lambda, z) = \{q_1, Sz\}$$

$$\delta(q_1, a, S) = \{q_1, A\}$$

$$\delta(q_1, a, A) = \{q_1, ABC\}, \{q_1, \lambda\}$$

$$\delta(q_1, b, A) = \{q_1, B\}$$

$$\delta(q_1, b, B) = \{q_1, \lambda\}$$

$$\delta(q_1, c, C) = \{q_1, \lambda\}$$

$$\delta(q_1, \lambda, z) = \{q_f, \lambda\}$$

$$aaabc \Rightarrow (q_0, aaabc, z) \xrightarrow{(q_1, aabbc, Sz)} F(q_1, abbc, Az)$$

first part starts with  $F(q_1, abbc, Az)$  and  $\textcircled{1}$

next part for  $i$  is  $F(q_1, bc, Bz)$  and  $\textcircled{2}$

last part of  $F(q_1, c, Cz)$  is  $(q_f, \lambda)$   $\textcircled{3}$

equation of output and  $F(q_1, \lambda, z)$  is  $\textcircled{4}$

$\text{std } F(q_f, \lambda)$  is final state

$\text{Q. } S \rightarrow AA/a$  for  $f(S, a) = \{q_f\}$  and  $\text{std } \textcircled{1}$

$$A \rightarrow SA/b$$

Sol:

$$\delta(q_0, \lambda, z) / \text{std}$$

$$S \rightarrow SAA/a/b/A$$

$$A \rightarrow SA/b$$

$$S \rightarrow as^1/bas'$$

$$S^1 \rightarrow AAS^1/\lambda$$

$$A \rightarrow SA/b$$

standard of



$$\{q_1, \lambda\} = \{q_f, \lambda\}$$

$$\{q_1, \lambda\} = \{q_f, \lambda\}$$

$$S \rightarrow aS' / bAS'$$

$$S' \rightarrow AAS' / \lambda$$

$$A \rightarrow aS'A / bAS'A / b$$

$$S \rightarrow aS' / bAS'$$

$$S' \rightarrow aS'AAS' / bAS'AAS' / bAS' / \epsilon$$

$$A \rightarrow bAS'A / aS'A / b$$

13/9/19

\* Deterministic Push down Automata:-

It is said to be deterministic if it satisfies

the following 8 conditions

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

① The transition of  $\delta(q, a, b)$  contains atmost 1 element

② If transition of  $\delta(q, \lambda, b)$  is not empty then

$\delta(q, c, b)$  must be empty for every  $c \in \Sigma$

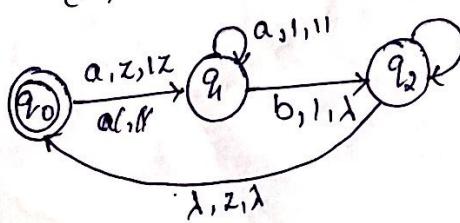
→ when a  $\lambda$  move is possible for some configuration, no input consuming alternative is available.

Problem

① The language  $L = \{a^n b^n : n \geq 0\}$  ... construct PDA

Sol:-

$$\{\lambda, ab, aabb\}$$



$$\delta(q_0, a, z) = \{(q_1, 1z)\}$$

$$\delta(q_0, a, \lambda) = \{(q_1, 11)\}$$

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

$$\delta(q_2, b, 1) = \{(q_3, \lambda)\}$$

$$\delta(q_3, \lambda, z) = \{(q_{\text{f}}, \lambda)\}$$

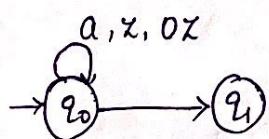
Q1)  $L = \{a^n b^{2n}; n \geq 0\}$

Q2)  $L = \{a^n b^m; m > n+2\}$

\* Q3)  $L = \{a^n b^n; n \geq 1\} \cup \{b\}$  Is it deterministic?

Solution:-

Ans:-  $L = \{a^n b^{2n}; n \geq 0\}$



$$\delta(q_0, a, \bullet z) = \{(q_0, 0z)\}$$

$$\delta(q_0, \lambda, z) = \{(q_f, \lambda)\}$$

\* Properties of CFL:-

- ① Union of 2 CFL is CFL.
- ② CFL are closed under (\*) closure.
- ③ Intersection of 2 CFL is CFL
- ④ Concatenation need not be CFL.

18/9/19

Q. Construct a PDA for language  $L = \{a^n b^n c^n, n \geq 1\}$

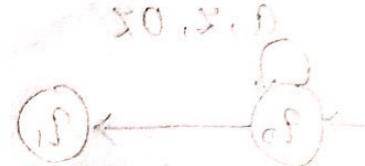
Sol:-

$$\delta(q_0, a, z) = (q_0, a1z)$$

$$\delta(q_0, a, 1) = (q_1, 1)$$

Q. Consider the language  $L_1 = \{a^n b^n c^m\} ; L_2 = \{a^n b^m c^m\} ; n, m \geq 0$ .

$$\{a^n b^m c^m\} = 1$$



$$\{a^n b^m c^m\} = 1$$

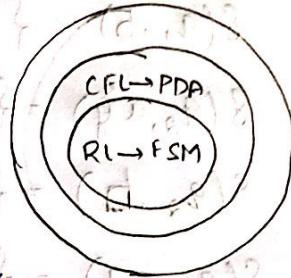
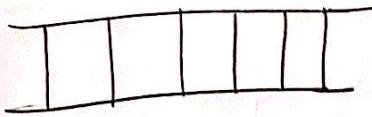
$$\{a^n b^n c^m\} = 1$$

adalgia v.v. Emp

\* \* Turing Machine :-

It is another type of Automata.

tape →



→ Turing Machine. It is an mechanical, or algorithmic computation.

→ It is an automaton. whose storage is a tape which is associated with read write head that can travel b/w right or left on the tape. and it can read or write a single symbol on each move.

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

The transitions will be  $\delta: Q \times \Gamma \rightarrow Q \times \Gamma \times \{L, R\}$ .

e.g.:  $\frac{ab|c}{(a)} \rightarrow \frac{d|bc}{(b)}$

$$\delta(q_0, a) \rightarrow (q_1, d, R)$$

③ Design a turing Machine for the given language  $\{00^*\}^*$ .

Sol.:  $\delta(q_0, 0) \rightarrow (q_1, 0, R)$

$$\delta(q_1, \square) \rightarrow (q_1, \square, R)$$

$$\delta(q_1, \square) \rightarrow (q_2, \square, R)$$

$$q_2 \in F$$

$$\frac{q_0 \downarrow q_1 \downarrow q_2 \downarrow}{0 \mid 0 \mid 0 \mid 0}$$

Q18 Design a turing Machine  $L = \{aba^*b\}$

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

$$\delta(q_1, b) = \{q_2, b, R\}$$

$$\delta(q_2, a) = \{q_2, a, R\}$$

$$\delta(q_2, b) = \{q_3, b, R\}$$

$$\delta(q_3, \square) = \{q_4, \square, R\}$$

Q19  $L = \{w : |w| \text{ is multiple of } 3\}$

$\Rightarrow$  It will take off 3 bits at a time and then move to next cell

Q20  $L = \{w : |w| \text{ is even}\}$

$\Rightarrow$  It will take off 2 bits at a time and then move to next cell

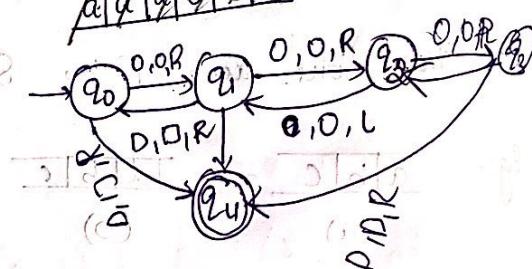
Q21  $L = \{a^n b^n : n \geq 1\}$



3.6.9(1a, E)

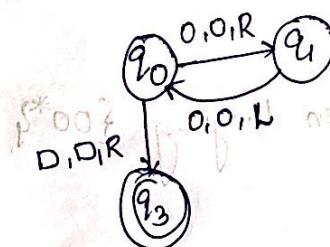
$$\delta(q_0, a) = \{q_1, 0, R\}$$

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



$(q_0, a, \square) \leftarrow (q_1, 0, \square)$

Any



$$\delta(q_0, a) = \{q_1, 0, R\}$$

$$\delta(q_1, 0) = \{q_3, 0, L\}$$

$$\delta(q_3, a) = \{q_0, 0, R\}$$

$(q_0, a, \square) \leftarrow (q_1, 0, \square)$

$(q_1, 0, \square) \leftarrow (q_3, 0, \square)$

$(q_3, a, \square) \leftarrow (q_0, 0, \square)$

$[q_0, q_1, q_3]$

$\square \square \square$