AUTOMATA THEORY & COMPUTABILITY

31/7/18

Module 1: Why study the theory of computation. languages and strings.

Module 2: Regular expressions.

Module 3: contextfree Grammars (CFG)

Module 4: Contextfree and Non-contextfree languages.

Module 5: Variants of Turing Machines (TM)

TextBooks:

1) Elaine Rich, Automata, Computability and complexity, 1st Edition, Pearson Education 2012/2013.

2) KLP Mishra, N Chandrashekaran, 3rd Edition, the of computer science, PHI, 2012.

Module 1:

*Concepts of Automata Theory :-

Alphabet: > A language consists of various symbols from which the words, statements etc... can be obtained.
These symbols are called alphabets.

> Formally, an alphabet is defined as a finite non-empty set of symbols.
The symbol Σ denotes the set of alphabets of language.

01/8/18

Eg(1): The alphabets of C language has the letters from A to Z, a to z, digits from 0 to 9, symbols such as +, -, *, /, (,), {, } etc... and is denoted by

$$\Sigma = \{A, B, \ldots Z, a, b, \ldots z, \ldots 0, 1, \ldots 9, +, -, *, /, (,), \ldots\}$$

Eg(2): The machine language is made up of only 0's and 1's. The alphabets of machine language is denoted by

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

String :> The sequence of symbols applied from the alphabets of a language is called a string.

> Formally a string is defined as a finite sequence of symbols from the alphabet Σ.

Note: An empty string is denoted by the symbol ε (or) λ.
And note that ε doesnot belongs to Σ.
$[\varepsilon \notin \Sigma]$

Eg(1): Let $\Sigma = \{0, 1\}$ is set of alphabets. The various strings that can be obtained from Σ are,

$$\Sigma = \{\varepsilon, 0, 1, 00, 01, 10, 11, 000, 001, \dots\}$$

Note: Notethat an infinite no. of strings can be generated from Σ and once the string is generated it has finite no. of symbols init.

Concatenation of 2 strings:-

The concatenation of 2 strings u and v is a string obtained by writing the letters of string u followed by the letters of string v. i.e, appending symbols of v to the right of u.

If $u = a_1, a_2, \dots a_n$
$v = b_1, b_2, \dots b_m$

then the concatenation of u and v is denoted,

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

Eg(1): Let 2 strings u and v be u = computer
v = science

the concatenation of u and v is denoted by,

uv = computerscience

Note: concatenation is not commutative.

$uv \neq vu$

$\therefore$ computerscience $\neq$ sciencecomputer
[$\because$ pattern differs].

Substring:- 8/8/18

Let w is the string obtained from symbols in Σ. The string w if it can be decomposed into 3 substring x, y, z such that

$W = x\,y\,z$

then x is the substring
y is the substring
z is the substring etc.. of the string w.

Prefix:- A prefix is a string of any no. of leading symbols.

eg(1) Let w is the string and let $w = xyz$, the string w has prefix ε, x, xy, xyz.

(2) In the string "Rama" the various prefixes are. ε, R, Ram, Rama.

Suffix:- Suffix is a string of any no. of trailing symbols.

eg(1) If $w = xyz$, then string w has suffix ε, z, yz, xyz.

2) In the string "Rama" the strings ε, a, ma, ama, Rama are the suffixes.

Reversal of a string:- The reversal of a string is obtained by writing symbols in reversal order.

ie, if $u = a_1 a_2 a_3 \ldots a_n$ then the reverse of u is denoted by, u^R & is given by, $\boxed{u^R = a_n \ldots a_3 a_2 a_1}$

So if u is an emptystring denoted by ε and u has only one symbol then,

$\varepsilon^R = \varepsilon$

$a^R = a$

* The reverse of a string can be defined recursively as follows,

if "a" is a symbol and "w" is a string derived from the alphabet Σ then the reverse of a string can be defined as,

$$w^R = \begin{cases} \varepsilon, & \text{if } w = \varepsilon \\ a, & \text{if } w = a \\ (xa)^R = ax^R, & \text{if } w = xa \end{cases}$$

[∵ x is string
a is symbol]

for each $a \in \Sigma$, $w, x \in \Sigma^*$

Note: x^R in the definition indicates the reverse string of string x.

Length of a string:-

The length of a string u is the no. of symbols in u and is denoted by $|u|$.

ie, if $u = a_1 a_2 a_3 \ldots a_n$ then the length of u is given by,

$|u| = n$.

The length of an emptystring ε is zero & is denoted by

$|\varepsilon| = 0$

Note: $\varepsilon w = w\varepsilon = w$

[∵ ε doesnot occupy any space].

Power of an alphabet :— 9/8/18

The power of an alphabet is denoted by, Σ^i is the set of words of length i.

Variation of power of an Alphabet —
- → Kleen closure (or) Kleen star (Σ^*)
- → Kleen plus (Σ^+)

$\Sigma^* = \Sigma^+ + \varepsilon$

$\Sigma^+ = \Sigma^* - \varepsilon$

Kleen closure :— The Kleen closure is defined as follows,

$$\Sigma^* = \Sigma^0 \cup \Sigma^1 \cup \Sigma^2 \cup \dots$$

$$= \{\varepsilon\} \cup \{0,1\} \cup \{00, 01, 10, 11\} \cup \dots$$

$$\{0,1\}^* = \Sigma^* = \{\varepsilon, 0, 1, 00, 01, 10, 11, \dots\} \quad [\because \Sigma = \{0,1\}]$$

which is a set of words of any lengths. [i.e, null string]. Each string is made up of symbols only from Σ.

eg(1) Kleen star can be applied to set of strings $\Sigma = \{a, bc\}$. The set of strings consisting of a's & bc's of any length can be obtained as shown below,

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

$$\Sigma^* = \Sigma^0 \cup \Sigma^1 \cup \Sigma^2 \cup \dots$$

$$\Sigma^* = \{\varepsilon\} \cup \{a, bc\} \cup \{abc, bca, aa, bcbc\} \cup \dots$$

$$\{a, bc\}^* = \Sigma^* = \{\varepsilon, a, bc, abc, aa, bca, bcbc\}$$

[$\because$ bc is a pattern & treated as string with 1 length]

- Klein plus:— It is a variation of Klein star operator. It is denoted by Σ^+ is defined as follows,

$$\Sigma^+ = \Sigma^1 \cup \Sigma^2 \cup \Sigma^3 \cup \ldots$$

which is a set of words of any length except null string i.e, ε is not part of Σ^+ and hence ε doesnot belongs to Σ^+ [$\varepsilon \notin \Sigma^+$].

eg(1) Let $\Sigma = \{0, 1\}$

The Σ^+ is shown below,

$$\Sigma^+ = \Sigma^1 \cup \Sigma^2 \cup \Sigma^3 \cup \ldots$$

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

$\{0,1\}^+ = \Sigma^+ = \{0, 1, 00, 10, 01, 11, \ldots\}$

is a set of strings of 0's and 1's of any length except null string.

[$*$ → zero (or) more
$+$ → one (or) more]

Note: $\Sigma^* = \Sigma^+ + \varepsilon$

$\Sigma^+ = \Sigma^* - \varepsilon$

- Recursive Definition for length of the string :-

If a is the symbol and w is the string derived from alphabet Σ then the length of the string can be defined as,

$$|w| = \begin{cases} 0 & \text{if } w = \varepsilon \\ 1 & \text{if } w = a \\ |ua| = |u| + 1 & \text{otherwise } w = ua \end{cases}$$

For each $a \in \Sigma$, and $u, w \in \Sigma^*$ [i.e, a is the symbol in Σ, u & w are strings in Σ^*]

Language:- A language can be defined as set of strings obtained from Σ^* where, Σ is the alphabet set for a particular language.

In other words,

A language is a subset of Σ^* which is denoted by,

$L \subseteq \Sigma^*$.

eg(1): A language of strings consisting of equal no. of 0's and 1's can be represented as,

$$L = \{ \varepsilon, 01, 10, 0011, 1100, 0110, 1001, 1010, \ldots \}$$

eg(2): A language of strings consisting of 'n' no of 0's followed by 'n' no. of 1's can be represented using set as shown below,

$$L = \{ \varepsilon, 01, 0011, 000111, \ldots \}$$

eg(3): A language containing empty string ε is defined by,

$L = \{ \varepsilon \}$ — is not empty language

eg(4): An empty language is denoted by,

$L = \{ \}$ (or) ϕ

Sentence:- A string that belongs to language is called word (or) sentence of that language.

eg: a language of strings consistings of equal no. of 0's + 1's can be represented as,

$$L = \{ \varepsilon, 01, 0011, 000111, \ldots \}$$

each word separated by comma is sentence (or) word.

$\therefore$ ε is a sentence, 01 is a sentence etc.

10/8/18

- **Different types of languages:-**

The languages are classified as shown below,

types of languages
- → type 0 language (phrase structured lang.)
- → type 1 language (context sensitive lang.)
- → type 2 language (context free lang)
- → type 3 language (Regular lang)

- **Regular Languages:-**

The regular lang. are defined as languages accepted by DFA's (FSM → Finite state machine; ↳ Deterministic Finite Auto...), NFA's (↳ Nondeterministic FA), ε-NFA's (↳ NFA with ...)

- **Abstract Machines:** It is a conceptual (or) theoritical model of computer hardware (OR) software system which really doesnot exists.

These machines are not actual machines & hence they are also called hypothetical computers. These machines have commonly encounted hardware features & concepts & avoids most of the details that are often found in real machines.

The various types of abstract computers/machines are:-

1) Finite automata
2) Linear bounded automata
3) Pushdown automata
4) Turing machines.

14/8/18

1) Finite Automata (FA):-

A finite automata is a mathematical model which is used to study abstract machines (or) abstract computing devices with the inputs chosen from Σ.

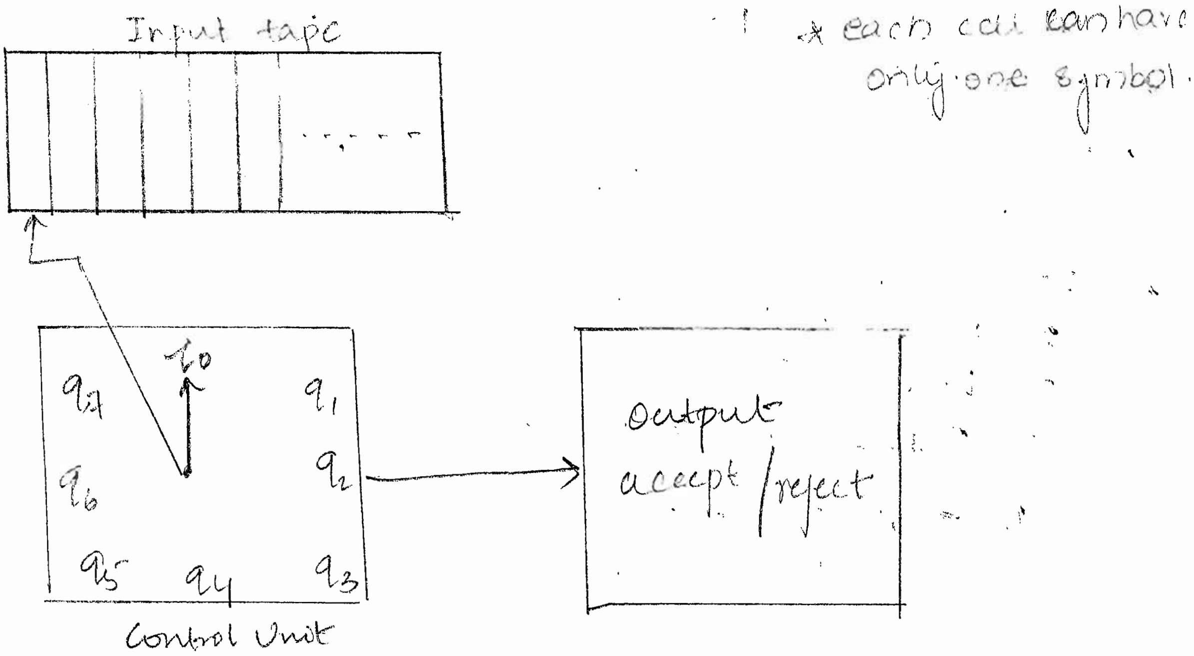
Here Σ stands for set of alphabets using which any string can be obtained. On reading the string the machine may accept the string (or) reject the string.

Using this abstract model the behaviour of the actual system can be understood & build to perform various activities.

Finite automata is an abstract model of digital computer which has 3 components as shown below:-

- Input tape
- Control unit
- Output unit

The pictorial representation (block diagram) of FA is shown below,



• Input Tape:- The input tape is divided into cells, each cell which can hold one symbol.
The string to be processed on these cells.

• Control Unit:- The machine has some states one of which is the start state, designated as q_0 and atleast one final states.
Apart from these it has some finite states designated as $q_1, q_2, q_3 \ldots$. Based on the current input symbol the state of the machine can change.

• Output Unit:- Output may be accepted/rejected when the end of the input is encountered, the control unit may be in accept/reject state.

Working:- The Finite automata works as shown,

1) The machine is assumed to be in start state q_0.
2) The input pointer points to the 1st cell of the tape, pointing to the string to be processed.
3) After scanning the current input symbol the machine can enter into any of the states $q_0, q_1, q_2 \ldots$ and the input pointer points to next character by moving one cell towards right.
4) When the end of the string is encountered the string is accepted iff the automata will be in one of the final states otherwise string is rejected.

eg: Consider an electric switch which has only 2 states OFF & ON. To start with, the switch will be in OFF state. When we push button it goes to ON state. If you push again it goes to OFF state. This can be represented as shown below,

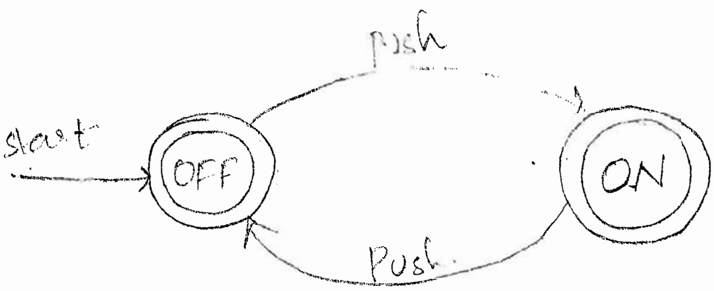


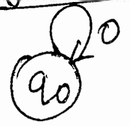
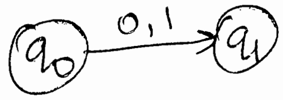
fig: state Transition diagram

Note: * Circles represents the state and labels associated with arc's represents the input given to go to another state.

* The arrow with label start (not originating from any state) is considered as the start state.

* Symbols used in FA :-

Symbol	Meaning
	→ A circle used to represent a state. Here q_0 is a state of the machine.
	→ A circle with an arrow is not originating from any node represents the start state of the machine.
	→ Two circles are used to represent a final state. Here q_0 is the final state.
	→ An arrow with label 1 goes from state q_0 to state q_1. This indicates there is a transition from state q_0 on input symbol 1 to state q_1. This is represented as, $\delta(q_0, 1) = q_1$.

Symbol	Meaning
	→ An arrow with label 0 starts & ends in q_0. This indicates the machine is in state q_0 on reading a zero ~~it~~ remains on state q_0. This is represented as, $\delta(q_0, 0) = q_0$
	→ An arrow with label 0,1 goes from state q_0 to state q_1. This indicates that the machine in state q_0 on reading 0 (or) 1 enters into state q_1. This is represented as, $\delta(q_0, 0) = q_1$ $\delta(q_0, 1) = q_1$

types of FA (or) FSM
- → DFA (or) DFSM
- → NFA (or) NDFA, NFSM (or) NDFSM
- → ε-NFA (or) ε-NDFA, ε-NFSM (or) ε-NDFSM

a) <u>DFA</u> :- The deterministic FA (DFA) is 5-tuple (or) quin tuple indicating 5 components,

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

where,

M → name of the machine. It can also be called by any name.

Q → is non-empty, finite set of states.

Σ → is non-empty, finite set of input alphabets

δ: $Q \times \Sigma$ to Q (i.e., $Q \times \Sigma \rightarrow Q$) is a transition function which is a mapping from $Q \times \Sigma$ to Q

Based on current state & I/P symbol the machine enters into another state. (2)

$q_0 \in Q$ is the start state.

F is the subset of Q ie, $F \subseteq Q$: is set of final (or) accepting states.

(3)

16/8/18

- The language accepted by DFA is,

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

- The non-acceptance of the string w by DFA can be defined as follows, (4)

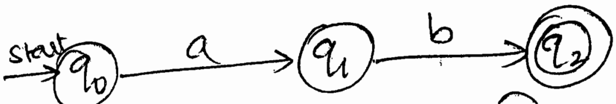
$$\overline{L(M)} = \{ w \mid w \in \Sigma^* \text{ and } \delta^*(q_0, w) \notin F \}$$

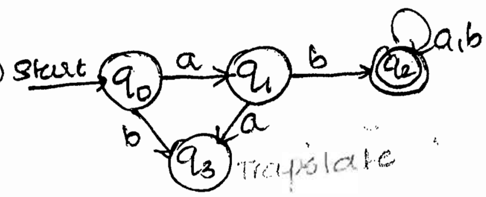
Examples to create/construct DFA:-

(1) Obtain a DFA to accept strings of A's and B's starting the string AB. (5)

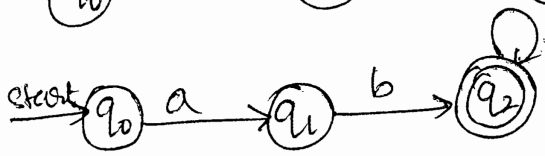
① $\Sigma = \{a, b\}$

② ab

③ 

(or) ③ 

(6)

④ 

δ	a	b
$\rightarrow q_0$	q_1	—
q_1	—	q_2
$*q_2$	q_2	q_2

δ	a	b
$\rightarrow q_0$	q_1	q_3
q_1	q_3	q_2
q_2	q_2	q_2
q_3	q_3	q_3

To accept string: aba

$q_0 \xrightarrow{a} q_1 \xrightarrow{b} q_2 \xrightarrow{a} q_2$

To reject string: aab

$q_0 \xrightarrow{a} q_1 \xrightarrow{a}$ struct

(or) $q_0 \xrightarrow{a} q_1 \xrightarrow{a} q_3 \xrightarrow{b} q_3$

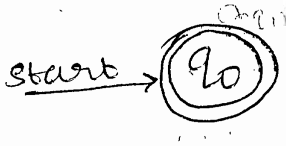
18/8/18

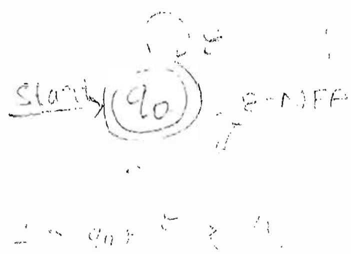
(2) DFA to accept empty language.

↳ $L = \phi$ (or) $\{ \}$

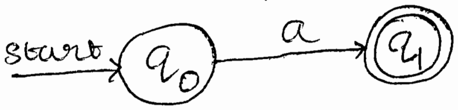
(3) DFA to accept empty string.

↳ $L = \{ \varepsilon \}$

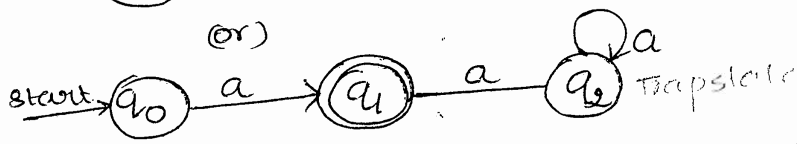




(4) DFA to accept exactly one 'a'. $\Sigma = \{a\}$

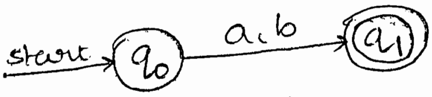


(or)

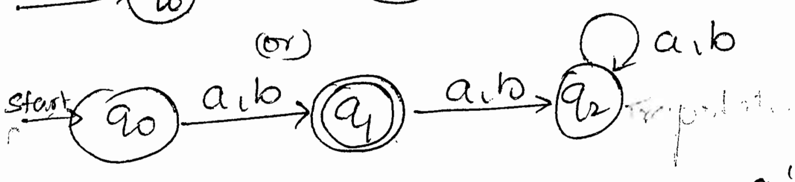


(5) DFA to accept one 'a' (or) one 'b'. $\Sigma = \{a, b\}$

a (or) b



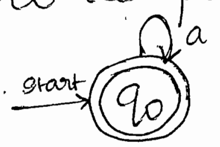
(or)

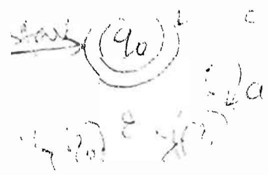


(6) DFA to accept zero (or) more a's. $\Sigma = (\text{set } a) = \{a\}$

↳ [any no. of a's]

ε, a, aa





$\varepsilon \notin \Sigma$

* DFA Design Techniques:-

There are 3 types of problems for which we can construct a DFA,

(1.) Pattern Recognition problems.
(2) Divisible by K problems.
(3) Modulo-K counter problems.

1) Pattern Recognition problems:-

For these type of problems that involve pattern recognition the DFA can be constructed very easily. The various steps to be followed are:-

Step 1: Identify the input alphabets.

Step 2: Identify the minimal string.

Step 3: Construct a skeleton DFA.

Step 4: Identify other transitions not defined in step 3. (missing transitions)

Step 5: Construct a DFA using transitions on step 3 and step 4.

Problem (1): Construct DFA to accepts strings of a's having atleast one 'a'.

Soln:

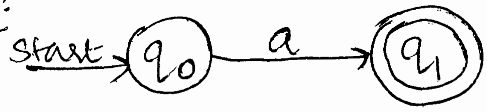
Step 1: Identify input alphabets.

$$\Sigma = \{a\}$$

Step 2: Identify the minimal string.

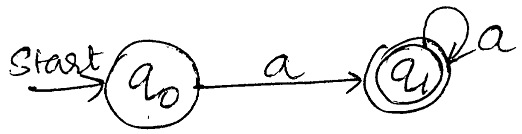
minstring = a

step3:



Step 4: $\delta(q_1, a) = ?$

Step 5: $\delta(q_1, a) = q_1$



$\therefore$ The resulting DFA is,

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

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

$$\Sigma = \{a\}$$

δ	a
$\rightarrow q_0$	q_1
$* q_1$	q_1

$q_0 \in Q$ is the start state

$F \subseteq Q = \{q_1\}$ is set of final states

To accept the string: aaa

$q_0 \xrightarrow{a} q_1 \xrightarrow{a} q_1 \xrightarrow{a} q_1$

(start) (final)

$\therefore$ The string aaa is accepted.

To reject the string : ε

$q_0 \xrightarrow{\varepsilon}$ struct

$\therefore$ The string ε is rejected.

DFA:

For an input there should be zero (or) only 1 transition.

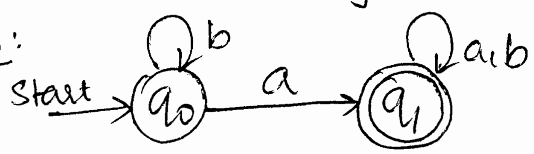
Q2) Draw a DFA to accept strings of a's & b's having atleast one 'a'.

Soln:

Step 1: $\Sigma = \{a, b\}$

Step 2: min. string = a

Step 3:



∴ The resulting DFA is,

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

$Q = \{q_0, q_1\}$

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

δ	a	b
$\rightarrow q_0$	q_1	q_0
$*q_1$	q_1	q_1

$q_0 \in Q$ is the start state.

$F \subseteq Q = \{q_1\}$ set of final states.

To accept string: aba

$q_0 \xrightarrow{a} q_1 \xrightarrow{b} q_1 \rightarrow q_1$

(start) (final)

∴ string aba is accepted.

To reject a string: bbb

$q_0 \xrightarrow{b} q_0 \xrightarrow{b} q_0$

(Start)

∴ string bbb is rejected.

[∵ on accepting bbb we cannot reach any of the final states].

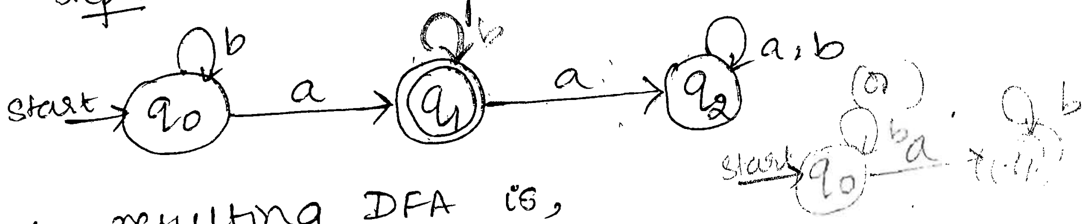
Q3) Draw a DFA to accept strings of a's & b's having exactly one 'a'.

Soln:

Step 1: $\Sigma = \{a, b\}$

Step 2: min. string = a

Step 3:



$\therefore$ The resulting DFA is,

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

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

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

δ	a	b
$\rightarrow q_0$	q_1	q_0
$* q_1$	q_2	q_1
q_2	q_2	q_2

$q_0 \in Q$ is the start state.

$F \subseteq Q = \{q_1\}$ set of finalstates.

To accept a string: abb

$q_0 \xrightarrow{a} q_1 \xrightarrow{b} q_1 \xrightarrow{b} q_1$

(Start) (final)

$\therefore$ string abb is accepted.

To reject a string: baa

$q_0 \xrightarrow{b} q_0 \xrightarrow{a} q_1 \xrightarrow{a}$ struct

(Start)

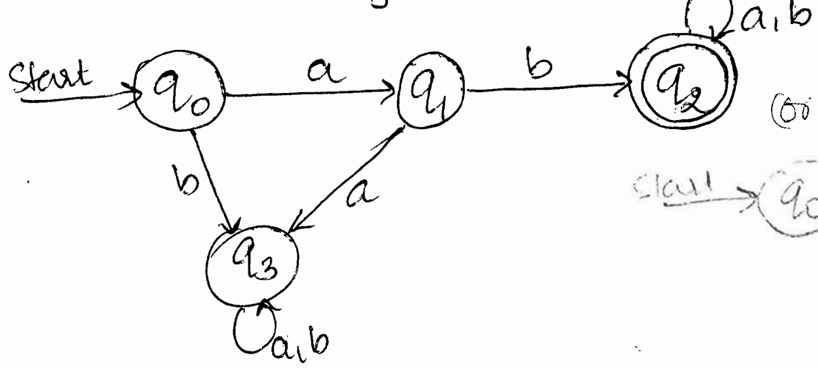
$\therefore$ string baa is rejected.

[$\because$ on accepting string baa we cannot reach any of the final states].

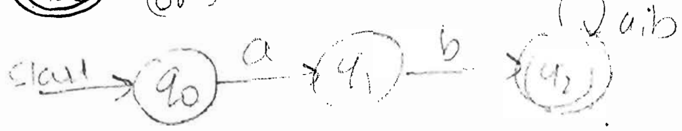
Q4) Obtain DFA to accept strings of a's & b's starting the string 'ab'.

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

min. string = ab



(or)



$\therefore$ The resulting DFA is,

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

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

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

δ	a	b
$\rightarrow q_0$	q_1	-
q_1	-	q_2
$* q_2$	q_2	q_2

$q_0 \in Q$ is the start state.

$F \subseteq Q = \{q_2\}$ is set of final states.

To accept a string: abb.

$$\underset{\text{(start)}}{q_0} \xrightarrow{a} q_1 \xrightarrow{b} q_2 \xrightarrow{b} \underset{\text{(final)}}{q_2}$$

$\therefore$ string abb is accepted.

To reject a string: bab.

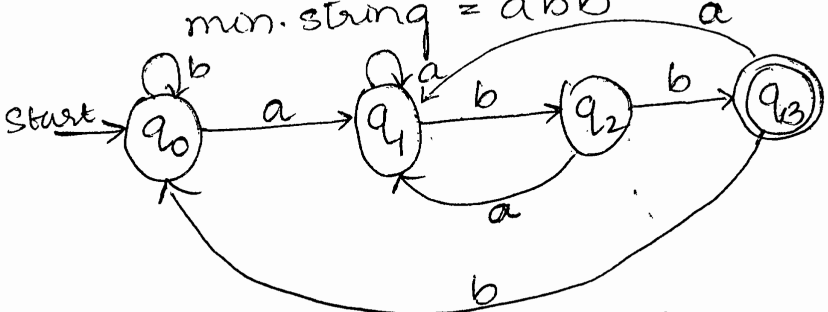
$q_0 \xrightarrow{b}$ struct

$\therefore$ string bab is rejected.

Q5) Draw a DFA to accept strings of a's & b's ending with string 'abb'.

solu?: $\Sigma = \{a, b\}$

min. string = abb



∴ The resulting DFA is,

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

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

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

δ	a	b
$\rightarrow q_0$	q_1	q_0
q_1	q_1	q_2
q_2	q_1	q_3
$* q_3$	q_1	q_0

$q_0 \in Q$ is the start state.

$F \subseteq Q = \{q_3\}$ is set of final states.

To accept a string: bababb

$q_0 \xrightarrow{b} q_0 \xrightarrow{a} q_1 \xrightarrow{b} q_2 \xrightarrow{a} q_1 \xrightarrow{b} q_2 \xrightarrow{b} q_3$

(start) (final)

∴ string bababb is accepted.

To reject a string: aab

$q_0 \xrightarrow{a} q_1 \xrightarrow{a} q_1 \xrightarrow{b} q_2$

(start) (nonfinal)

∴ string aab is rejected

[∵ we cannot reach the final state on accepting string aab].

21/8/18

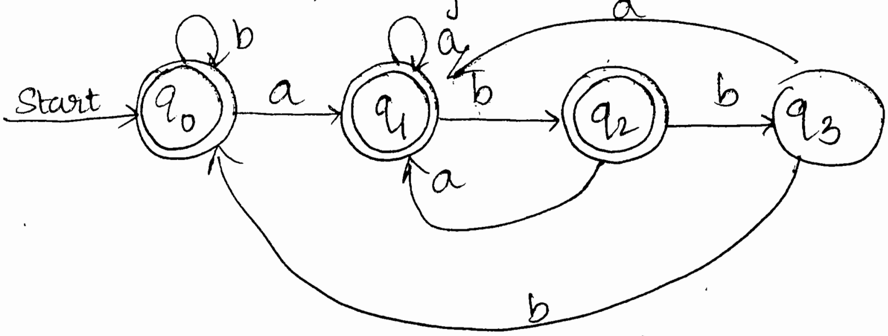
Q5) Draw a DFA to accept strings of a's & b's which donot end with the substring 'abb'.

Make final as nonfinal + nonfinal as final

Soln:

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

min. string = abb, ε



$\therefore$ The resulting DFA is,

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

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

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

δ	a	b
$\rightarrow * q_0$	q_1	q_0
$* q_1$	q_1	q_2
$* q_2$	q_1	q_3
q_3	q_1	q_0

$q_0 \in Q$ is the start state

$F \subseteq Q = \{q_0, q_1, q_2\}$ are set of final states.

To accept a string: abba

$q_0 \xrightarrow{a} q_1 \xrightarrow{b} q_2 \xrightarrow{b} q_3 \xrightarrow{a} q_1$

(Start) (final)

$\therefore$ string abba is accepted.

To reject a string: aabb

$q_0 \xrightarrow{a} q_1 \xrightarrow{a} q_1 \xrightarrow{b} q_2 \xrightarrow{b} q_3$

(Start) (non-final)

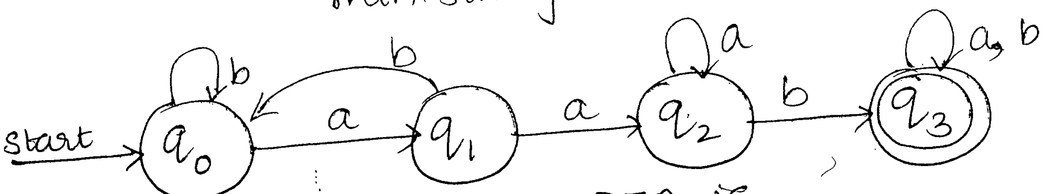
$\therefore$ string aabb is rejected.

Q7) Draw a DFA to accept strings of a's & b's having a substring 'aab'.

Solu^n:

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

min. string = aab



∴ The resulting DFA is,

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

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

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

δ	a	b
$\to q_0$	q_1	q_0
q_1	q_2	q_0
q_2	q_2	q_3
$* q_3$	q_3	q_3

$q_0 \in Q$ is the start state.

$F \subseteq Q = \{q_3\}$ is set of final states.

To accept a string: babaab

$q_0 \xrightarrow{b} q_0 \xrightarrow{a} q_1 \xrightarrow{b} q_0 \xrightarrow{a} q_1 \xrightarrow{a} q_2 \xrightarrow{b} q_3$

(start) (final)

∴ string babaab is accepted.

To reject a string: abba

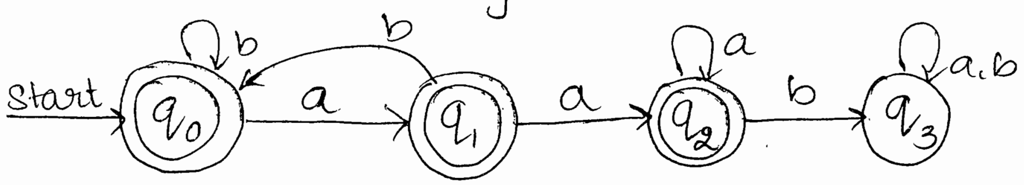
$q_0 \xrightarrow{a} q_1 \xrightarrow{b} q_0 \xrightarrow{b} q_0 \xrightarrow{a} q_1$

(start) (non final)

∴ string abba is rejected.

Q8) Draw a DFA to accept strings of a's and b's. except those having substring 'aab'.

Soln:
$\Sigma = \{a, b\}$
min. string = aab



∴ The resulting DFA is,
$M = \{Q, \Sigma, \delta, q_0, F\}$

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

δ	a	b
$\rightarrow * q_0$	q_1	q_0
$* q_1$	q_2	q_0
$* q_2$	q_2	q_3
q_3	q_3	q_3

$q_0 \in Q$ is the start state.

$F \subseteq Q = \{q_0, q_1, q_2\}$ are set of final states.

To accept string: abaa

$q_0 \xrightarrow{a} q_1 \xrightarrow{b} q_0 \xrightarrow{a} q_2 \xrightarrow{a} q_2$
(Start) (final)

∴ String abaa is accepted.

To reject string: baab

$q_0 \xrightarrow{b} q_0 \xrightarrow{a} q_1 \xrightarrow{a} q_2 \xrightarrow{b} q_3$
(start) (non final)

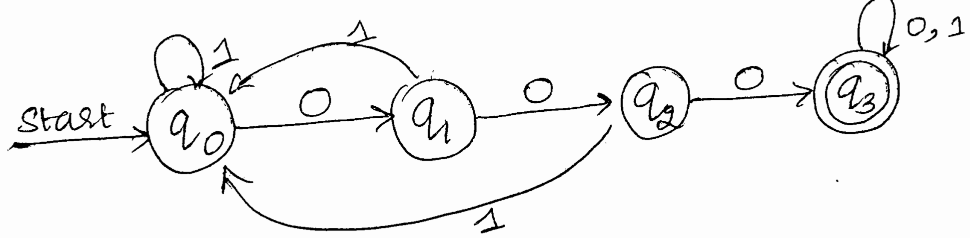
∴ String baab is rejected.

Q9) Draw DFA to accept strings of 0's and 1's having 3 consecutive 0's

Soln:

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

min. string = 000



$\therefore$ The resulting DFA is,

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

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

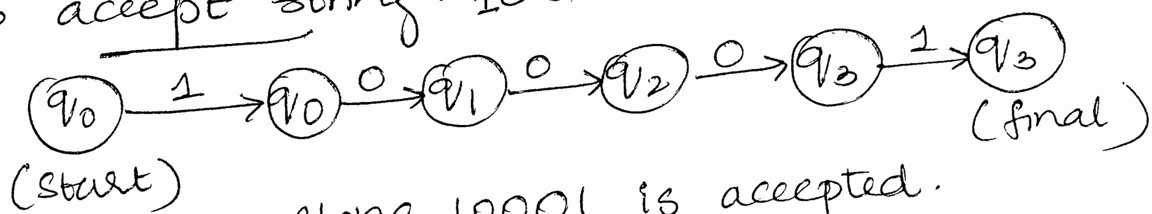
$\Sigma = \{0, 1\}$

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

$q_0 \in Q$ is the start state.

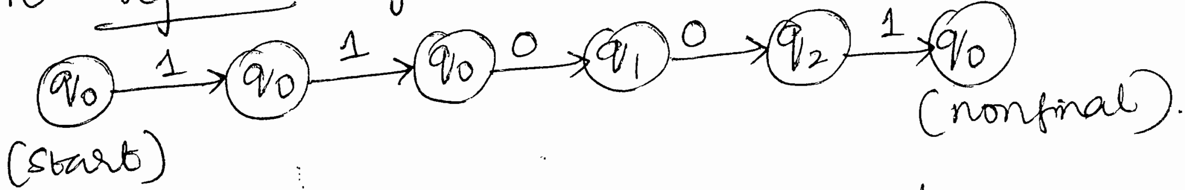
$F \subseteq Q = \{q_3\}$ is the final state.

To accept string: 10001



$\therefore$ string 10001 is accepted.

To reject string: 11001



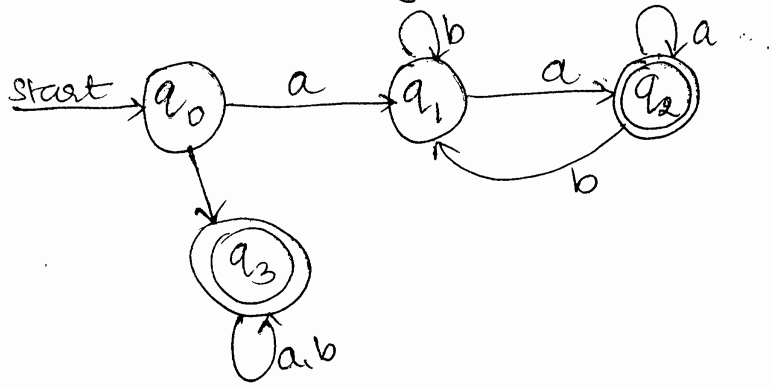
$\therefore$ string 11001 is rejected.

Q10) Draw a DFA to accept strings of a's & b's such that $L = \{ awa \mid w \in \{a,b\}^n$ where $n \geq 0 \}$.

Soln:

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

min. string = aa

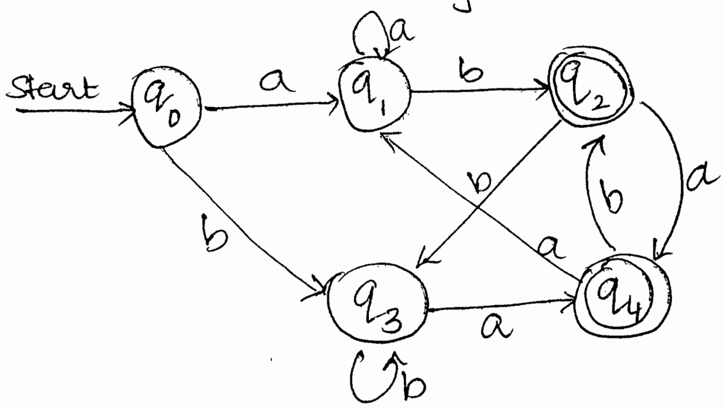


Q11) Draw a DFA to accept strings of a's & b's ending with 'ab' (or) 'ba'

Soln:

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

min. string = ab or ba

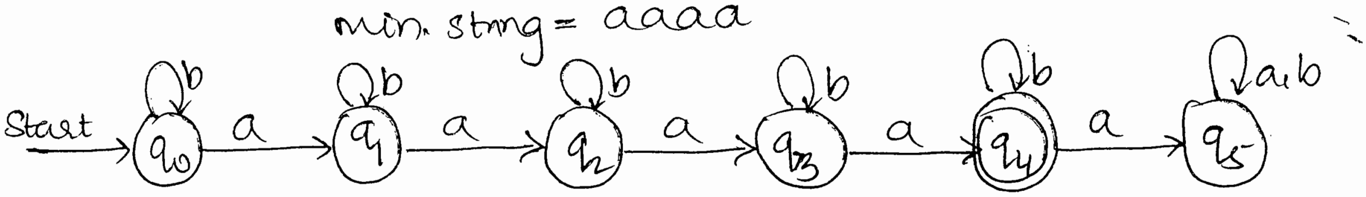


Q12) Obtain a DFA to accept strings of a's & b's having exactly four a's where $\Sigma = \{a, b\}$.

Soln:

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

min. string = aaaa



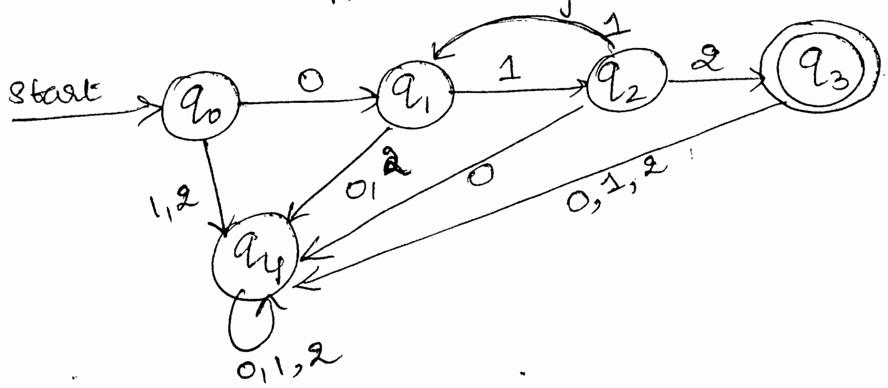
Q13) Obtain a DFA to accept strings of 0's, 1's & 2's beginning with a '0' followed by odd no. of 1's & ending with a '2'.

DFA → Regular language

Soln:

$\Sigma = \{0, 1, 2\}$

min. string = 012

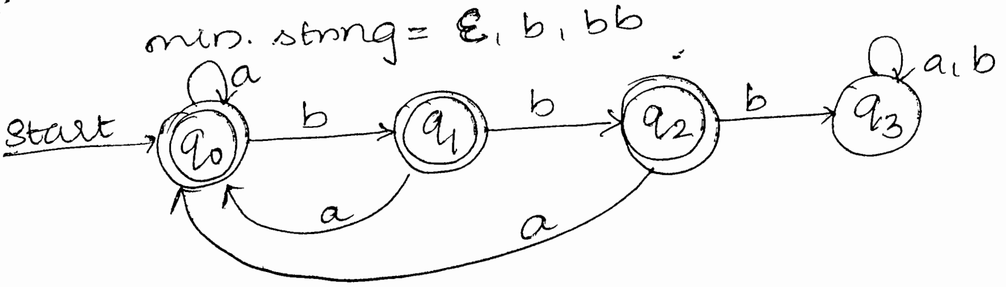


Q14) Obtain a DFA to accept strings of a's & b's with atmost two consecutive b's. ≤ 2

Soln:

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

min. string = ε, b, bb



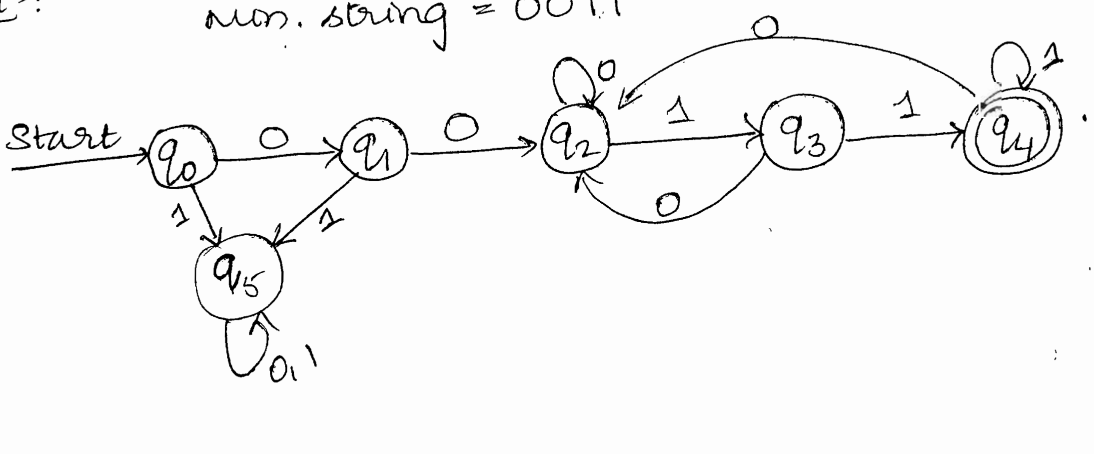
23/8/18

Q15) Obtain a DFA to accept strings of 0's & 1's starting with atleast two 0's & ending with atleast two 1's.

Soln:

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

min. string = 0011

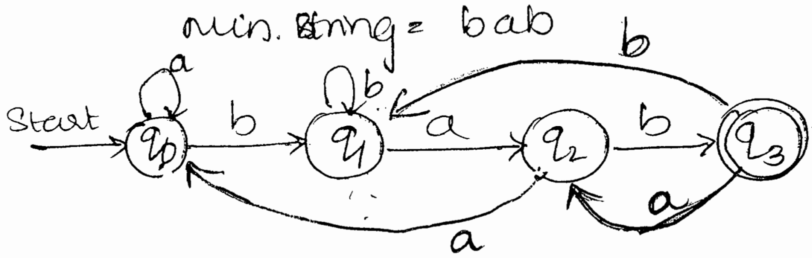


Q16) Obtain DFA to accept the language,

$L = \{ wbab \mid w \in \{a,b\}^* \}$

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

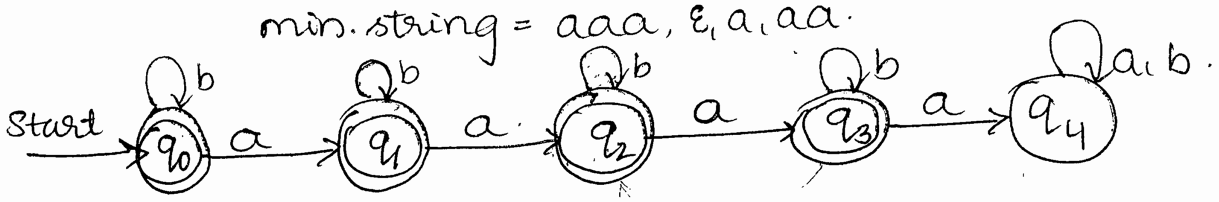
min. string = bab



Q17) Draw a DFA to accept strings of a's & b's having not more than 3 a's. $(a \le 3)$

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

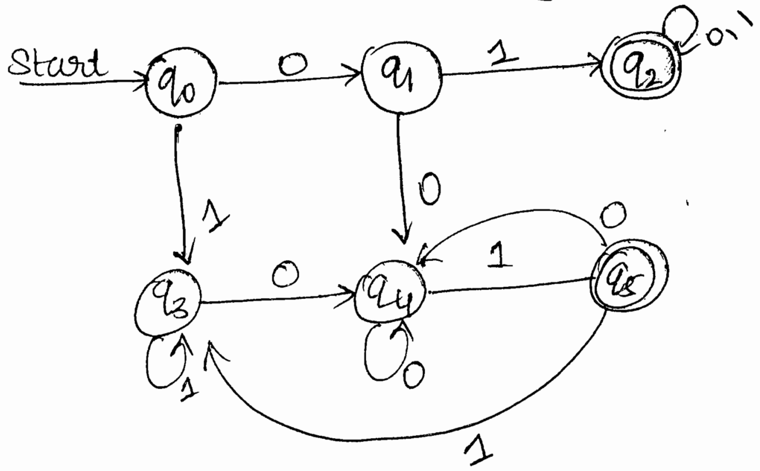
min. string = aaa, ε, a, aa.



Q18) Draw a DFA to accept set of all strings on the alphabet $\Sigma = \{0, 1\}$ that either begins (or) ends with substring 01

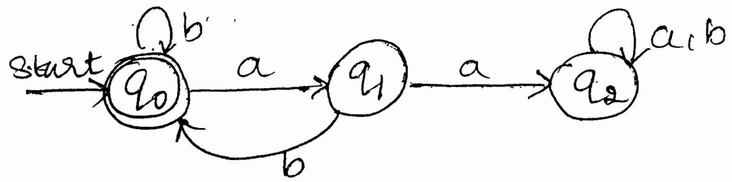
Soln: $\Sigma = \{0, 1\}$

min. string = 01

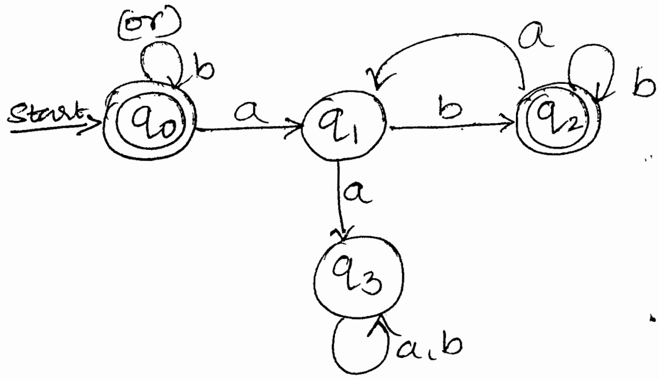


Q19) Obtain a DFA to accept a language,
$L = \{ w \in \{a,b\}^* :$ every 'a' is immediately followed by a 'b' $\}$

Soln: $\Sigma = \{a, b\}$
min. string = ε, b^*, ab^+



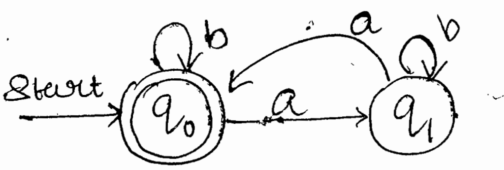
(or)



25/8/18

Q20) Obtain a DFA to accept language
$L = \{ w \in \{a,b\}^* :$ every 'a' region in w is of even length $\}$.

Soln: $\Sigma = \{a, b\}$
min. string = ε, b^*, aa

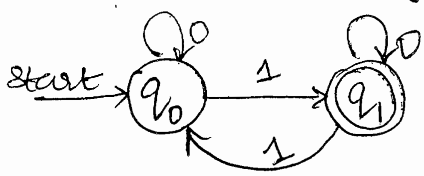


Q21) Obtain a DFA to accept language, $L = \{ w \in \{0,1\}^* :$ w has odd parity $\}$.
$\Sigma = \{0, 1\}$

Odd parity : If the no. of 1's is odd then it is odd parity string.

Solⁿ: $\Sigma = \{0, 1\}$

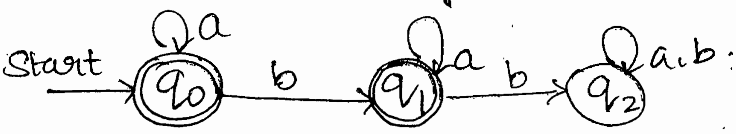
min. string = 1



Q22) Obtain DFA to accept language, $L = \{w \in \{a, b\}^* : w$ contains no more than one 'b'$\}$

Solⁿ: $\Sigma = \{a, b\}$

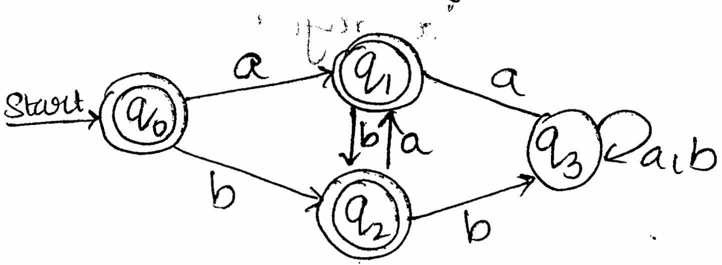
min. string = ε, b



Q23) Obtain a DFA to accept language, $L = \{w \in \{a, b\}^* :$ no consecutive characters are the same$\}$.

Solⁿ: $\Sigma = \{a, b\}$

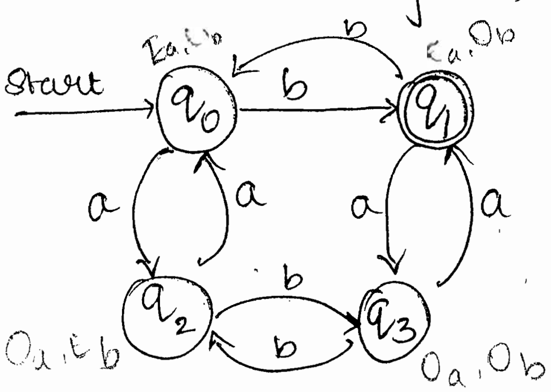
min. string = ε, a, b, $(ab)^+$



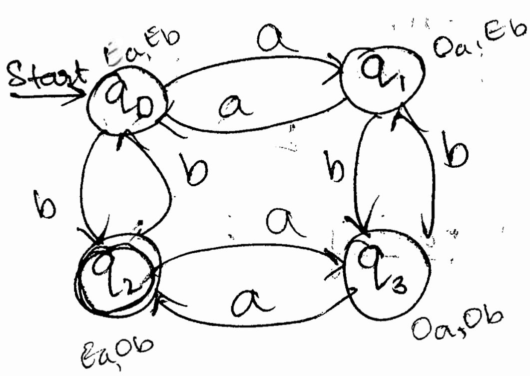
Q24) Obtain a DFA to accept language, $L = \{w \in \{a, b\}^* ;$ w contains an even no. of a's and ~~odd~~ odd no. of b's$\}$.

Solⁿ: $\Sigma = \{a, b\}$

min. string = b,



(or)



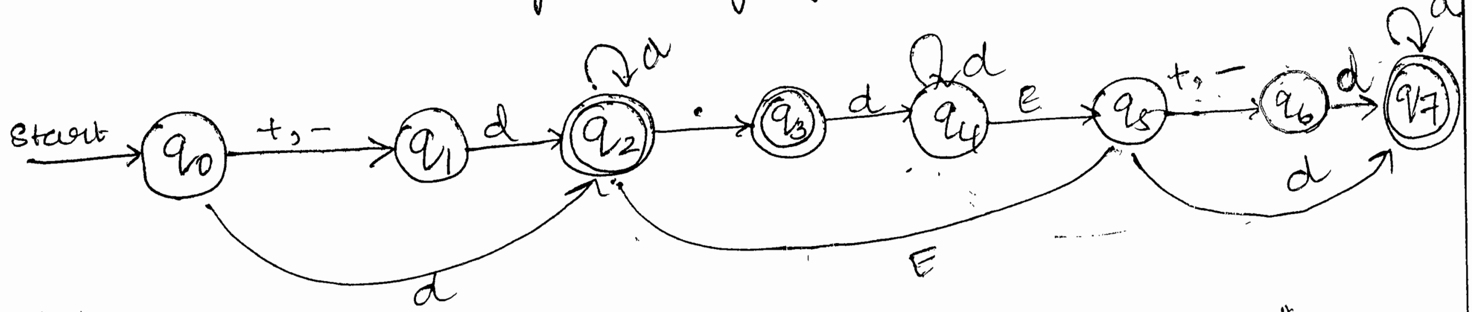
Q25) Obtain DFA to accept language, $L_{FLAT} = \{ w : w$ is the string representing a floating point number $\}$.

***

Soln: a) A optional sign (+, −) followed by string of digits followed by '.' followed by a string of digits.

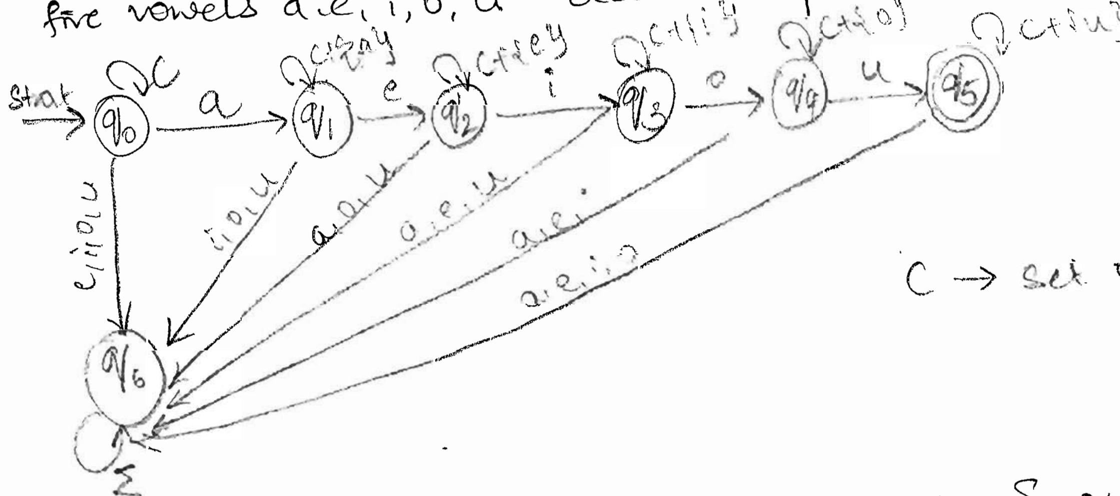
b) option @ followed by an exponent 'E' followed by optional sign followed by string of digits.

c) a string of digits followed by exponent 'E' followed by string of digits.



27/8/18

Q26) Obtain DFA to accept language, $L = \{ w \in \{a-z\}^* :$ all five vowels a, e, i, o, u occur in alphabetical order $\}$.

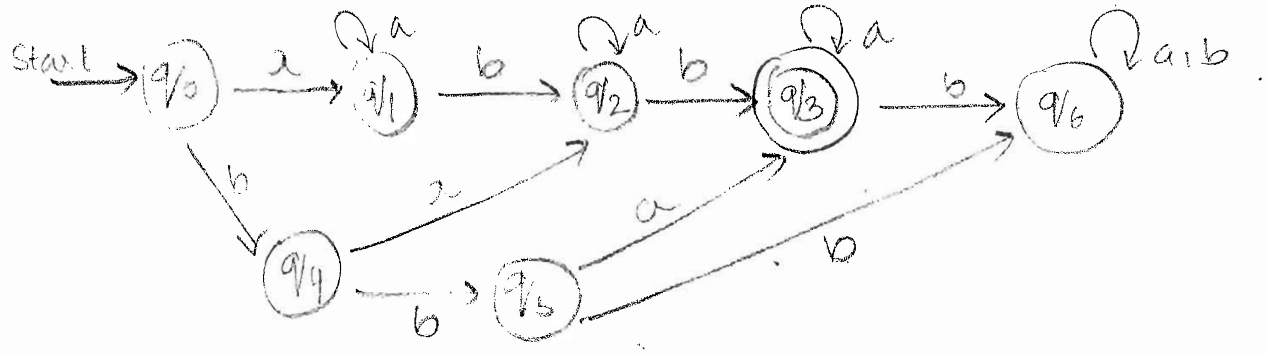


C → set of consonants

Q27) Obtain DFA to accept language, $L = \{ w : n_a(w) \geq 1, n_b(w) = 2 \}$.

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

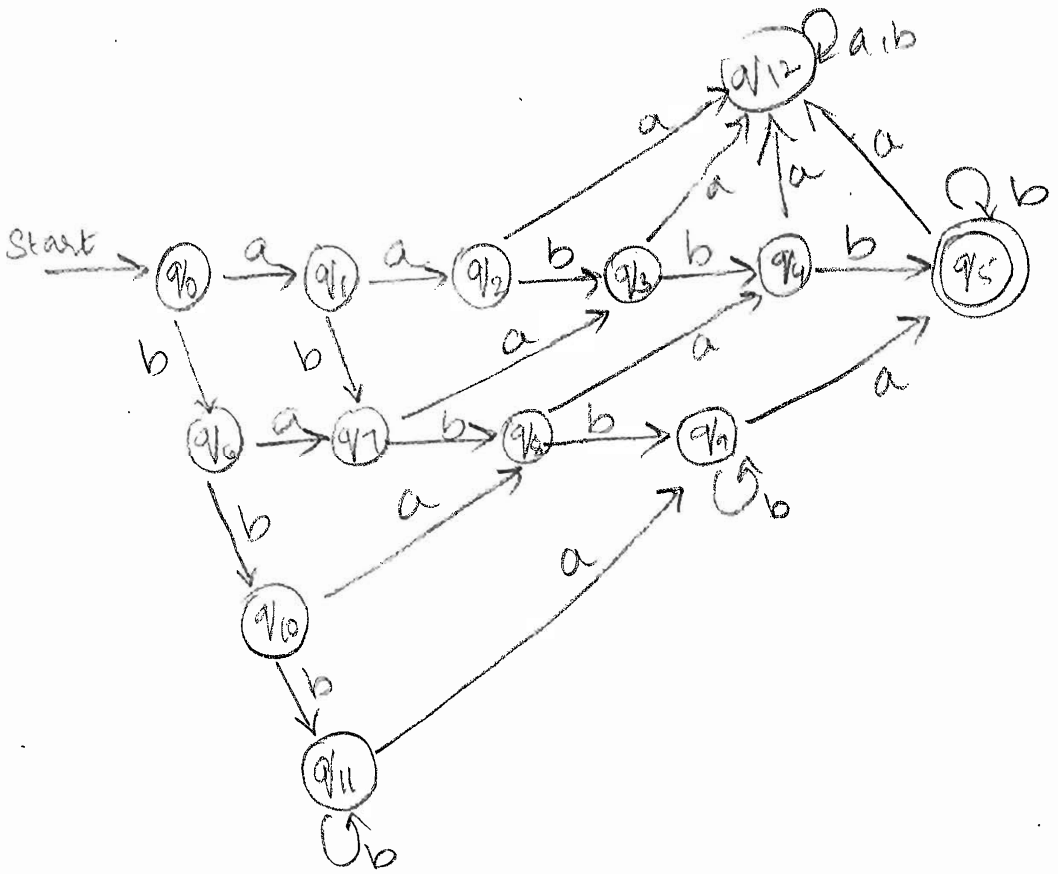
min. string = abb, bba, bab



Q28) Obtain a DFA to accept language
$L = \{w : n_a(w) = 2,\ n_b(w) \geq 3\}$.

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

min string = aabbb, ababb, abbab, abbba, bbbaa, bbaba, bbaab, baabb, babab, babba



28/8/18

Definition of NFA :-

The non-deterministic finite automata [NFA] is 5 tuple (or) quintuple indicating 5 components.

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

M : name of the machine.

Q : non-empty finite set of states.

Σ : non-empty finite set of input alphabets.

δ : $Q \times \Sigma \rightarrow 2^Q$ ie,

δ is a transition function which is mapping from $Q \times \Sigma \rightarrow 2^Q$. Based on current state and input symbol the machine enters into one (or) more states.

$q_0 \in Q$ is the start state.

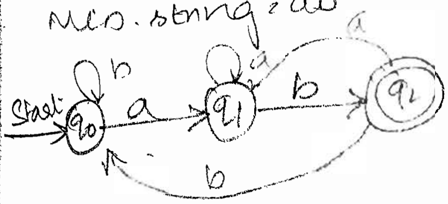
$F \subseteq Q$ is set of accepting (or) final states.

eg(1): Obtain an NFA to accept strings of a's & b's ending with 'ab'.

DFA

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

Min. string = ab

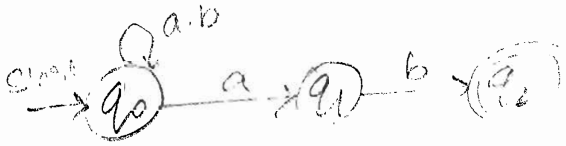


δ_D	a	b
$\rightarrow q_0$	q_1	q_0
q_1	q_1	q_2
$*q_2$	q_1	q_0

NFA

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

Min. string = ab



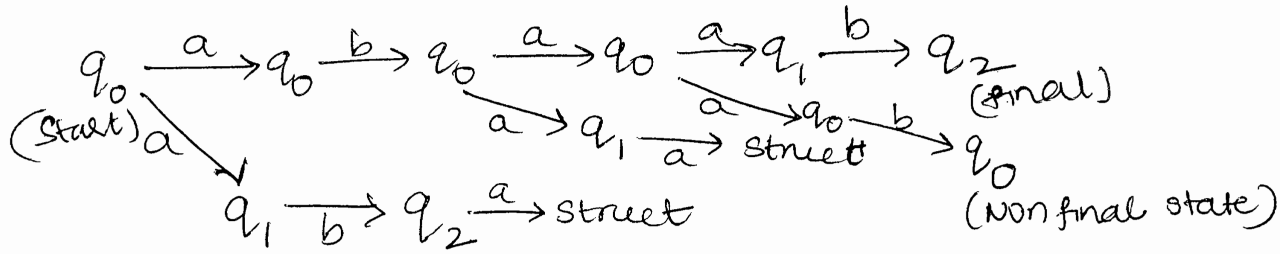
δ_N	a	b
$\rightarrow q_0$	$\{q_0, q_1\}$	$\{q_0\}$
q_1	ϕ	$\{q_2\}$
$*q_2$	ϕ	ϕ

M: $(Q, \Sigma, \ldots$

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

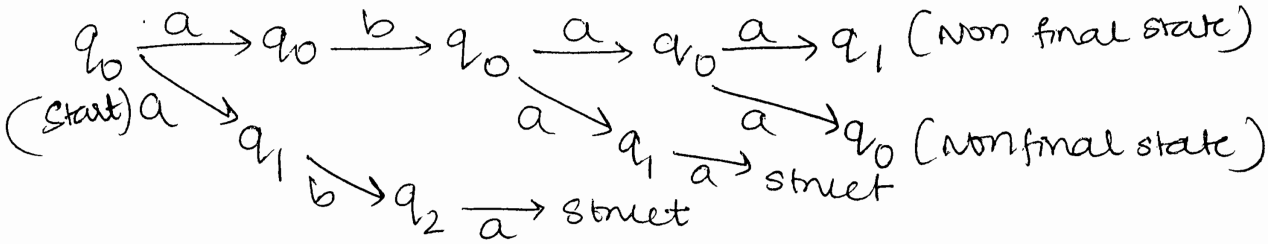
$\delta = \ldots$

To accept string: abaab



∴ string abaab is accepted.

To reject string: abaa

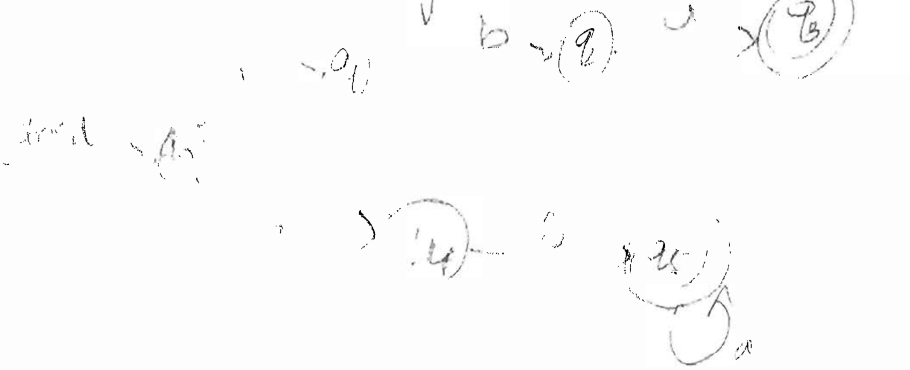


∴ string abaa is rejected.

Q1) Obtain an NFA to accept the following language $L = \{ w \mid w \in abab^n \text{ (or) } aba^n \text{ where } n \geq 0 \}$.

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

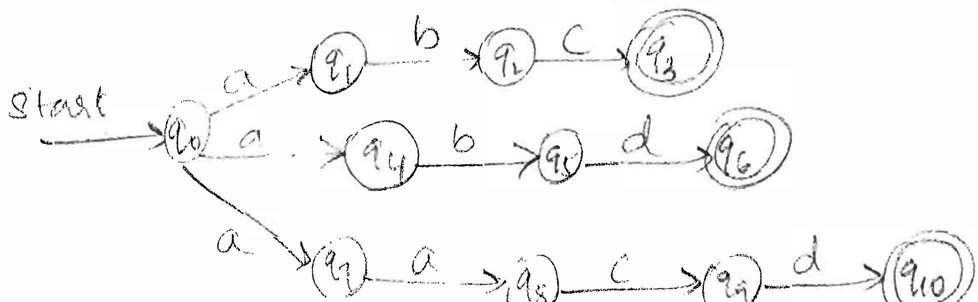
min string = aba, ab



Q2) Design an NFA to recognize the following set of strings abc, abd and aacd.

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

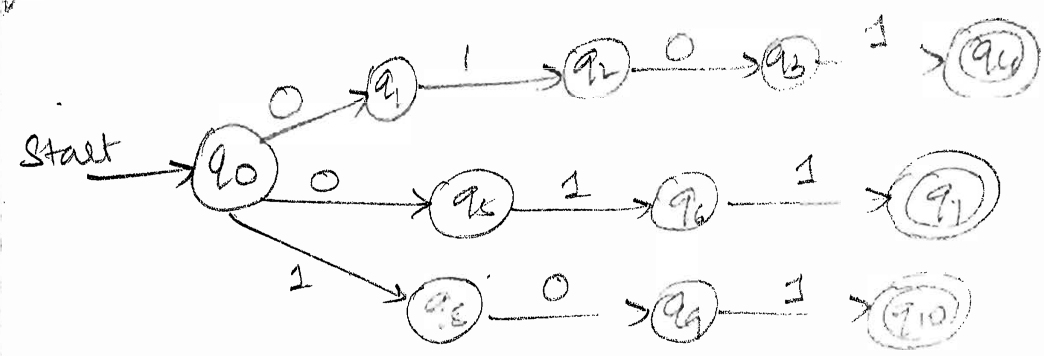
min string = abc, abd, aacd



Q3) Obtain NFA for the following strings 0101, 101, 011

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

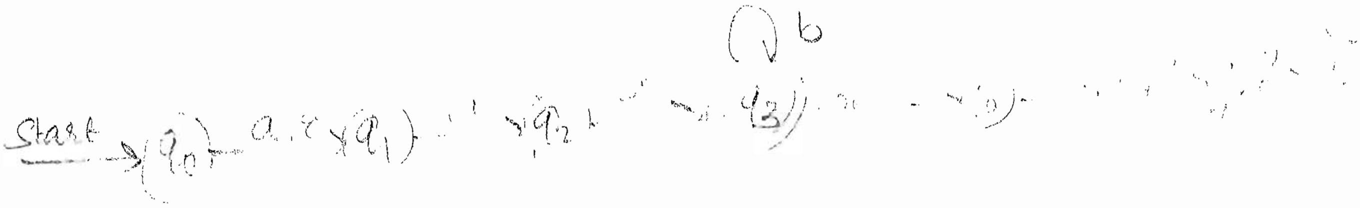
min string = 0101, 101, 011



Q4) Obtain an NFA to accept language
$L = \{ w \in \{a, b\}^* :$ w is made up of an optional 'a' followed by 'aa' followed by zero (or) more b's $\}$

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

min string $= (a+\epsilon)aab^*$

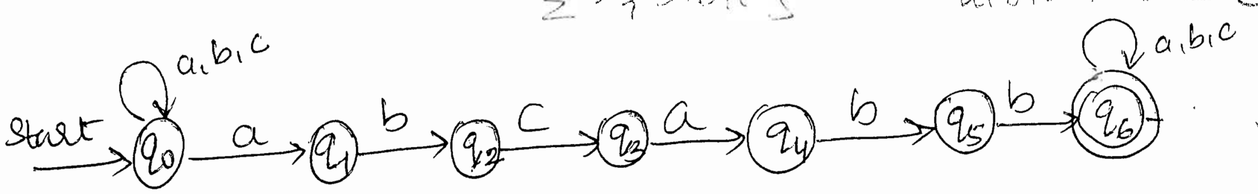


29/8/18

Q5) Obtain an NFA to accept a language
~~$L = \{ w \in \{a, b\}^* : w = aba$~~ (or)

$L = \{ w \in \{a, b, c\}^* : \exists x, y \in \{a, b, c\}^* \ (w = x\,abcabb\,y) \}$

$\Sigma = \{a, b, c\}$ $\quad (a+b+c)^*\,abcabb\,(a+b+c)^*$

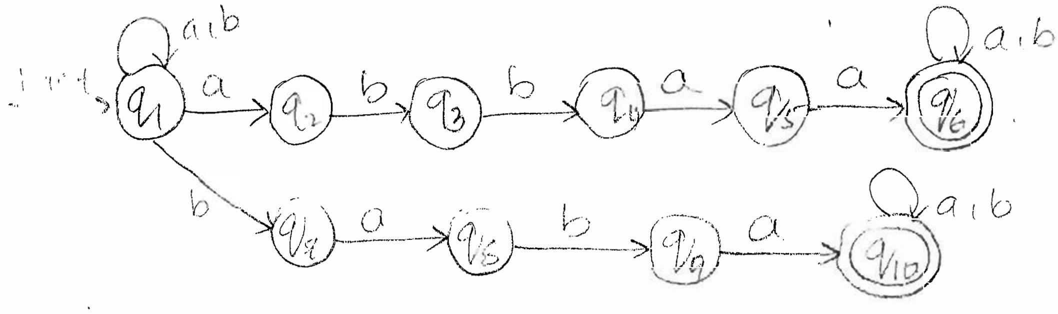


DFA:

Q6) Obtain an NFA to accept language,

$$L = \{ w \in \{a,b\}^* : \exists x,y \in \{a,b\}^* \; ((w = xabbaay) \vee (w = xbabay)) \}$$

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

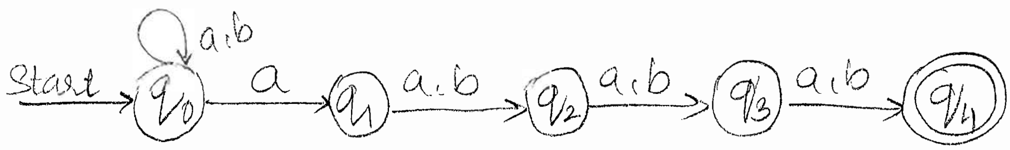


Q7) Obtain an NFA to accept a language,

$L = \{ w \in \{a,b\}^* :$ the fourth from the last character is 'a' $\}$.

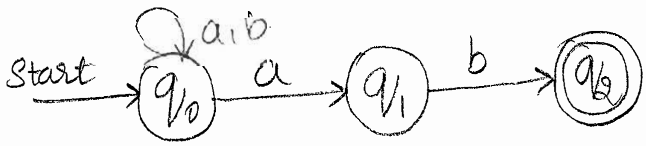
$$(a+b)^* a (a+b)^3$$

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



Conversion from NFA to DFA

Q1) Obtain DFA for the following NFA



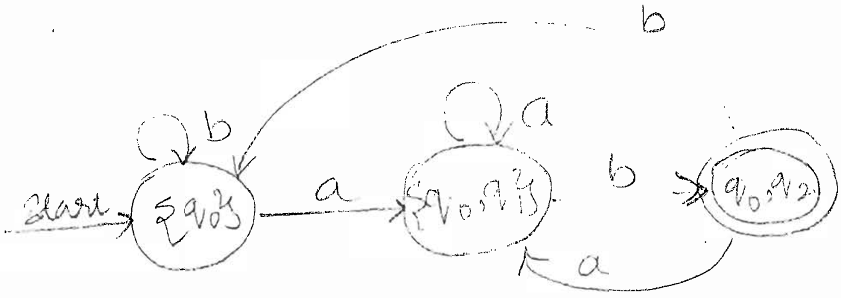
Soln: a) write a transition table for NFA.

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

b) NFA to DFA

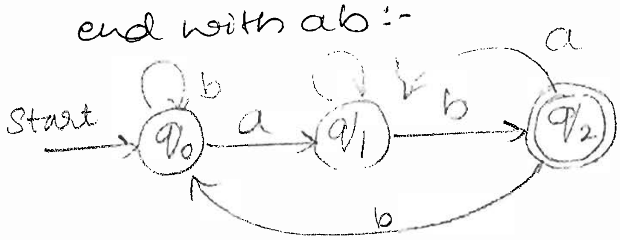
transition table for DFA

δ_D	a	b
$\rightarrow \{q_0\}$	$\{q_0,q_1\}$	$\{q_0\}$
$\{q_0,q_1\}$	$\{q_0,q_1\}$	$\{q_0,q_2\}$
$* \{q_0,q_2\}$	$\{q_0,q_1\}$	$\{q_0\}$

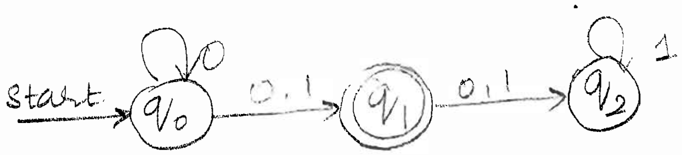


∴ The resulting DFA,
M = {

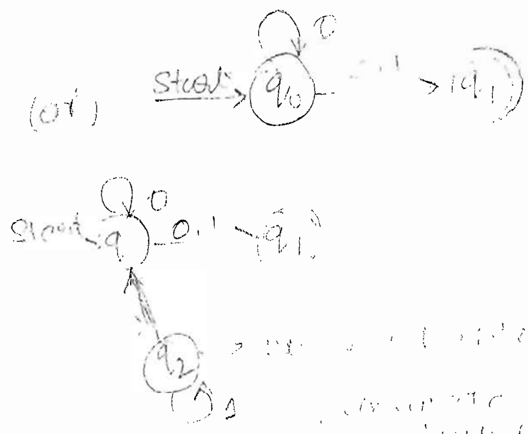
end with ab :-



Q2) Convert the following NFA to its equivalent DFA.



(or)

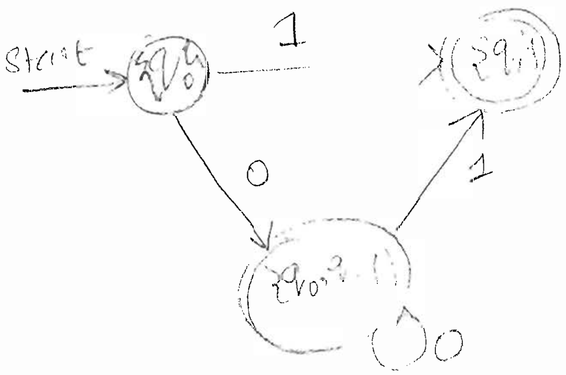


Soln:

δ_N	0	1
$\rightarrow q_0$	$\{q_0,q_1\}$	$\{q_1\}$
$* q_1$	ϕ	ϕ

NFA to DFA

δ_D	0	1
$\rightarrow \{q_0\}$	$\{q_0,q_1\}$	$\{q_1\}$
$* \{q_1\}$	ϕ	ϕ
$* \{q_0,q_1\}$	$\{q_0,q_1\}$	$\{q_1\}$



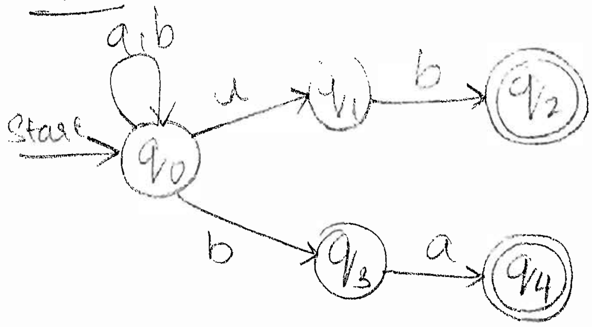
30/8/18

Q3) Obtain an NFA to accept strings of a's and b's ending with 'ab' (or) 'ba'. From this NFA obtain an equivalent DFA.

Soln:

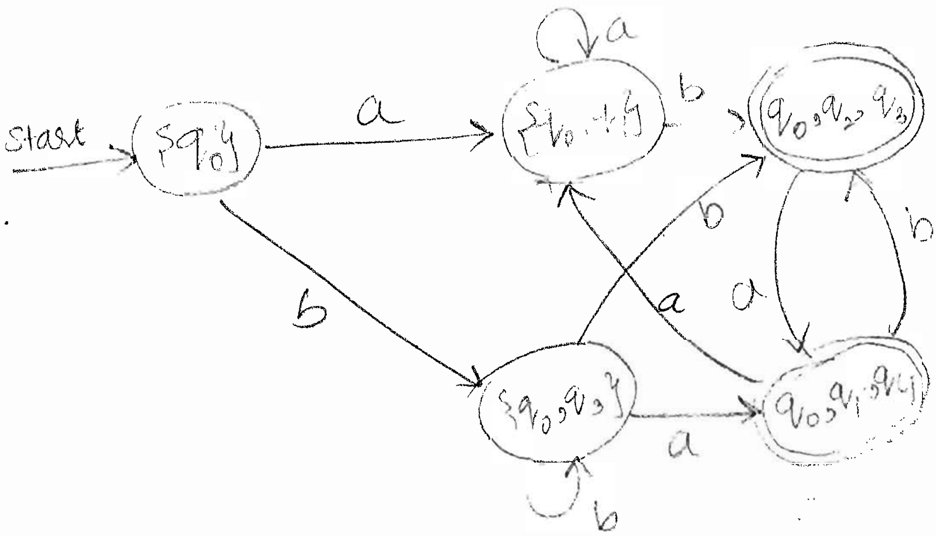
$(a+b)^*(ab+ba)$

NFA:



δ_N	a	b
$\rightarrow q_0$	$\{q_0, q_1\}$	$\{q_0, q_3\}$
q_1	ϕ	$\{q_2\}$
$* q_2$	ϕ	ϕ
q_3	$\{q_4\}$	ϕ
$* q_4$	ϕ	ϕ

δ_D	a	b
$\rightarrow \{q_0\}$	$\{q_0, q_1\}$	$\{q_0, q_3\}$
$\{q_0, q_1\}$	[illegible]	[illegible] $q_3\}$
$\{q_0, q_3\}$	[illegible]	$\{q_0, q_3\}$
$* \{q_0, q_2, q_3\}$	$\{q_0, q_1\}$	$\{q_0, q_2, q_3\}$
$*$ [illegible]	$\{q_0, q_1, q_4\}$	$\{q_0, q_3\}$



[illegible] happens without any input closure

cascading transitions

31

D

5

Q

31/8/18

Definition of ε-NFA:

The ε-NFA is a five tuple (or) quintuple indicating 5 components, $M = (Q, \Sigma, \delta, q_0, F)$

where, $M \to$ name of the machine

$Q \to$ nonempty finite set of states.

$\Sigma \to$ non empty finite set of input alphabets.

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

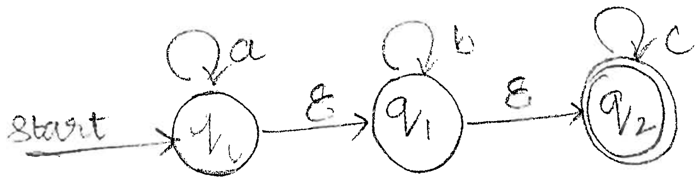
ie, δ is a transition function, mapping from $Q \times \{\Sigma \cup \varepsilon\} \to 2^Q$.

Based on the current state, there can be transitions to other states with (or) without any input symbols.

$q_0 \in Q$ is the start state.

$F \subseteq Q$ is set of accepting/final states

Q1) Obtain an ε-NFA which accepts strings consisting of zero (or) more a's followed by zero (or) more b's followed by zero (or) more c's.



∴ The resulting ε-NFA is,

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

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

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

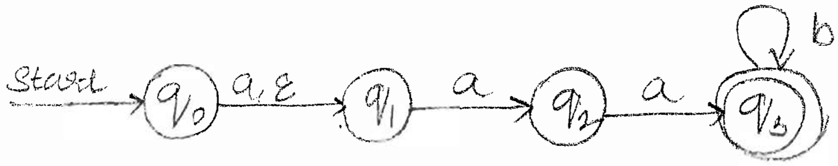
$q_0 \in Q$ is the start state.

$F \subseteq Q = \{q_2\}$ is the set of final state.

δ_E	a	b	c	ε
$\to q_0$	$\{q_0\}$	ϕ	ϕ	$\{q_1\}$
q_1	ϕ	$\{q_1\}$	ϕ	$\{q_2\}$
$* q_2$	ϕ	ϕ	$\{q_2\}$	ϕ

Q2) Obtain an ε-NFA to accept the language,

$L = \{ w \in \{a,b\}^* :$ w is made up of an optional 'a' followed by 'aa' followed by zero (or) more b's $\}$.



The resulting ε-NFA is,

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

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

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

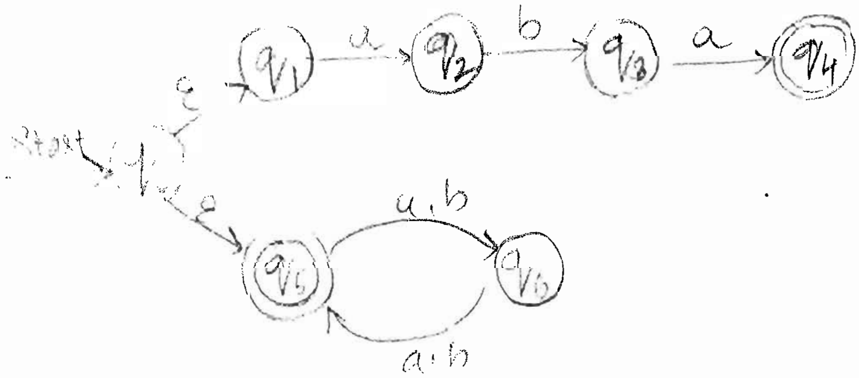
$q_0 \in Q$ is the start state

$F \subseteq Q = \{q_3\}$ is final state.

δ_E	a	b	ε
$\rightarrow q_0$	$\{q_1\}$	ϕ	$\{q_1\}$
q_1	$\{q_2\}$	ϕ	ϕ
q_2	$\{q_3\}$	ϕ	ϕ
$* q_3$	ϕ	$\{q_3\}$	ϕ

Q3) Obtain an ε-NFA to accept a language,

$L = \{ w \in \{a,b\}^* : w = aba$ (or) $|w|$ is even$\}$



The resulting ε-NFA is,

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

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

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

$q_0 \in Q$ is the start state

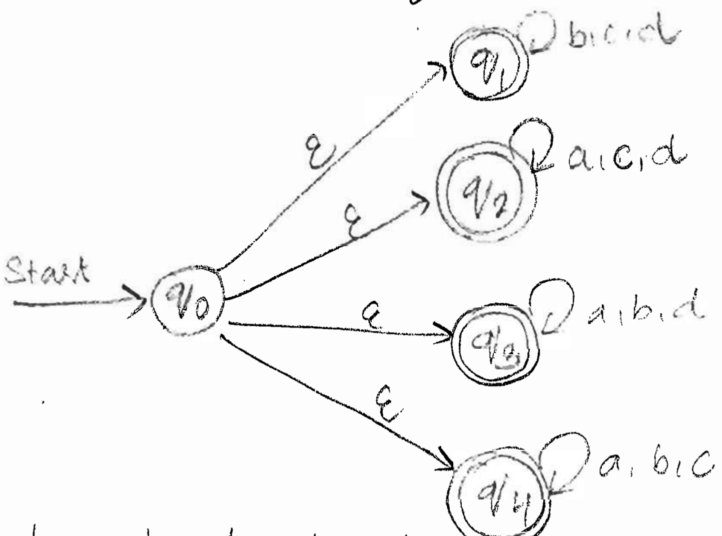
$F \subseteq Q = \{q_4, q_5\}$ are set of final states

δ_E	a	b	ε
$\rightarrow q_0$	ϕ	ϕ	$\{q_1, q_5\}$
q_1	$\{q_2\}$	ϕ	ϕ
q_2	ϕ	$\{q_3\}$	ϕ
q_3	$\{q_4\}$	ϕ	ϕ
$* q_4$	ϕ	ϕ	ϕ
$* q_5$	$\{q_6\}$	$\{q_6\}$	ϕ
q_6	$\{q_5\}$	$\{q_5\}$	ϕ

Q4) Obtain an ε-NFA to accept the language,

$L_{missing} = \{w :$ there is a symbol $a_i \in \Sigma$ not appearing in $w\}$

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



The resulting ε-NFA is

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

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

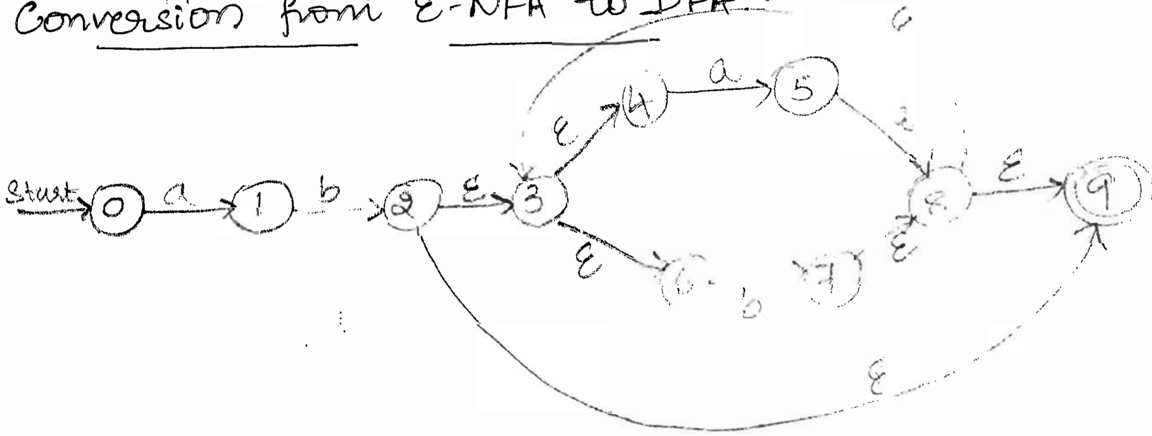
$\Sigma = \{a, b, c, d\}$

$q_0 \in Q$ is start state

$F \subseteq Q = \{q_1, q_2, q_3, q_4\}$ is set of final state

δ_E	a	b	c	d	ε
$\rightarrow q_0$	φ	φ	φ	φ	$\{q_1, q_2, q_3, q_4\}$
$*q_1$	φ	$\{q_1\}$	$\{q_1\}$	$\{q_1\}$	φ
$*q_2$	$\{q_2\}$	φ	$\{q_2\}$	$\{q_2\}$	φ
$*q_3$	[illegible]	[illegible]	φ	[illegible]	φ
$*q_4$	[illegible]	$\{q_4\}$	$\{q_4\}$	φ	φ

Conversion from ε-NFA to DFA:-



Q1) Convert the following ε-NFA to DFA.

Step1:-

ECLOSE(0) = {0}

ECLOSE(1) = {1}

ECLOSE(2) = {2, 3, 4, 6, 9}

ECLOSE(3) = {3, 4, 6}

ECLOSE(4) = {4}

ECLOSE(5) = {3, 4, 5, 6, 8, 9}

ECLOSE(6) = {6}

ECLOSE(7) = {3, 4, 6, 7, 8, 9}

ECLOSE(8) = {3, 4, 6, 8, 9}

ECLOSE(9) = {9}

Step2: Start state

ECLOSE$(0) = \{0\}$ – A

Step3: compute the transitions.

* consider state A

$\delta(A, a) = \text{ECLOSE}(\delta_E(A, a))$

$= \text{ECLOSE}(\delta_E(\{0\}, a))$

$= \text{ECLOSE}(\{1\}) = \{1\}$ – B

$\therefore \boxed{\delta(A, a) = B}$

$\delta(A, b) = \text{ECLOSE}(\delta_E(\{0\}, b))$

$= \text{ECLOSE}(\phi) = \phi$

$\therefore \boxed{\delta(A, b) = \phi}$

* consider state B

$\delta(B, a) = \text{ECLOSE}(\delta_E((\{1\}), a))$

$= \text{ECLOSE}(\delta_E(\phi))$

$= \phi$

$\therefore \boxed{\delta(B, a) = \phi}$

$\delta(B, b) = \text{ECLOSE}(\delta_E(\{1\}, b))$

$= \text{ECLOSE}(\{2\}) = \{2, 3, 4, 6, 9\}$ – C

$\therefore \boxed{\delta(B, b) = C}$

* consider C

$\delta(C, a) = \text{ECLOSE}(\delta_E(\{2, 3, 4, 6, 9\}, a))$

$= \text{ECLOSE}(\{5\}) = \{3, 4, 5, 6, 8, 9\}$ – D

$\therefore \boxed{\delta(C, a) = D}$

$\delta(C, b) = \text{ECLOSE}(\delta_E(\{2, 3, 4, 6, 9\}, b))$

$= \text{ECLOSE}(\{7\}) = \{3, 4, 6, 7, 8, 9\}$ – E

$\therefore \boxed{\delta(C, b) = E}$

* consider D

$\delta(D,a) = ECLOSE(\delta_E(\{3,4,5,6,8,9\},a))$

$= ECLOSE(\{5\}) = \{3,4,5,6,8,9\}$ — D

$\therefore \boxed{\delta(D,a) = D}$

$\delta(D,b) = ECLOSE(\delta_E(\{3,4,5,6,8,9\},b))$

$= ECLOSE(\{7\})$ — E

$\therefore \boxed{\delta(D,b) = E}$

* consider E

$\delta(E,a) = ECLOSE(\delta_E(\{3,4,6,7,8,9\},a))$

$= ECLOSE(\{5\})$ — D

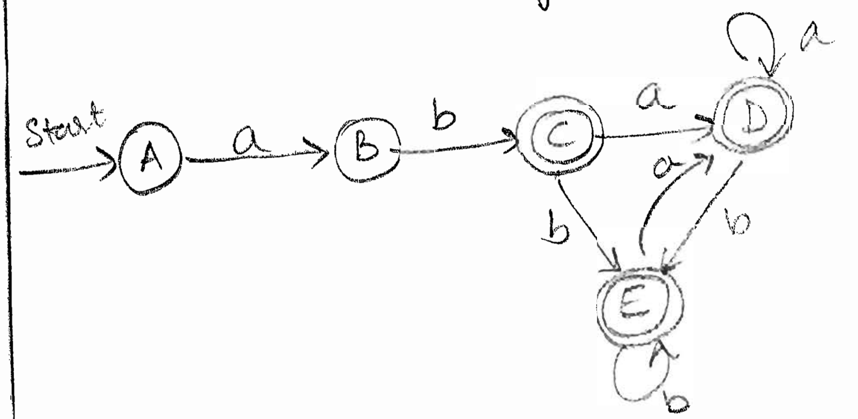
$\therefore \boxed{\delta(E,a) = D}$

$\delta(E,b) = ECLOSE(\delta_E(\{3,4,6,7,8,9\},b))$

$= ECLOSE(\{7\})$ — E

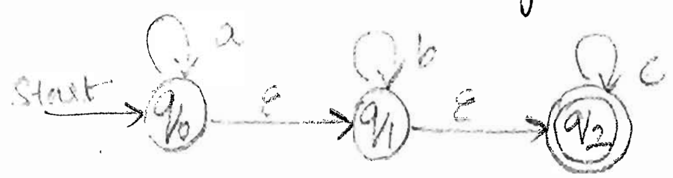
$\therefore \boxed{\delta(E,b) = E}$

$\therefore$ The resulting DFA,



δ_D	a	b
→A	B	ϕ
B	ϕ	C
*C	D	E
*D	D	E
*E	D	E

Q2) convert the following ε-NFA to its equivalent DFA.



Step1: $ECLOSE(q_0) = \{q_0, q_1, q_2\}$

$ECLOSE(q_1) = \{q_1, q_2\}$

$ECLOSE(q_2) = \{q_2\}$

Step2: start state.

$ECLOSE(q_0) = \{q_0, q_1, q_2\}$ — A

Step3: Compute the transitions.

$\delta(A, a) = ECLOSE(\delta_E(\{q_0, q_1, q_2\}, a))$

$= ECLOSE(\{q_0\}) = \{q_0, q_1, q_2\}$ — A

$\therefore \boxed{\delta(A, a) = A}$

$\delta(A, b) = ECLOSE(\delta_E(\{q_0, q_1, q_2\}, b))$

$= ECLOSE(\{q_1\}) = \{q_1, q_2\}$ — B

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

$\delta(A, c) = ECLOSE(\delta_E(\{q_0, q_1, q_2\}, c))$

$= ECLOSE(\{q_2\}) = \{q_2\}$ — C

$\therefore \boxed{\delta(A, c) = C}$

Consider B

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

$= ECLOSE(\phi)$ — ϕ

$\therefore \boxed{\delta(B, a) = \phi}$

$\delta(B,b) = ECLOSE(\delta_E(\{q_1,q_2\},b))$

$= ECLOSE(\{q_1\}) = \{q_1,q_2\}$ — B

$\therefore \boxed{\delta(B,b) = B}$

$\delta(B,c) = ECLOSE(\delta_E(\{q_1,q_2\},c))$

$= ECLOSE(\{q_2\})$ — C

$\therefore \boxed{\delta(B,c) = C}$

consider C

$\delta(C,a) = ECLOSE(\delta_E(\{q_2\},a))$

$= \phi$

$\therefore \boxed{\delta(C,a) = \phi}$

$\delta(C,b) = ECLOSE(\delta_E(\{q_2\},b))$

$= \phi$

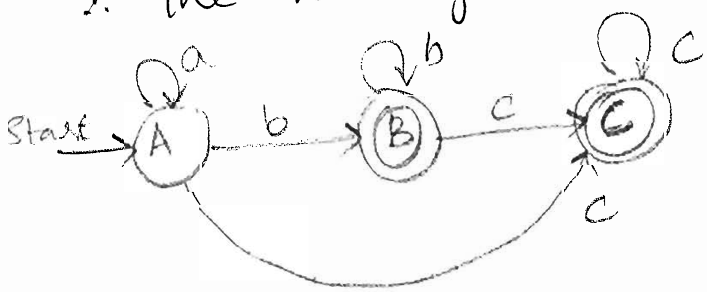
$\therefore \boxed{\delta(C,b) = \phi}$

$\delta(C,c) = ECLOSE(\delta_E(\{q_2\},c))$

$= ECLOSE(\{q_2\})$ — C

$\therefore \boxed{\delta(C,c) = C}$

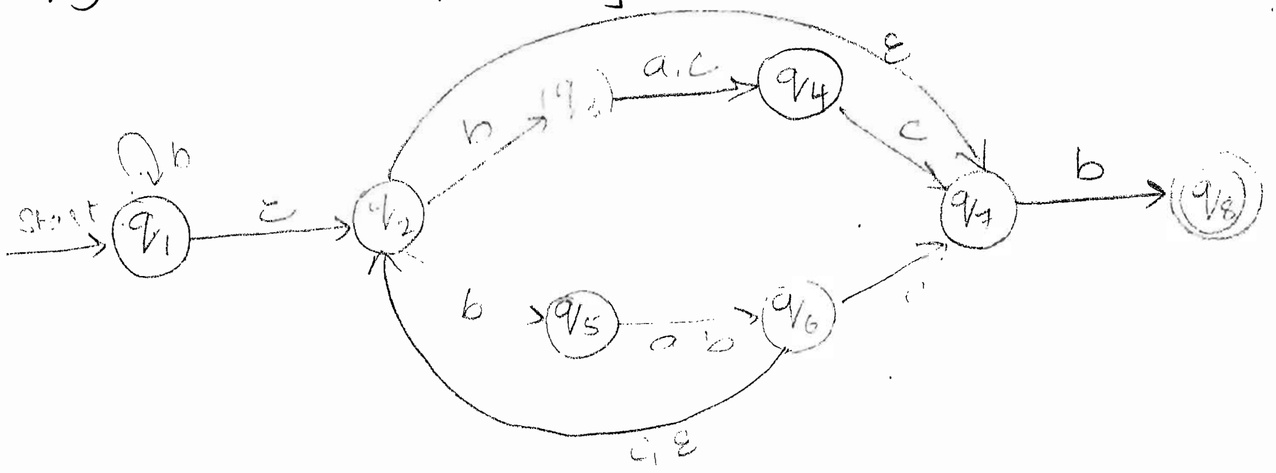
$\therefore$ The resulting DFA,



δ_D	a	b	c
→A	A	B	C
*B	ϕ	B	C
*C	ϕ	ϕ	C

01/09/18

Q3) convert the following ε-NFA to DFA.



Step1: $ECLOSE(q_1) = \{q_1, q_2, q_7\}$

$ECLOSE(q_2) = \{q_2, q_7\}$

$ECLOSE(q_3) = \{q_3\}$

$ECLOSE(q_4) = \{q_4\}$

$ECLOSE(q_5) = \{q_5\}$

$ECLOSE(q_6) = \{q_2, q_6, q_7\}$

$ECLOSE(q_7) = \{q_7\}$

$ECLOSE(q_8) = \{q_8\}$

Step2: Start state

$ECLOSE(q_1) = \{q_1, q_2, q_7\}$ — A

Step 3: compute the transition.

$\delta(A, a) = ECLOSE(\delta_E(\{q_1, q_2, q_7\}, a))$

$= ECLOSE(\phi)$

$\therefore \boxed{\delta(A, a) = \phi}$

$\delta(A, b) = ECLOSE(\delta_E(\{q_1, q_2, q_7\}, b))$

$= ECLOSE((\{q_1\} \cup \{q_3, q_5\} \cup \{q_8\}))$

$= \{q_1, q_2, q_3, q_5, q_7, q_8\}$ B.

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

$\delta(A,c) = ECLOSE(\delta_E(\{q_1, q_2, q_7\}, c))$

$= \phi$

$\therefore \boxed{\delta(A,c) = \phi}$

consider B

$\delta(B,a) = ECLOSE(\delta_E(\{q_1, q_2, q_3, q_5, q_7, q_8\}, a))$

$= ECLOSE(\{q_4\} \cup \{q_6\})$

$= \{q_2, q_4, q_6, q_7\}$ — C

$\therefore \boxed{\delta(B,a) = C}$

$\delta(B,b) = ECLOSE(\delta_E(\{q_1, q_2, q_3, q_5, q_7, q_8\}, b))$

$= ECLOSE(\{q_1\} \cup \{q_3\} \cup \{q_6\} \cup \{q_8\})$

$= \{q_1, q_2, q_3, q_5, q_6, q_7, q_8\}$ — D

$\therefore \boxed{\delta(B,b) = D}$

$\delta(B,c) = ECLOSE(\delta_E(\{q_1, q_2, q_3, q_5, q_7, q_8\}, c))$

$= ECLOSE(\{q_4\} \cup \phi)$

$= \{q_4\}$ — E

$\therefore \boxed{\delta(B,c) = E}$

consider C : (q_2, q_4, q_6, q_7)

$\delta(C,a) = ECLOSE(\delta_E(\{q_2, q_4, q_6, q_7\}, a))$

$= \phi$

$\therefore \boxed{\delta(C,a) = \phi}$

$\delta(C,b) = ECLOSE(\{q_3\} \cup \{q_8\})$

$= \{q_3, q_5, q_8\}$ — F

$\therefore \boxed{\delta(C,b) = F}$

$\delta(C,c) = ECLOSE(\{q_7\} \cup \{q_2, q_7\} \cup \phi)$

$= \{q_2, q_7\}$ — G

$\therefore \boxed{\delta(C,c) = G}$

Consider D : $D \rightarrow \{q_1, q_2, q_3, q_5, q_6, q_7, q_8\}$

$\delta(D,a) = ECLOSE(\{q_4\} \cup \{q_6\})$

$= \{q_2, q_4, q_6, q_7\}$ — C

$\therefore \boxed{\delta(D,a) = C}$

$\delta(D,b) = ECLOSE(\{q_1\} \cup \{q_3, q_5\} \cup \{q_6\} \cup \{q_8\})$

$= \{q_1, q_2, q_3, q_5, q_6, q_7, q_8\}$ — D

$\therefore \boxed{\delta(D,b) = D}$

$\delta(D,c) = ECLOSE(\{q_4\} \cup \{q_2, q_7\})$

$= \{q_2, q_4, q_7\}$ — H

$\therefore \boxed{\delta(D,c) = H}$

Consider E : $E \rightarrow \{q_4\}$

$\delta(E,a) = ECLOSE(\phi)$

$= \phi$

$\therefore \boxed{\delta(E,a) = \phi}$

$\delta(E,b) = ECLOSE(\phi)$

$= \phi$

$\therefore \boxed{\delta(E,b) = \phi}$

$\delta(E,c) = ECLOSE(\{q_7\})$

$= \{q_2, q_7\}$ — G

$\therefore \boxed{\delta(E,c) = G}$

Consider F : $F \rightarrow \{q_3, q_6, q_8\}$

$\delta(F,a) = ECLOSE(\{q_4\} \cup \{q_6\})$

$= \{q_2, q_4, q_6, q_7\}$ — C

$\therefore \boxed{\delta(F,a) = C}$

$\delta(F,b) = ECLOSE(\{q_6\})$

$= \{q_2, q_6, q_7\}$ — D

$\therefore \boxed{\delta(F,b) = D}$

$\delta(F,c) = ECLOSE(\{q_4\})$

$= \{q_4\}$ — E

$\therefore \boxed{\delta(F,c) = E}$

Consider G : $G \rightarrow \{q_2, q_7\}$

$\delta(G,a) = ECLOSE(\phi)$

$= \phi$

$\therefore \boxed{\delta(G,a) = \phi}$

$\delta(G,b) = ECLOSE(\{q_3, q_6\} \cup \{q_8\})$

$= \{q_3, q_6, q_8\}$ — F

$\therefore \boxed{\delta(G,b) = F}$

$\delta(G,c) = ECLOSE(\phi)$

$\therefore \boxed{\delta(G,c) = \phi}$

Consider H : $H \rightarrow \{q_2, q_4, q_7\}$

$\delta(H,a) = ECLOSE(\phi)$

$\therefore \boxed{\delta(H,a) = \phi}$

$\delta(H,b) = ECLOSE(\{q_3, q_5\} \cup \{q_8\})$

$= \{q_3, q_5, q_8\}$ — F

$\therefore \boxed{\delta(H,b) = F}$

$\delta(H, c) = ECLOSE(\{q_7\})$

$= \{q_2, q_7\}$ — G

$\therefore \boxed{\delta(H, c) = G}$

Consider I : $I \rightarrow \{q_2, q_6, q_7\}$

$\delta(I, a) = ECLOSE(\phi)$

$\therefore \boxed{\delta(I, a) = \phi}$

$\delta(I, b) = ECLOSE(\{q_3, q_5\} \cup \{q_8\})$

$= \{q_3, q_5, q_8\}$ — F

$\therefore \boxed{\delta(I, b) = F}$

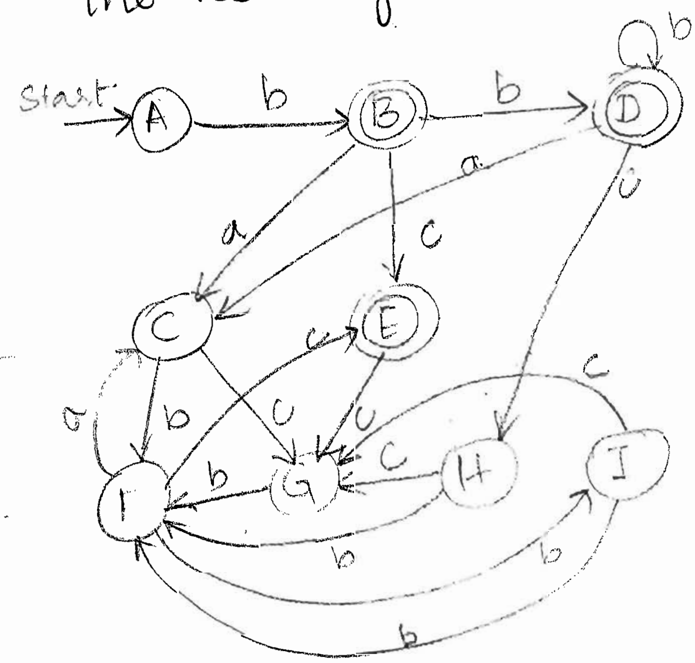
$\delta(I, c) = ECLOSE(\{q_2, q_7\})$

$= \{q_2, q_7\}$ — G

$\therefore \boxed{\delta(I, c) = G}$

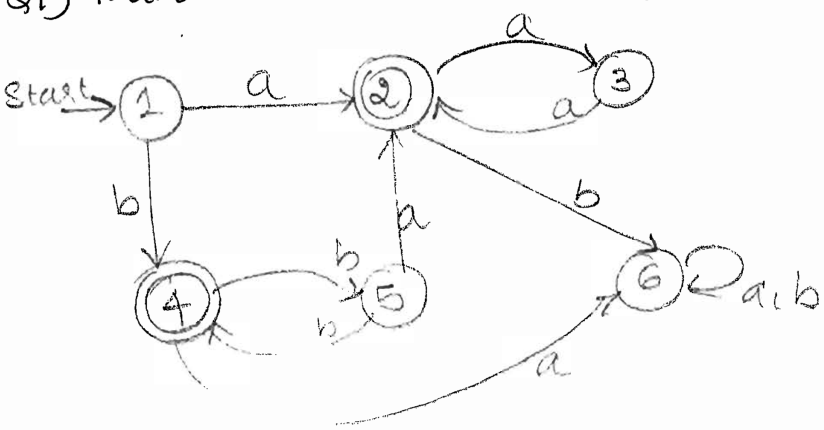
δ_D	a	b	c
→ A	ϕ	B	ϕ
* B	C	D	E
C	ϕ	F	G
* D	C	D	H
E	ϕ	ϕ	G
* F	C	I	E
G	ϕ	F	ϕ
H	ϕ	F	G
I	ϕ	F	G

The resulting DFA is,



Minimizing FSM's

Q1) Minimize the following DFA.



Initial class {[2,4], [1,3,5,6]}

Step 1:

((2,a), [1,3,5,6]) ((1,a), [2,4])
((2,b), [1,3,5,6]) ((1,b), [2,4])
((4,a), [1,3,5,6]) ((3,a), [2,4])
((4,b), [1,3,5,6]) ((3,b), [2,4])
((5,a), [2,4])
((5,b), [2,4])
((6,a), [1,3,5,6])
((6,b), [1,3,5,6])

∴ The resulting class {[2,4], [1,3,5], [6]}

Step 2:

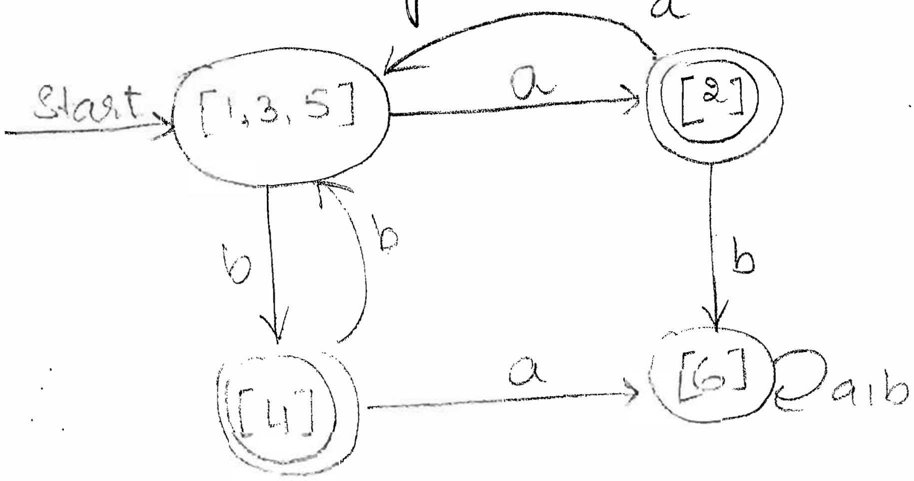
((2,a), [1,3,5]) ((1,a), [2,4])
((2,b), [6]) ((1,b), [2,4])
((4,a), [6]) ((3,a), [2,4])
((4,b), [1,3,5]) ((3,b), [2,4])
((5,a), [2,4])
((5,b), [2,4])
((6,a), [6])
((6,b), [6])

∴ The resulting class {[2], [4], [1,3,5], [6]}

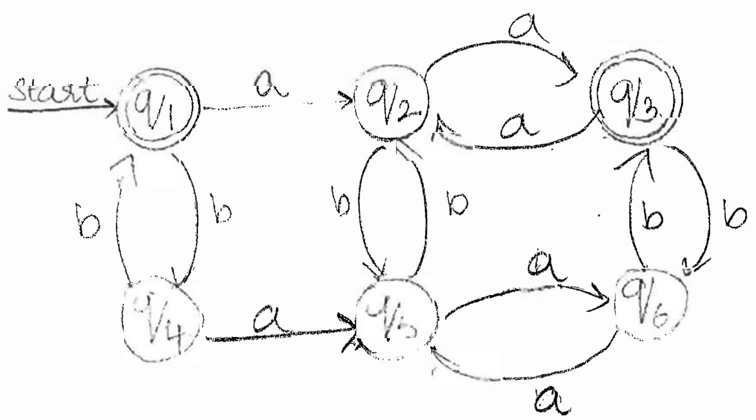
step 3: ((1,a), [2])
((1,b), [4])
((3,a), [2])
((3,b), [4])
((5,a), [2])
((5,b), [4])

∴ The resulting class {[2], [4], [1,3,5], [6]}

∴ The resulting DFA is,



Q2) Minimize the following DFA.



Initial class $\{[q_1, q_3], [q_2, q_4, q_5, q_6]\}$

Step2: $((q_0, a), [q_1, q_2, q_3])$

$((q_0, b), [q_1, q_2, q_3])$

$((q_1, a), [q_1, q_2, q_3])$

$((q_1, b), [q_4])$

$((q_2, a), [q_1, q_2, q_3])$

$((q_2, b), [q_4])$

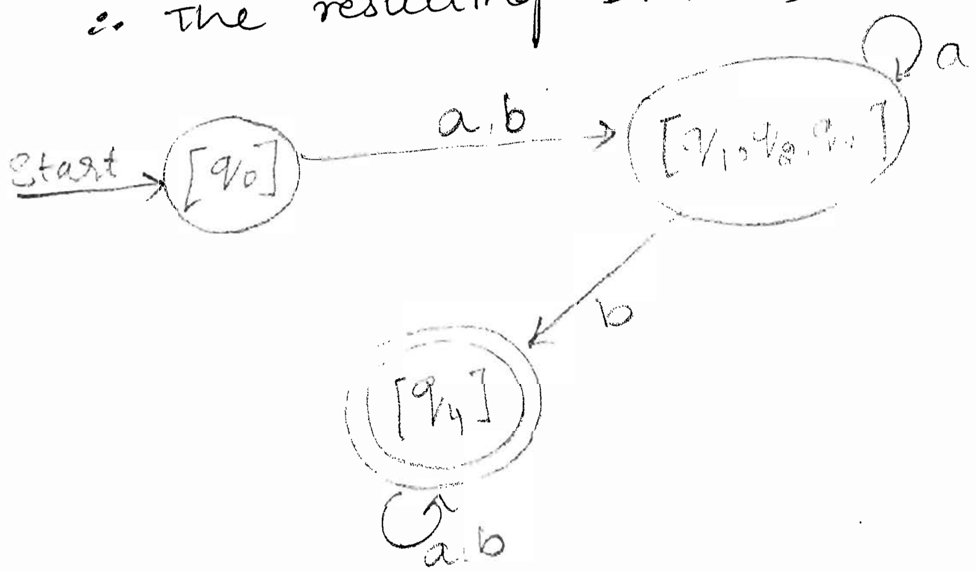
$((q_3, a), [q_1, q_2, q_3])$

$((q_3, b), [q_4])$

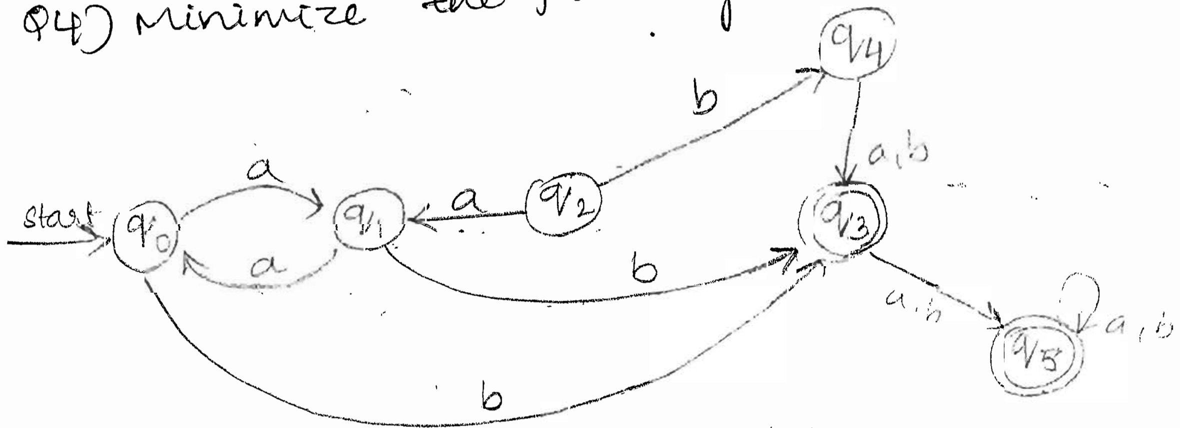
$\therefore$ the resulting class is,

$\{[q_0], [q_1, q_2, q_3], [q_4]\}$

$\therefore$ The resulting DFA is,



Q4) Minimize the following DFA.



\# q_2, q_4 are non reachable states.

\# q_0 is start state

\# q_0, q_1 non final states

Initial class $\{[q_0, q_1], [q_3, q_5]\}$

Step 1:

$((q_0, a), [q_0, q_1])$ $((q_3, a), [q_3, q_5])$

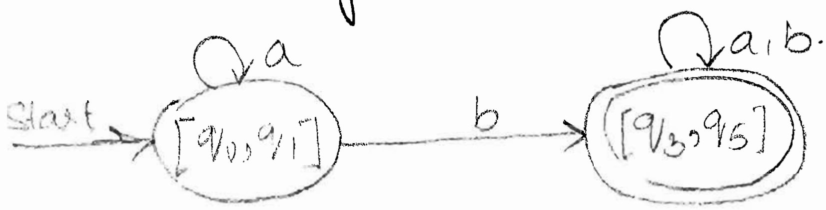
$((q_0, b), [q_3, q_5])$ $((q_3, b), [q_3, q_5])$

$((q_1, a), [q_0, q_1])$ $((q_5, a), [q_3, q_5])$

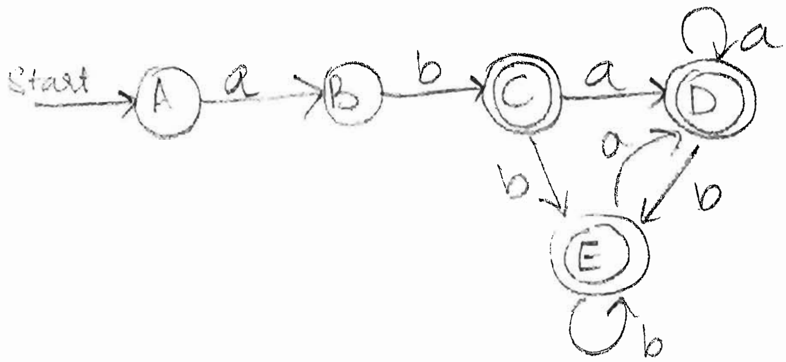
$((q_1, b), [q_3, q_5])$ $((q_5, b), [q_3, q_5])$

$\therefore$ The resulting class is $\{[q_0, q_1], [q_3, q_5]\}$

$\therefore$ Resulting DFA is,



Q5) Minimize the following DFA:-



Initial class $\{[A, B], [C, D, E]\}$

Step 1:

$((A, a), [A, B])$ $((C, a), [C, D, E])$

$((A, b), -)$ $((C, b), [C, D, E])$

$((B, a), -)$ $((D, a), [C, D, E])$

$((B, b), [C, D, E])$ $((D, b), [C, D, E])$

$((E, a), [C, D, E])$

$((E, b), [C, D, E])$

$\therefore$ The resulting class $\{[A], [B], [C, D, E]\}$

step1: $((q_1,a),[q_2,q_4,q_5,q_6])$

$((q_1,b),[q_2,q_4,q_5,q_6])$

$((q_3,a),[q_2,q_4,q_5,q_6])$

$((q_3,b),[q_2,q_4,q_5,q_6])$

$((q_2,a),[q_1,q_3])$

$((q_2,b),[q_2,q_4,q_5,q_6])$

$((q_4,a),[q_2,q_4,q_5,q_6])$

$((q_4,b),[q_1,q_3])$

$((q_5,a),[q_2,q_4,q_5,q_6])$

$((q_5,b),[q_2,q_4,q_5,q_6])$

$((q_6,a),[q_2,q_4,q_5,q_6])$

$((q_6,b),[q_1,q_3])$

∴ The resulting class

$\{[q_1,q_3],[q_2],[q_5],[q_4,q_6]\}$

step2: $((q_1,a),[q_2])$

$((q_1,b),[q_4,q_6])$

$((q_3,a),[q_2])$

$((q_3,b),[q_4,q_6])$

$((q_2,a),[q_1,q_3])$

$((q_2,b),[q_5])$

$((q_4,a),[q_5])$

$((q_4,b),[q_1,q_3])$

$((q_5,a),[q_4,q_6])$

$((q_5,b),[q_2])$

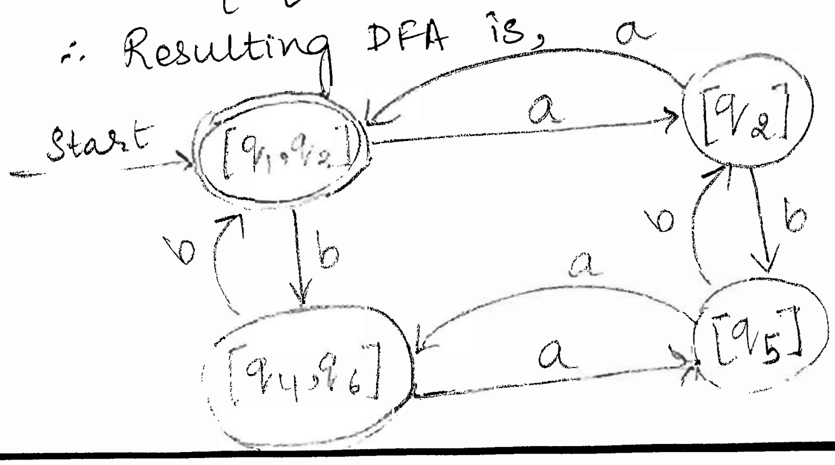
$((q_6,a),[q_5])$

$((q_6,b),[q_1,q_3])$

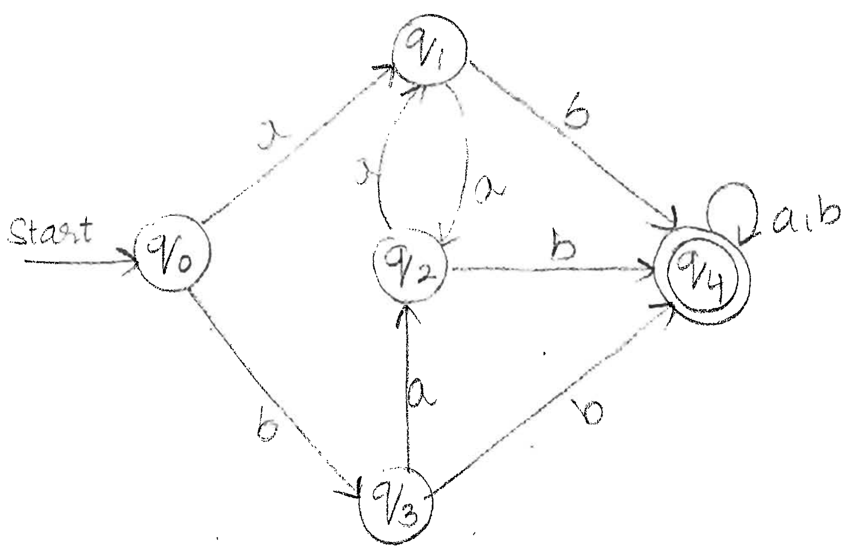
∴ The resulting class is,

$\{[q_1,q_3],[q_2],[q_5],[q_4,q_6]\}$

∴ Resulting DFA is,



31 Q3) Minimize the following DFA.



Initial class $\{[q_0, q_1, q_2, q_3], [q_4]\}$

Step1:

$((q_0, a), [q_0, q_1, q_2, q_3])$

$((q_0, b), [q_0, q_1, q_2, q_3])$

$((q_1, a), [q_0, q_1, q_2, q_3])$

$((q_1, b), [q_4])$

$((q_2, a), [q_0, q_1, q_2, q_3])$

$((q_2, b), [q_4])$

$((q_3, a), [q_0, q_1, q_2, q_3])$

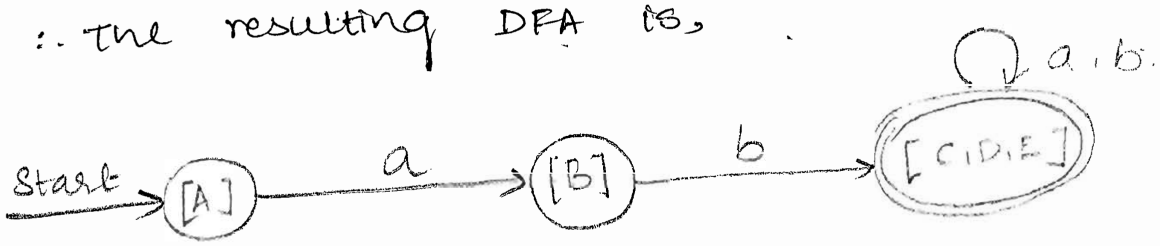
$((q_3, b), [q_4])$

$((q_4, a), [q_4])$

$((q_4, b), [q_4])$

$\therefore$ The resulting class is $\{[q_0], [q_1, q_2, q_3], [q_4]\}$

∴ The resulting DFA is,



Q6) Minimize the following DFA.

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

Solution

Initial classes {[A, B, D, E, G, H], [C, F, I]}

Step 1:

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

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

∴ resulting classes are {[A, D, G], [B, E, H], [C, F, I]}

Step 2:

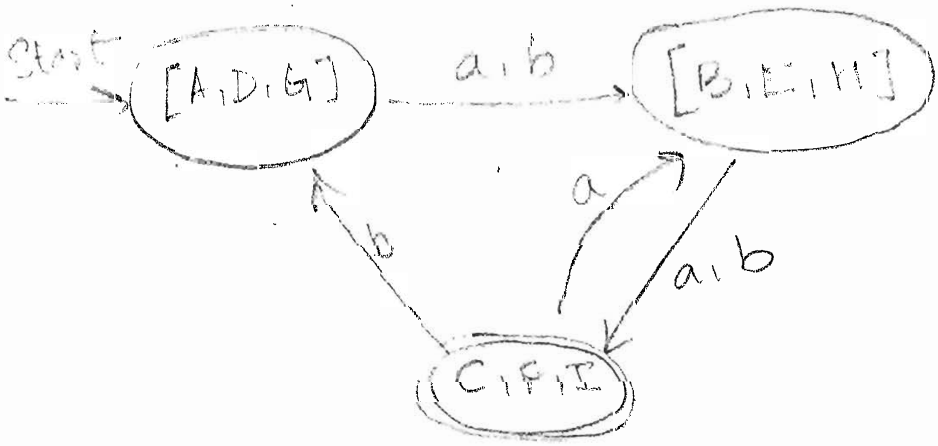
((A,0), [B,E,H])
((A,1), [B,E,H])
((D,0), [B,E,H])
((D,1), [B,E,H])
((G,0), [B,E,H])
((G,1), [B,E,H])

((B,0), [C,F,I])
((B,1), [C,F,I])
((E,0), [C,F,I])
((E,1), [C,F,I])
((H,0), [C,F,I])
((H,1), [C,F,I])

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

∴ Resulting classes are {[A,D,G], [B,E,H], [C,F,I]}

∴ Resulting DFA is,



Conversion from ε-NFA to DFA.

01/09/18

Q1) Consider the following ε-NFA

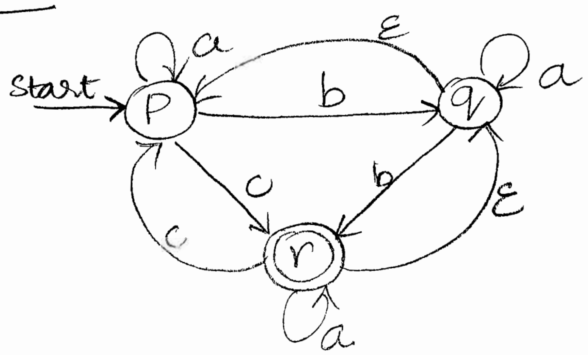
δ	ε	a	b	c
→ p	φ	{p}	{q}	{r}
q	{p}	{q}	{r}	φ
* r	{q}	{r}	φ	{p}

(a) Compute the ECLOSURE of each state.

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

(c) Convert the automata to DFA.

Soln:



(a) ECLOSE(p) = {p}

ECLOSE(q) = {p, q}

ECLOSE(r) = {p, q, r}

(b) c, bb, bc, cb and cc (length 2 or less)

bbb, bbc, bcb, bcc, cbb, cbc, ccb, ccc (length 3)

(c) start state

ECLOSE (p) = {p} — A.

consider A.

$$\delta(A,a) = \text{ECLOSE}(\delta_E(\{p\},a))$$
$$= \text{ECLOSE}(\{p\})$$
$$= \{p\} \text{ — } A$$
$$\therefore \boxed{\delta(A,a) = A}$$

$$\delta(A,b) = \text{ECLOSE}(\delta_E(\{p\},b))$$
$$= \text{ECLOSE}(\{q\})$$
$$= \{p,q\} \text{ — } B$$
$$\therefore \boxed{\delta(A,b) = B}$$

$$\delta(A,c) = \text{ECLOSE}(\delta_E(\{p\},c))$$
$$= \text{ECLOSE}(\{r\})$$
$$= \{p,q,r\} \text{ — } C$$
$$\therefore \boxed{\delta(A,c) = C}$$

consider B :

$$\delta(B,a) = \text{ECLOSE}(\delta_E(\{p,q,r\},a))$$
$$= \text{ECLOSE}(\{p\} \cup \{q\} \cup \{r\})$$
$$= \{p,q,r\} \text{ — } C$$
$$\therefore \boxed{\delta(B,a) = C}$$

$$\delta(B,b) = \text{ECLOSE}(\delta_E(\{p,q,r\},b))$$
$$= \text{ECLOSE}(\{q\} \cup \{r\} \cup \phi)$$
$$= \{p,q,r\} \text{ — } C$$
$$\therefore \boxed{\delta(B,b) = C}$$

$$\delta(B,c) = \text{ECLOSE}(\delta_E(\{p,q,r\},c))$$
$$= \text{ECLOSE}(\{r\} \cup \phi \cup \{p\})$$
$$= \{p,q,r\} \text{ — } C \Rightarrow \boxed{\delta(B,c) = C}$$

$\therefore$ ~~$\delta(B,C$~~

consider C

$$\delta(C, a) = \text{ECLOSE}(\delta_E(\{p, q, r\}, a))$$
$$= \text{ECLOSE}(\{p\} \cup \{q\} \cup \{r\})$$
$$= \{p, q, r\} - C$$

$\therefore \boxed{\delta(C, a) = C}$

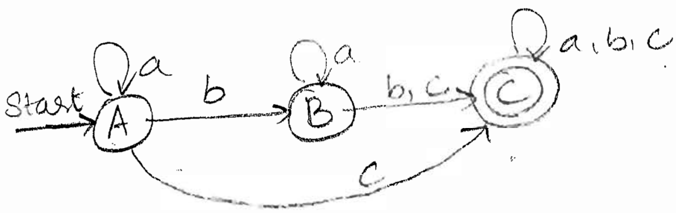
$$\delta(C, b) = \text{ECLOSE}(\delta_E(\{p, q, r\}, b))$$
$$= \text{ECLOSE}(\{q\} \cup \{r\} \cup \phi)$$
$$= \{p, q, r\} - C$$

$\therefore \boxed{\delta(C, b) = C}$

$$\delta(C, c) = \text{ECLOSE}(\delta_E(\{p, q, r\}, c))$$
$$= \text{ECLOSE}(\{r\} \cup \{\Phi\})$$
$$= \{p, q, r\} - C$$

$\therefore \boxed{\delta(C, c) = C}$

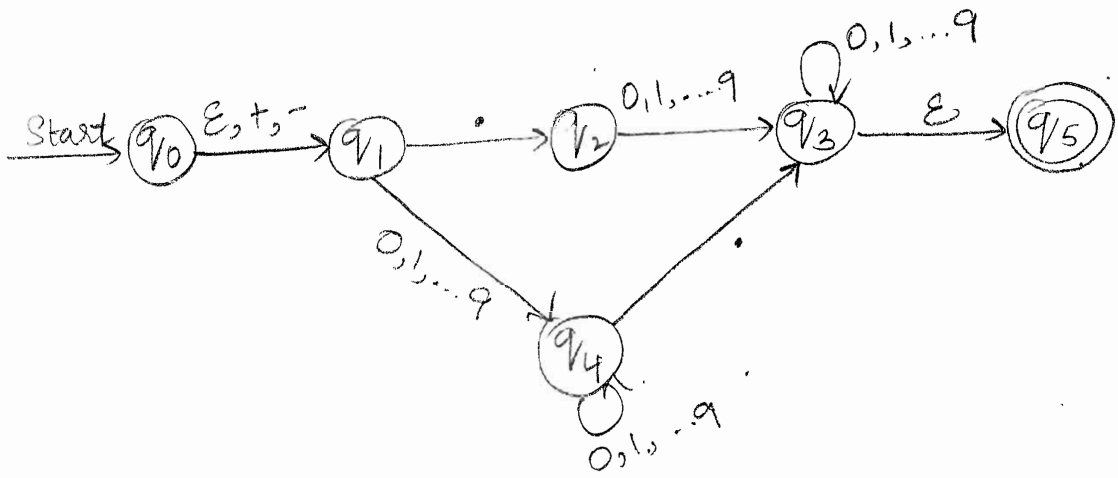
δ	a	b	c
$\rightarrow$ A	A	B	C
B	B	C	C
* C	C	C	C



*** Q2) obtain an NFA with ε-transitions (ε-NFA) to accept decimal numbers and convert into DFA and obtain $\delta^*(q_0, 4.7)$

Ans: A decimal number has
(i) An optional '+' or '-' sign followed by '.' followed by string of digits
(ii) An optional '+' or '-' sign followed by a string of digits followed by a '.' and followed by string of digits.

$\therefore$ The ε-NFA is shown as follows



Step 1: $ECLOSE(q_0) = \{q_0, q_1\}$

$ECLOSE(q_1) = \{q_1\}$

$ECLOSE(q_2) = \{q_2\}$

$ECLOSE(q_3) = \{q_3, q_5\}$

$ECLOSE(q_4) = \{q_4\}$

$ECLOSE(q_5) = \{q_5\}$

Step 2: Start state

$ECLOSE(q_0) = \{q_0, q_1\}$ — A

consider A

$\delta(A, \pm) = ECLOSE(\delta_E(\{q_0, q_1\}, \pm))$

$= ECLOSE(\{q_1\})$

$= \{q_1\}$ — B

$\therefore \boxed{\delta(A, \pm) = B}$

$\delta(A, \cdot) = ECLOSE(\delta_E(\{q_0, q_1\}, \cdot))$

$= ECLOSE(\{q_2\})$

$= \{q_2\}$ — C

$\therefore \boxed{\delta(A, \cdot) = C}$

$$\delta(A, \{0 \cdots 9\}) = ECLOSE(\delta_E(\{q_0, q_1\}, \{0, 1 \cdots 9\}))$$
$$= ECLOSE(\{q_4\})$$
$$= \{q_4\} \text{ — } D$$

$\therefore$ $\boxed{\delta(A, \{0 \cdots 9\}) = D}$

consider B

$$\delta(B, \pm) = ECLOSE(\delta_E(\{q_1\}, \pm))$$
$$= \phi$$

$\therefore$ $\boxed{\delta(B, \pm) = \phi}$

$$\delta(B, \cdot) = ECLOSE(\delta_E(\{q_1\}, \cdot))$$
$$= \{q_2\} \rightarrow C$$

$\therefore$ $\boxed{\delta(B, \cdot) = C}$

$$\delta(B, d) = ECLOSE(\delta_E(\{q_1\}, d))$$
$$= ECLOSE(\{q_4\})$$
$$= \{q_4\} \text{ — } D$$

$\therefore$ $\boxed{\delta(B, d) = D}$

consider C

$$\delta(C, \pm) = ECLOSE(\delta_E(\{q_2\}, \pm))$$
$$= \phi$$

$\therefore$ $\boxed{\delta(C, \pm) = \phi}$

$$\delta(C, \cdot) = ECLOSE(\delta_E(\{q_2\}, \cdot))$$
$$= \phi$$

$$\delta(C, \text{digits}) = ECLOSE(\delta_E(\{q_2\}, \text{digits}))$$
$$= ECLOSE(\{q_3\})$$
$$= \{q_3, q_5\} \text{ — } E$$

$\boxed{\delta(C, \text{digits}) = E}$

consider D

$\delta(D, \pm) = ECLOSE(\delta_E(\{q_4\}, \pm))$

$= \phi$

$\boxed{\delta(D, \pm) = \phi}$

$\delta(D, \cdot) = ECLOSE(\delta_E(\{q_4\}, \cdot))$

$= ECLOSE(\{q_3\})$

$= \{q_3, q_5\}$ — E

$\boxed{\delta(D, \cdot) = E}$

$\delta(D, d) = ECLOSE(\delta_E(\{q_4\}, d))$

$= ECLOSE(\{q_4\}) = \{q_4\}$ — D

$\boxed{\delta(D, d) = D}$

consider E

$\delta(E, \pm) = ECLOSE(\delta_E(\{q_3, q_5\}, \pm))$

$= \phi$

$\therefore \boxed{\delta(E, \pm) = \phi}$

$\delta(E, \cdot) = ECLOSE(\delta_E(\{q_3, q_5\}, \cdot))$

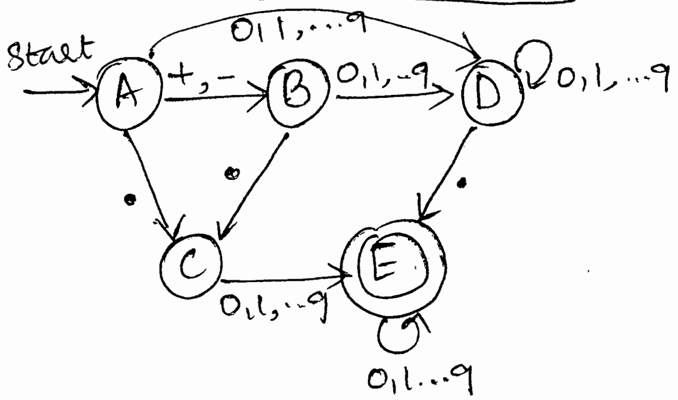
$= \phi$

$\boxed{\delta(E, \cdot) = \phi}$

$\delta(E, d) = ECLOSE(\delta_E(\{q_3, q_5\}, d))$

$= \phi$

$\therefore \boxed{\delta(E, d) = \phi}$



compute,

$\delta^*(q_0, 4 \cdot 7)$

$= \delta(\delta(\delta(A, 4), \cdot), 7)$

$= \delta(\delta(D, \cdot), 7)$

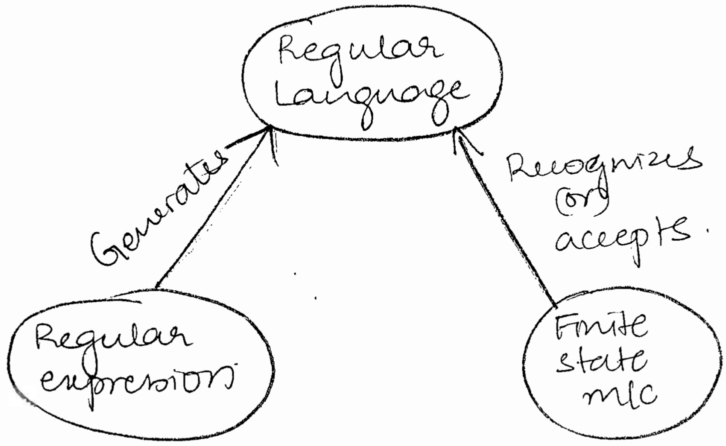
$= \delta(E, 7)$

$= E$

4/9/18

Module - 2

Regular Expression, Regular & Non-Regular Language.



$0^* 1^*$ → regular expression

$0^n 1^n$ → non-regular expression.

Regular Expressions	Meaning
1) a^*	String consistings of any no. of a's (or) strings consisting of zero (or) more a's.
2) a^+	String consisting of 1 'a' (or) string consisting of one (or) more a's.
3) $a+b$	string consisting of a's and b's with either one 'a' (or) one 'b'.
4) $(a+b)^*$	set of strings of a's & b's of any length including null string (ε).
5) $(a+b)^* abb$	set of strings of a's & b's ending with 'abb'.
6) $ab(a+b)^*$	set of strings of a's & b's starting with 'ab'.
7) $(a+b)^* aa (a+b)^*$	set of strings of a's & b's having 'aa' as substring.
8) $a^* b^* c^*$	string consisting of any no. of a's followed by any no. of b's & followed by any no. of c's.

Regular Expression	Meaning.	5/9
$a^+ b^+ c^+$	String consistings of atleast one 'a' followed by atleast one 'b' and followed by atleast one 'c'.	Q1) 06 ho
$aa^* bb^* cc^*$	atleast one 'a' followed by atleast one 'b' followed by atleast one 'c'.	
$(a+b)^*(a+bb)$	strings of a's and b's which are ending with one 'a' (or) two b's.	Q2) of
$(aa)^*$	Set of strings consisting of even no of a's. (or) Set of strings having length which are multiples of 2.	Q3)
$(aa)^*(bb)^* b$	even no. of a's followed by odd no. of b's.	
$(0+1)^* 000$	strings of 0's and 1's ending with 3 0's.	Q4
$01^* + 1$	zero followed by any no. of 1's (or) one no of 1.	
8 $(01)^* + 1$	any no. of 0's and 1's (or) single one.	Q5
$0(1^*+1) = 01^* + 01$ ↳ 01^*	zero followed by any no. of 1's (or) zero followed by single one.	
$(a+b)(a+b)$ (or) $aa+ab+ba+bb$ (or) $(a+b)^2$	strings of a's & b's with length 2.	Q6
$(a+b)^* = (a \cup b)^*$		

5/9/18

Q1) Obtain a regular expression, representing strings of a's & b's having length 2.

$\Rightarrow$ $RE = (a+b)(a+b)$

(or) $RE = (a+b)^2$

(or) $RE = aa+ab+ba+bb$

Q2) Obtain a regular expression, to accept strings of a's & b's of length ≤ 2.

$\Rightarrow$ $R.E = (\varepsilon+a+b)^2$

(or) $RE = (\varepsilon+a+b)(\varepsilon+a+b)$

Q3) Obtain a regular expression, to accept strings of a's and b's of length ≤ 10.

$\Rightarrow$ $R.E = (\varepsilon+a+b)^{10}$

Q4) Obtain a regular expression, represing strings of a's & b's having even length.

$\Rightarrow$ $R.E = ((a+b)^2)^*$

(or) $R.E = ((a+b)(a+b))^*$

Q5) Obtain a regular expression, representing strings of a's and b's having odd length.

$\Rightarrow$ $R.E = (a+b)((a+b)^2)^*$

(or) $R.E = (a+b)((a+b)(a+b))^*$

Q6) Obtain a regular expression to accept a language consisting of strings of a's & b's with alternative a's & b's.

$\Rightarrow$ $RE = ((\varepsilon+b)(ab)^*(\varepsilon+a) + (\varepsilon+a)(ba)^*(\varepsilon+b))$

Q7) Obtain a regular expression to accept language consisting of string's of 0's and 1's with atmost one pair of consecutive zeros.

$\Rightarrow$ RE $= 1^* + (01+1)^* 0 (10+1)^* + (01+1)^* 00 (10+1)^*$

Q8) Obtain a regular expression to accept a language containing atleast one 'a' and atleast one 'b', where $\Sigma = \{a, b, c\}$.

$ab + ba$

RE $= (a+b+c)^* a (a+b+c)^* b (a+b+c)^* + (a+b+c)^* b (a+b+c)^* a (a+b+c)^*$

Q9) Obtain a regular expression such that $L(R) = \{ w \mid w \in \{0,1\}^*$ with atleast 3 consecutive 0's $\}$

R.E $= (0+1)^* 000 (0+1)^*$

Q10) Obtain a regular expression to accept strings of a's & b's ending with 'b' and has no substring 'aa'.

RE $= (b+ab)^+$

6/9/18

Q11) Obtain a regular expression to accept strings of 0's and 1's having no two consecutive zeros.

R.E $= (01+1)^* (0+\varepsilon)$

Q12) Obtain a regular expression to accept strings of a's & b's starting with 'a' and ending with 'b'.

R.E) $= a (a+b)^* b$

Q13) Obtain a regular expression to accept strings of a's and b's whose 2nd symbol from right end is 'a'.

$$R.E = (a+b)^* a (a+b)$$

Note: Last but not 3rd symbol is 'a'.

$$R.E = (a+b)^* a (a+b)^2$$

Q14) Obtain a R.E representing strings of a's & b's whose 10th symbol from right end is 'a'.

$$R.E = (a+b)^* a (a+b)^9$$

Q15) Obtain a R.E to accept strings of a's and b's such that 3rd symbol from the right end is 'a' and 4th symbol from the right end is 'b'

$$R.E = (a+b)^* b a (a+b)^2$$

Q16) Obtain a R.E to accept the words with 2 (or) more letters but beginning and ending with same letter where $\Sigma = \{a, b\}$.

$$R.E = a (a+b)^* a + b (a+b)^* b$$

Q17) Obtain a R-E to accept strings of a's and b's whose length is either even (or) multiples of 3 (or) both.

$$R.E = ((a+b)(a+b))^* + ((a+b)(a+b)(a+b))^*$$

(or)

$$R.E = ((a+b)^2)^* + ((a+b)^3)^*$$

Q18) Obtain a R.E to accept strings of a's & b's such that every block of four consecutive symbols contains atleast two a's.

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

Q19) Obtain a R.E for the language, **
$L = \{a^n b^m \mid (n+m) \text{ is even}\}$ odd + odd, even + even $\to$ even

case 1: If n & m are even, then $n+m$ is even.
$R.E = (aa)^* (bb)^*$

case 2: If n & m are odd, then $n+m$ is even
$R.E = a(aa)^* (bb)^* b$

$$\boxed{\therefore RE = (aa)^* (bb)^* + (aa)^* a (bb)^* b}$$

Q20) Obtain a R.E for the language,
$L = \{a^n b^m \mid m \geq 1, n \geq 1, nm \geq 3\}$

$n \times m \geq 3$

n	m
1	3
3	1

case 1: If $m=1$, $n \geq 3$ then $mn \geq 3$.
$R.E = aaaa^* b$

case 2: If $n=1$, $m \geq 3$ then $nm \geq 3$.
$R.E = abbbb^*$

case 3: If $n \geq 2$, $m \geq 2$ then $mn \geq 3$.
$RE = aaa^* bbb^*$

$\therefore$ R.E $= aaaa^*b + abbbb^* + aaa^*bbb^*$

Q21) Obtain a R.E for the language,

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

R.E $= (aa)^*(bb)^*$

Q22) Obtain a R.E for strings of a's & b's containing not more than 3 a's. $a \leq 3$

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

Q23) Obtain a R.E for the language,

$L = \{a^n b^m \mid n \geq 4, m \leq 3\}$

R.E $= aaaaa^*(b+\varepsilon)^3$ (or) R.E $= a^4a^*(b+\varepsilon)^3$

Q24) Obtain R.E for strings of a's and b's whose lengths are multiples of 3.

R.E $= ((a+b)^3)^*$

Q25) Obtain a R.E for the language,

$L = \{w \cdot |w| \bmod 3 = 0$ where $w \in \{a,b\}^*\}$

R.E $= ((a+b)^3)^*$

Q26) Obtain a R.E for the language,

$L = \{w : n_a(w) \bmod 3 = 0$, where $w \in \{a,b\}^*\}$

R.E $= (b^*ab^*ab^*ab^*)^*$

Q27) Obtain a R.E for the set of all strings that do not end with 01 over $\{0,1\}^*$.

$$R.E = (0+1)^*(00+10+11)$$

Q28) Obtain a R.E for the language,

$$L = \{vuv : u, v \in \{a,b\}^* \text{ and } |v| = 2\}$$

$$R.E = aa(a+b)^*aa + ab(a+b)^*ab + ba(a+b)^*ba + bb(a+b)^*bb$$

Q29) Give the R.E for the following languages on $\Sigma = \{a,b,c\}$.

a) all strings contains exactly one a

b) all strings contains no more than 3 a's.

c) all strings that contain atleast one occurance of each symbol in Σ.

Soln:

a) $R.E = (b+c)^*a(b+c)^*$

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

c) $R.E = (a+b+c)^*a(a+b+c)^*b(a+b+c)^*c(a+b+c)^* +$
$(a+b+c)^*a(a+b+c)^*c(a+b+c)^*b(a+b+c)^* +$
$(a+b+c)^*b(a+b+c)^*a(a+b+c)^*c(a+b+c)^* +$
$(a+b+c)^*b(a+b+c)^*c(a+b+c)^*a(a+b+c)^* +$
$(a+b+c)^*c(a+b+c)^*a(a+b+c)^*b(a+b+c)^* +$
$(a+b+c)^*c(a+b+c)^*b(a+b+c)^*a(a+b+c)^*$

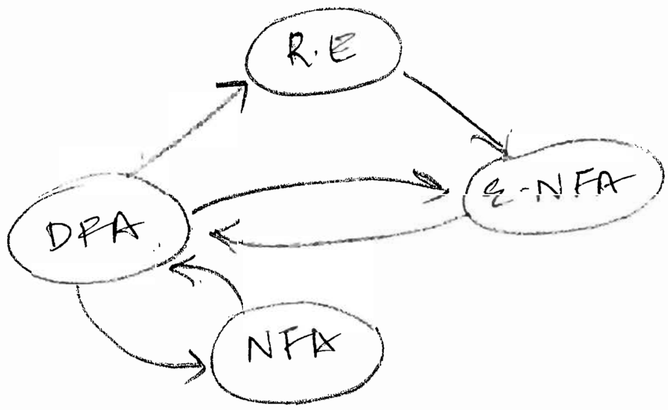
Q30) Obtain a regular expression for the language,
L = { w : string ends with ab or ba where $w \in \{a,b\}^*$ }

$$R.E = (a+b)^* (ab+ba)$$

Q31) Obtain a R.E for the language,
L = { $w \in \{a,b\}^*$: w contains an odd no. of a's }

$$R.E = (b^* a b^* a b^*)^* b^* a b^*$$

Finite Automata & Regular Expression



To obtain ε-NFA from Regular Expression

***** **Theorem:** Let R be a R.E, then there exists a finite automata $M = (Q, \Sigma, \delta, q_0, F)$ which accepts L(R).

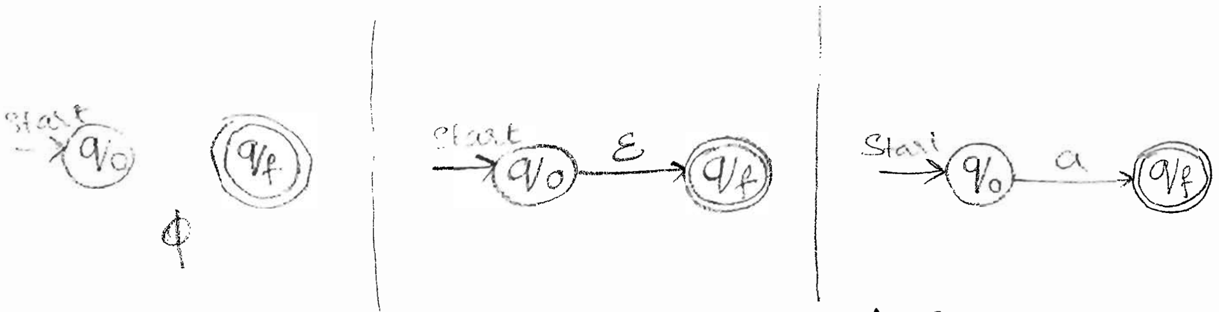
(OR)

Prove that there exists a finite automata to accept the language L(R) corresponding to R.E, R.

(OR)

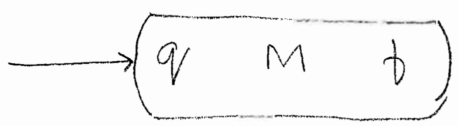
Any language that can be defined with R.E can be accepted by some FSM and so is regular.

Proof: By definition, ϕ, ε and a are regular expressions. So the corresponding machines to recognize the language for the respective expressions are shown in figure below.



ε-NFA's to accept ϕ, ε and a

The Schemantic representation of R.E, R to accept the language $L(R)$ is shown in the figure. where
q is the start state
f is the final state of machine M.



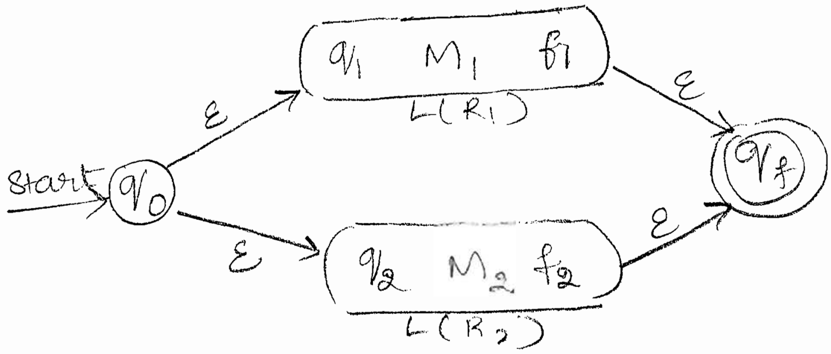
Schemantic Representation of FA accepting $L(R)$

Let $M_1 = (Q_1, \Sigma_1, \delta_1, q_1, f_1)$ be the machine which accepts the language $L(R_1)$ corresponding to R.E, R_1.
Let $M_2 = (Q_2, \Sigma_2, \delta_2, q_2, f_2)$ be the machine which accepts the language $L(R_2)$ corresponding to R.E, R_2.
Then, the various machines corresponding to the regular expressions, $R_1 \overset{\text{or}}{+} R_2$
$R_1 \cdot R_2$ concatenation
R_1^* closure are as shown below.

case(1): $R = R_1 + R_2$

We can construct an NFA which accepts either $L(R_1)$ or $L(R_2)$. which can be represented as, $L(R_1 + R_2)$ as shown in figure below.

To accept the language $L(R_1 + R_2)$

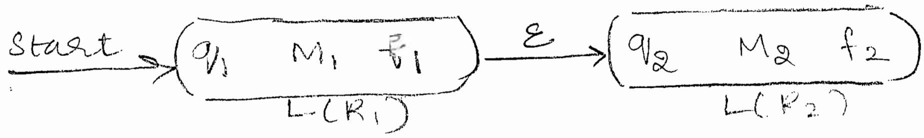


It is clear from the above figure that the machine can either accept $L(R_1)$ (or) $L(R_2)$. Here q_0 is the start state of combained machine & q_f is the final state of combained machine.

case(2): $R = R_1 \cdot R_2$

We can construct an NFA which accepts $L(R_1)$ followed by $L(R_2)$ which can be represented as, $L(R_1 \cdot R_2)$ as shown in the figure below,

To accept the language $L(R_1 \cdot R_2)$



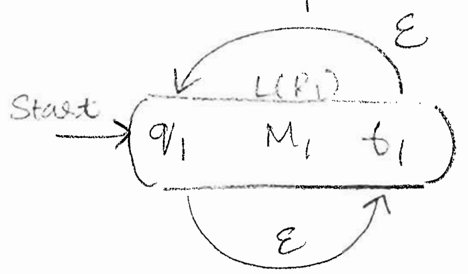
It is clear from the above diagram that the machine after accepting $L(R_1)$ moves from state q_1 to f_1. Since there is ε-transition without any input there will be a transition from f_1 to q_2. In state q_2 upon accepting $L(R_2)$ the machine moves to f_2 which is the final state.

Thus q_1 which is the start state of machine M_1 becomes start state of combained machine M and f_2 which is the final state of M_2 becomes the final state of machine M and accepts the language $L(R_1 \cdot R_2)$.

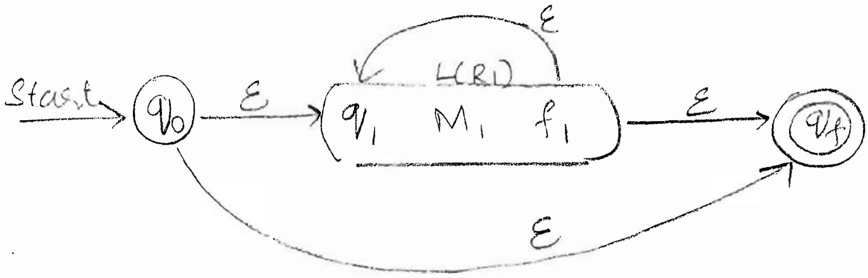
Case (3): $R = (R_1)^*$

We can construct an NFA which accepts $L(R_1)^*$ as shown in figure below,

To accept the language $R = (R_1)^*$



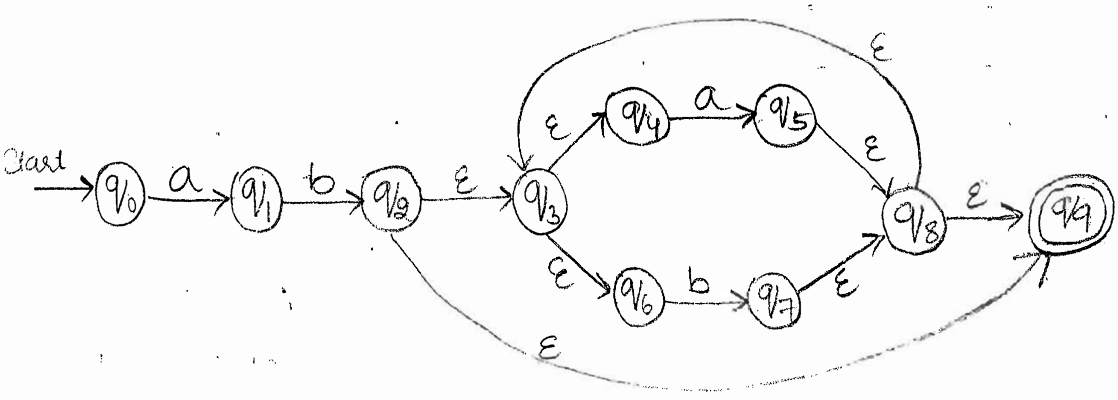
(OR)



It is clear from the above figure that, the machine can either accepts ε (or) any no. of $L(R_1)$'s. Thus accepting language is $L(R_1)^*$. Here, q_0 is the start state and q_f is the final state.

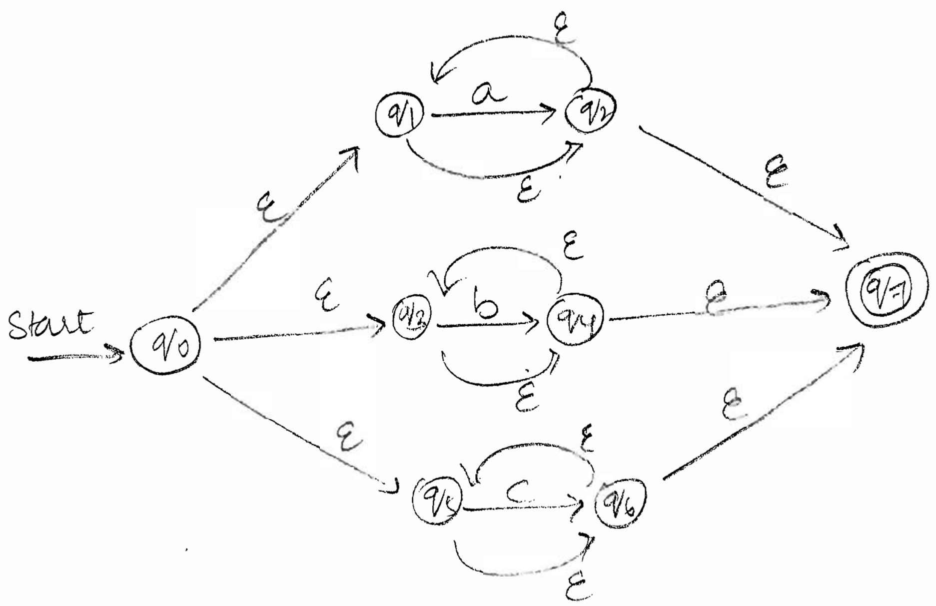
Q1) Obtain an ε-NFA which accepts strings of a's & b's starting with 'ab'.

R.E = $ab(a+b)^*$

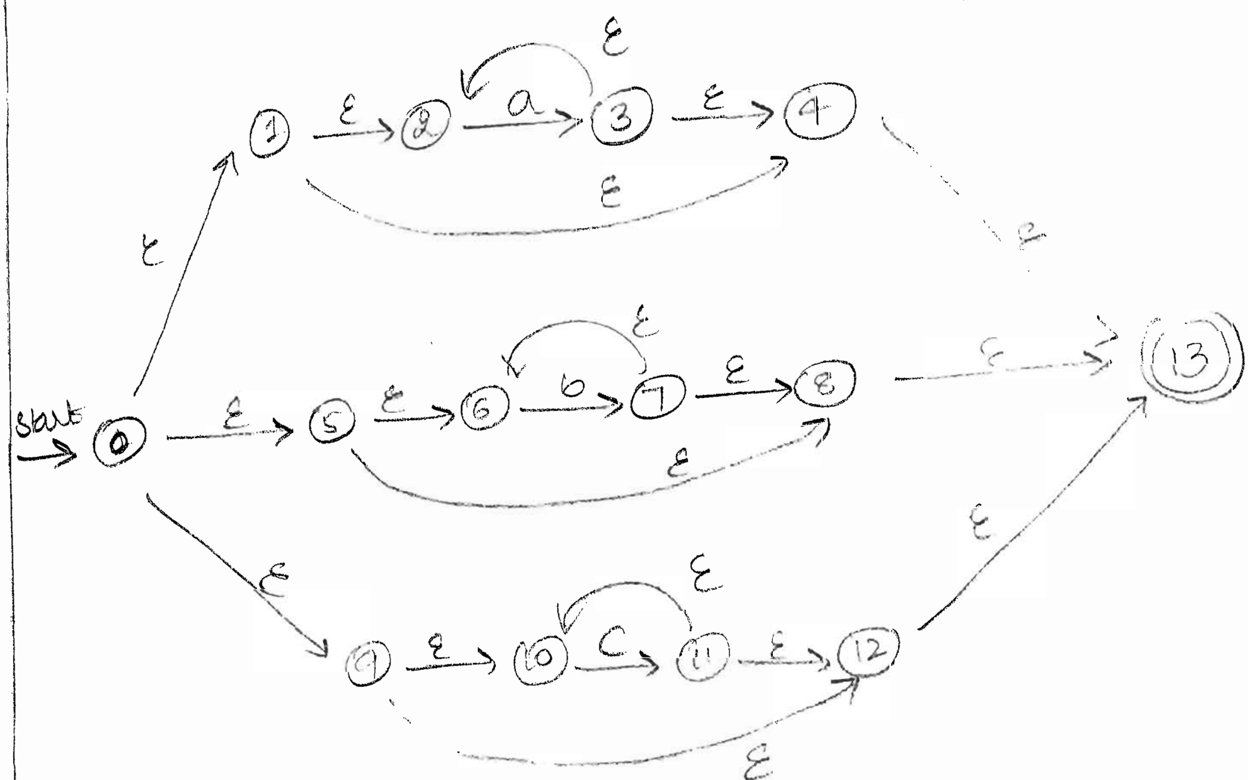


18/9/18

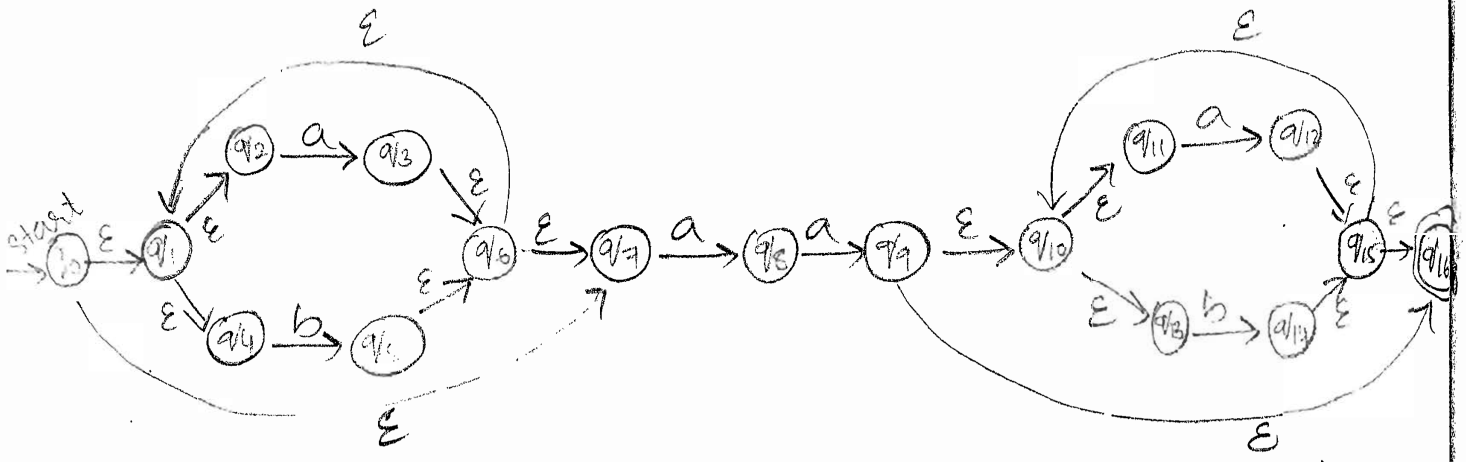
Q2) Obtain NFA for R.E $a^* + b^* + c^*$.



(OR)

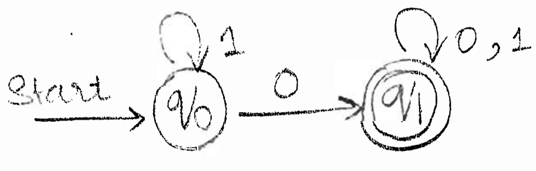


Q3) Obtain an NFA for R.E $(a+b)^* aa (a+b)^*$ (or) $(a \cup b)^* aa (a \cup b)^*$

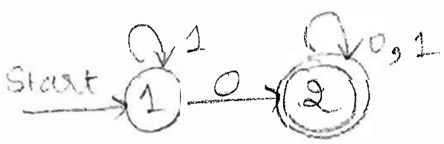


To obtain RE from FA (Kleene's theorem) 19/9/18

Q1) Obtain R.E for FA shown below.



Soln: Let $q_0 = 1$ and $q_1 = 2$. By renaming the states the above FA can be written as shown below,



Note: If the beginning and ending states are same, add ε which denotes the length zero.

Base step:

$R_{11}^{(0)} \Rightarrow (1+\varepsilon)$

$R_{12}^{(0)} \Rightarrow 0$

$R_{21}^{(0)} \Rightarrow \phi$

$R_{22}^{(0)} \Rightarrow (0+1+\varepsilon)$

Induction step:

$$R_{ij}^{(k)} = R_{ij}^{(k-1)} + R_{ik}^{(k-1)} \left[R_{kk}^{(k-1)}\right]^{*} R_{kj}^{(k-1)}$$ Kleene's formula.

when k=1

$$R_{11}^{(1)} = R_{11}^{(0)} + R_{11}^{(0)} \left(R_{11}^{(0)}\right)^{*} R_{11}^{(0)}$$

$$= (1+\varepsilon) + (1+\varepsilon)\,(1+\varepsilon)^{*}\,(1+\varepsilon)$$

q16

$$= (1+\varepsilon) + (1+\varepsilon)\,1^{*}\,(1+\varepsilon)$$

$$= (1+\varepsilon) + 1^{*}$$

$$= \underline{\underline{1^{*}}}$$

$$R_{12}^{(1)} = R_{12}^{(0)} + R_{11}^{(0)} \left(R_{11}^{(0)}\right)^{*} R_{12}^{(0)}$$

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

$$= 0 + 1^{*}0$$

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

$$R_{21}^{(1)} = R_{21}^{(0)} + R_{21}^{(0)} \left(R_{11}^{(0)}\right)^{*} R_{11}^{(0)}$$

$$= \phi + \phi\,(1+\varepsilon)^{*}\,(1+\varepsilon)$$

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

$$R_{22}^{(1)} = R_{22}^{(0)} + R_{21}^{(0)} \left(R_{11}^{(0)}\right)^{*} R_{12}^{(0)}$$

$$= (\varepsilon+0+1) + \phi\,(1+\varepsilon)^{*}\,(0)$$

$$= \underline{\underline{\varepsilon+0+1}}$$

when k=2

$$R_{11}^{(2)} = R_{11}^{(1)} + R_{12}^{(1)} \left(R_{22}^{(1)}\right)^{*} R_{21}^{(1)}$$

$$= 1^{*} + 1^{*}0\,(\varepsilon+0+1)^{*}\,\phi$$

$$= \underline{\underline{1^{*}}}$$

$$R_{12}^{(2)} = R_{12}^{(1)} + R_{12}^{(1)} \left(R_{22}^{(1)}\right)^{*} R_{22}^{(1)}$$

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

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

$R_{21}^{(2)} = R_{21}^{(1)} + R_{22}^{(1)} [R_{22}^{(1)}]^* R_{21}^{(1)}$

$= \phi + (\varepsilon+0+1)(\varepsilon+0+1)^* \phi$

$= \phi$

$R_{22}^{(2)} = R_{22}^{(1)} + R_{22}^{(1)} [R_{22}^{(1)}]^* R_{22}^{(1)}$

$= (\varepsilon+0+1) + (\varepsilon+0+1)(\varepsilon+0+1)^*(\varepsilon+0+1)$

$= (\varepsilon+0+1) + (0+1)^*$

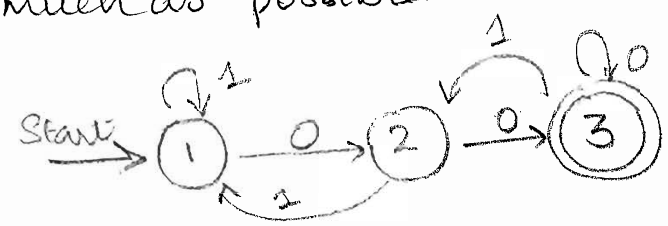
$= (0+1)^*$

$\therefore \boxed{R_{12}^{(2)} = 1^* 0 (1+0)^*}$

Q2) consider DFA shown below, 20/9/18

	0	1
$\rightarrow q_1$	q_2	q_1
q_2	q_3	q_1
$* q_3$	q_3	q_2

Obtain R.E $R_{ij}^{(0)}$, $R_{ij}^{(1)}$ and simplify R.E as much as possible.



Base step:

$R_{11}^{(0)} = \varepsilon + 1$

$R_{12}^{(0)} = 0$

$R_{13}^{(0)} = \phi$

$R_{21}^{(0)} = 1$

$R_{22}^{(0)} = \varepsilon$

$R_{23}^{(0)} = 0$

$R_{31}^{(0)} = \phi$

$R_{32}^{(0)} = 1$

$R_{33}^{(0)} = 0 + \varepsilon$

Induction step:

when $k=1$

$$R_{11}^{(1)} = R_{11}^{(0)} + R_{11}^{(0)} [R_{11}^{(0)}]^* R_{11}^{(0)}$$
$$= (\varepsilon+1) + (\varepsilon+1)(\varepsilon+1)^* (\varepsilon+1)$$
$$= (\varepsilon+1) + (\varepsilon+1)^*$$
$$= \underline{(\varepsilon+1)^*} = \underline{1^*}$$

$$R_{12}^{(1)} = R_{12}^{(0)} + R_{11}^{(0)} [R_{11}^{(0)}]^* R_{12}^{(0)}$$
$$= 0 + (\varepsilon+1)(\varepsilon+1)^* 0$$
$$= \underline{1^* 0}$$

$$R_{13}^{(1)} = R_{13}^{(0)} + R_{11}^{(0)} [R_{11}^{(0)}]^* R_{13}^{(0)}$$
$$= \phi + (1+\varepsilon)(\varepsilon+1)^* \phi$$
$$= \underline{\phi}$$

$$R_{21}^{(1)} = R_{21}^{(0)} + R_{21}^{(0)} [R_{11}^{(0)}]^* R_{11}^{(0)}$$
$$= 1 + 1 (\varepsilon+1)^* (\varepsilon+1)$$
$$= 1 + 1(\varepsilon+1)^*$$
$$= \underline{1 1^*} = 1^+$$

$$R_{22}^{(1)} = R_{22}^{(0)} + R_{21}^{(0)} [R_{11}^{(0)}]^* R_{12}^{(0)}$$
$$= \varepsilon + 1 (\varepsilon+1)^* 0$$
$$= \underline{\varepsilon + 1 1^* 0} \Rightarrow \varepsilon + 1^+ 0$$

$$R_{23}^{(1)} = R_{23}^{(0)} + R_{21}^{(0)} [R_{11}^{(0)}]^* R_{13}^{(0)}$$
$$= 0 + 1 (\varepsilon+1)^* \phi$$
$$= \underline{0}$$

$$R_{31}^{(1)} = R_{31}^{(0)} + R_{31}^{(0)} [R_{11}^{(0)}]^* R_{11}^{(0)}$$
$$= \phi + \phi (\varepsilon+1)^* (\varepsilon+1)$$
$$= \underline{\phi}$$

$$R_{32}^{(1)} = R_{32}^{(0)} + R_{31}^{(0)}[R_{11}^{(0)}]^* R_{12}^{(0)}$$
$$= 1 + \phi(\varepsilon+1)^* 0$$
$$= 1$$

$$R_{33}^{(1)} = R_{33}^{(0)} + R_{31}^{(0)}[R_{11}^{(0)}]^* R_{13}^{(0)}$$
$$= (\varepsilon+0) + \phi(\varepsilon+1)^* \phi$$
$$= \varepsilon+0$$

Now, $R_{13}^{(3)} = R_{13}^{(2)} + R_{13}^{(2)}[R_{33}^{(2)}]^* R_{33}^{(2)}$ — (1)

$$R_{13}^{(2)} = R_{13}^{(1)} + R_{12}^{(1)}[R_{22}^{(1)}] R_{23}^{(1)}$$
$$= \phi + 1^*0(\varepsilon+1+0)^* 0$$
$$= 1^*0(1+0)^*0$$

$$R_{33}^{(2)} = R_{33}^{(1)} + R_{32}^{(1)}[R_{22}^{(1)}]^* R_{23}^{(1)}$$
$$= (\varepsilon+0) + 1(\varepsilon+1+0)^* 0$$
$$= (\varepsilon+0) + 1(1+0)^* 0$$

Now (1),

$$R_{13}^{(3)} = R_{13}^{(2)} + R_{13}^{(2)}[R_{33}^{(2)}]^* R_{33}^{(2)}$$
$$= 1^*0(1+0)^*0 + 1^*0(1+0)^*0[(\varepsilon+0)+1(1+0)^*0]^+$$

$$\therefore R_{13}^{(3)} = 1^*0(1+0)^*0[\varepsilon+((\varepsilon+0)+1(1+0)^*0)^*]$$

To obtain RE from FA (By eliminating states)

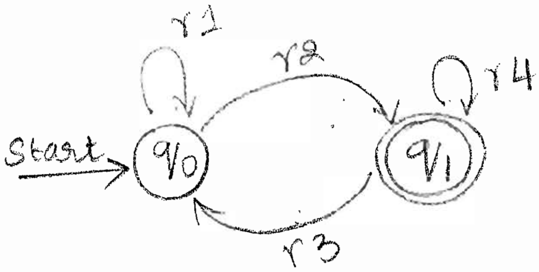


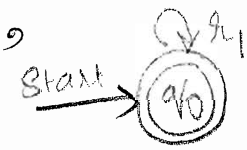
fig: Generalized transition diagram.

where r1, r2, r3, r4 are the R.E & corresponding to the labels for the edges. The regular expressions for this can take the form,

$$R.E = r_1^* r_2 (r_4 + r_3 r_1^* r_2)^*$$

Note: (1) For each final state q apply the reduction procedure & ~~tomng~~ the graph to the form as shown in the above.

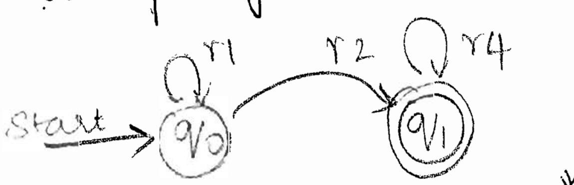
(2) If the start state is also an accepting state, the state elimination process should be performed so that the final automata will be of the form,



$$R.E = r_1^*$$

(3) The final R.E is the sum of regular expressions obtained from the reduced automata for each accepting states in which,

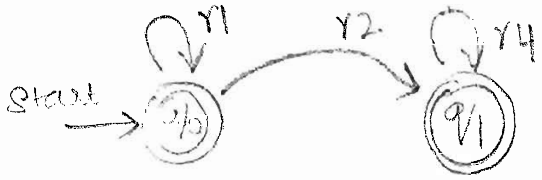
a)



$$R.E = r_1^* r_2 r_4^*$$

If r_3 is not there then the R.E is as shown.

b)

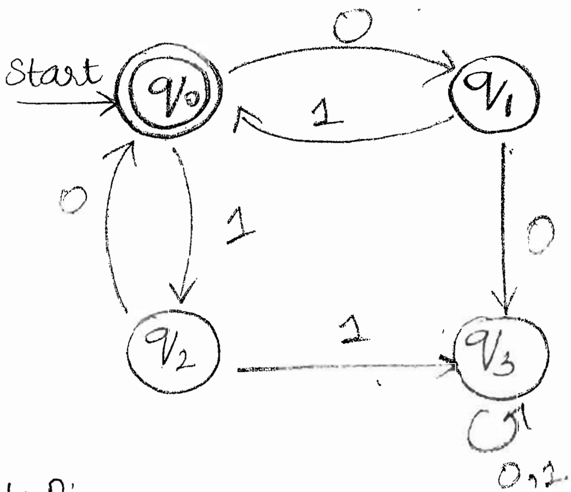


If q_0, q_1 are both, final states, then R.E is of the form,

$$R.E = r_1^* + r_1^* r_2 r_4^*$$

29/9/18

Q1) Obtain a R.E for FA shown below

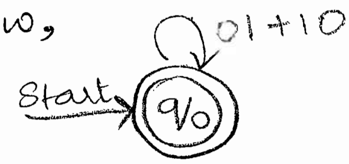


Soln:

Since q_3 is the trap state which is going to be eliminated, the resulting FA is,

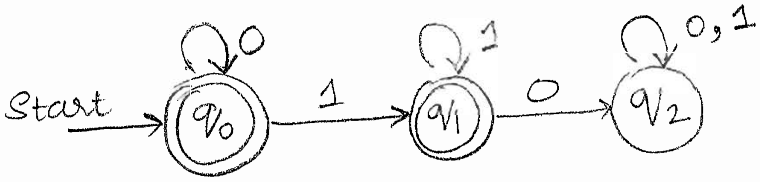
Here q_1, q_2 are intermediate states. we reach q_0 from q_1 upon accepting any no. of 01 and also upon accepting any no. of 10 we reach q_0 from q_2.

Now,

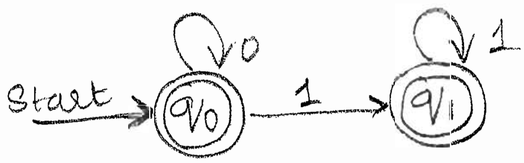


$\therefore$ R.E $= (01+10)^*$

Q2) Obtain R.E for FA shown below



Soln: Since q_2 is trap state, which is going to be eliminated, the resulting FA is,



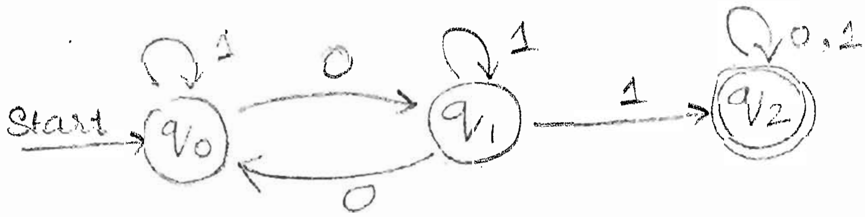
R.E $= 0^* + 0^* 1 1^*$

$= 0^* (\varepsilon + 11^*)$

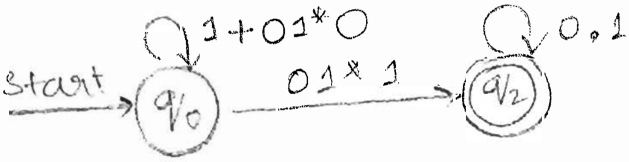
$= 0^* (\varepsilon + 1^+)$ [$\because \varepsilon + 1^+ = 1^*$]

$\therefore$ $\boxed{\text{R.E} = 0^* 1^*}$

Q3) Obtain R.E for FA shown below,



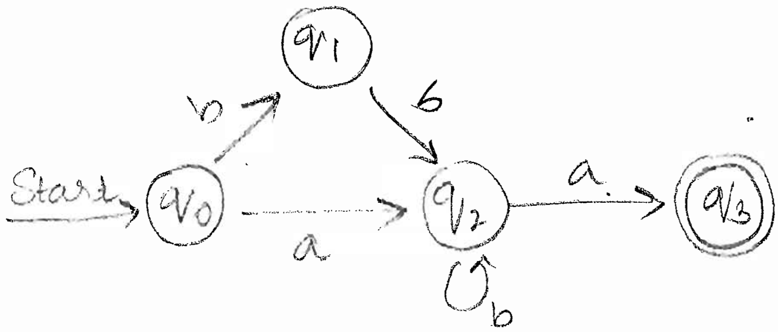
Soln:



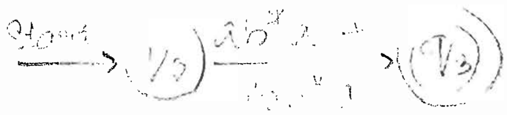
R.E $= (1+01^*0)\, 01^*1\, (0+1)^*$

$\therefore$ $\boxed{\text{R.E} = (1+01^*0)\, 01^+ (0+1)^*}$

Q4) Build a R.E from the following FSM.



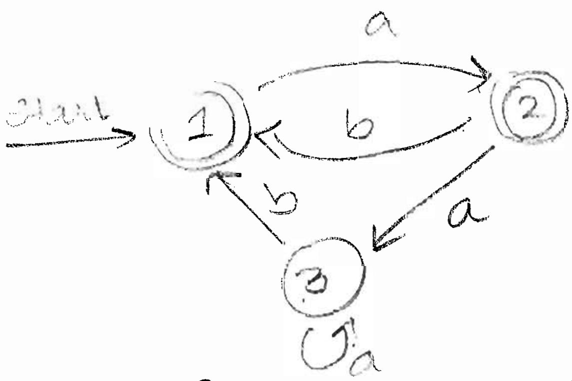
Soluⁿ:



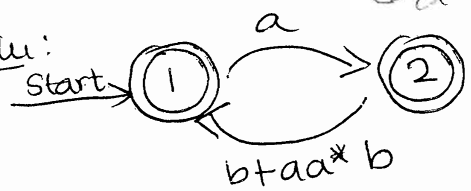
R.E = $ab^*a + bbb^*a$

$\boxed{\therefore R.E = (a + bb)\,b^*a}$

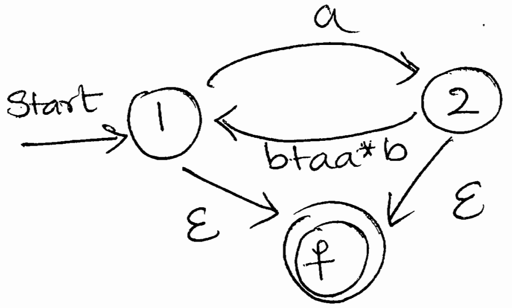
Q5) Build a R.E from the following FSM.



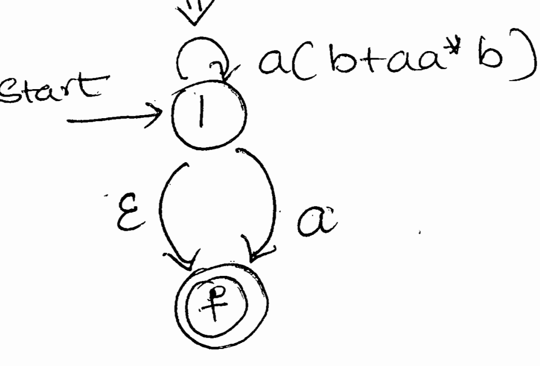
Solu:



⇒



⇓



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

03/10/18

Pumpping lemma:-

The general strategy used to prove that certain languages are not regular is shown below:

Step1: Assume that language L is regular and n be the finite no. of states of FA.

Step2: Select string x such that $|x| \geq n$ and brake the string into substrings u, v and w such that,

$$x = uvw \quad \text{with the constraints,}$$

- $v \neq \varepsilon$ ie, $|v| \geq 1$
- $|uv| \leq n$

Step3: Find any i such that uv^iw is not in L ie, $uv^iw \notin L$.

According to pumpping lemma uv^iw is in the language for $i \geq 0$ so the result is contradiction to the assumption that the language is regular.

$\therefore$ The given language L is not regular.

Q1) Show that the language $L = \{0^n 1^n \mid n \geq 1\}$ is not regular.

Soln:- Case(1):-

By pumpping lemma we write,

'n' should not be there in 'v' part

$x = 0^n 1^n$

$|x| = 2n \geq n$

$x = \underbrace{000\ldots00}_{n}\underbrace{111\ldots11}_{n}$

$x = \underbrace{000\ldots00}_{n-k}\underbrace{00\ldots0}_{k}\underbrace{111\ldots11}_{n}$

$$x = \underbrace{0^{n-K}}_{u}\underbrace{0^{K}}_{v}\underbrace{1^{n}}_{w}$$

consider, $u = 0^{n-K}$,
$v = 0^{K}$ and
$w = 1^{n}$

By pumpping lemma we write,

$$x = 0^{n-K}(0^{K})^{i}\,1^{n} \qquad [\because uv^{i}w]$$

for $i = 0$ we get,

$$x = 0^{n-K}\,1^{n}$$

which is a contradiction because,

$$0^{n-K}\,1^{n} \notin L.$$

In $0^{n-K}1^{n}$, the no. of 0's are less than the no. of 1's as $K \geq 1$

$\boxed{\therefore \text{ The given language is not regular.}}$

Case (2) :- By pumpping lemma,

$$x = 0^{n}1^{n}$$

$$|x| = 2n \geq n$$

$$x = \underbrace{00\ldots0}_{n}\ \underbrace{11\ldots1}_{K}\ \underbrace{11\ldots1}_{n-K}$$

$$x = 0^{n}\,1^{K}\,1^{n-K}$$

consider, $u = 0^{n}$, $v = 1^{K}$ and
$w = 1^{n-K}$

By pumpping lemma we write,

$$x = 0^{n}(1^{K})^{i}(1^{n-K})$$

for $i = 0$ we get,

$$x = 0^{n}\,1^{n-K}$$

which is a contradiction because,

$0^n 1^{n-k} \notin L$

In $0^n 1^{n-k}$, the no. of 1's are less than the no. of 0's as $k \geq 1$.

$\boxed{\therefore \text{ The given language is not regular.}}$

<u>case (3)</u>: By pumping lemma,

$x = 0^n 1^n$

$|x| = 2n \geq n$

$x = \underbrace{0 \ldots 0}_{n-p} \underbrace{0 \ldots 0}_{p} \underbrace{1 \ldots 1}_{q} \underbrace{11 \ldots 11}_{n-q}$

$x = 0^{n-p} 0^p 1^q 1^{n-q}$

Consider $u = 0^{n-p}$, $v = 0^p 1^q$ and $w = 1^{n-q}$

By pumping lemma we write,

$x = 0^{n-p} (0^p 1^q)^i 1^{n-q}$

for $i = 0$ we get,

$x = 0^{n-p} 1^{n-q}$

which is a contradiction because,

$0^{n-p} 1^{n-q} \notin L$

In $0^{n-p} 1^{n-q}$, the no. of 1's and no. of 0's are different as $p \neq q$ and $p, q \geq 1$.

$\boxed{\therefore \text{ The given language is not regular}}$

Q2) Show that the language $L = \{ww^R \mid w \in \{0,1\}^*\}$ is not regular. 04/11

↓ Palindrome with even length

Q3

Soln: By pumping lemma,

Sol

$x = 0^n 1^n 1^n 0^n$

$|x| = 4n \geq n$

$x = \underbrace{00\cdots0}_{n}\underbrace{11\cdots1}_{n}\underbrace{11\cdots1}_{n}\underbrace{00\cdots0}_{n}$

$x = \underbrace{00\cdots0}_{n-k}\underbrace{00\cdots0}_{k}\underbrace{11\cdots1}_{n}\underbrace{11\cdots1}_{n}\underbrace{00\cdots0}_{n}$

$x = \underbrace{0^{n-k}}_{u}\underbrace{0^k}_{v}\underbrace{1^n 1^n 0^n}_{w}$ → case (1)

consider $u = 0^{n-k}$
$v = 0^k$
$w = 1^n 1^n 0^n$

By pumping lemma we write,

$x = 0^{n-k}(0^k)^i 1^n 1^n 0^n$

for $i = 0$ we get,

$x = 0^{n-k} 1^n 1^n 0^n$

which is a contradiction because,

$0^{n-k} 1^n 1^n 0^n \notin L$

In $0^{n-k} 1^n 1^n 0^n$, the no. of 0's are less than the no. of 1's. as $k \geq 1$.

$\boxed{\therefore \text{ The language is not regular.}}$

//

04/10/18

Q3) Show that Language $L = \{a^i b^j \mid i > j\}$ is not regular.

Soln: By pumpping lemma,

$x = a^{n+1} b^n$

$x = a^n a b^n$

$|x| = 2n+1 \geq n$

$x = \underbrace{aa \ldots a}_{n} \, a \, \underbrace{bb \ldots b}_{n}$

$x = \underbrace{aa \ldots a}_{n-k} \, \underbrace{a \ldots a}_{k} \, a \, \underbrace{bb \ldots b}_{n}$

$x = a^{n-k} a^k a b^n$ case (1)

consider $u = a^{n-k}$, $v = a^k$, $w = ab^n$

By pumpping lemma we write,

$x = a^{n-k} (a^k)^i ab^n$

for $i = 0$ we get,

$x = a^{n-k} ab^n$

which is a contradcction because,

$a^{n-k} ab^n \notin L$

In $a^{n-k} ab^n$, the no. of a's are less than the no. of b's as $k \geq 1$.

$\boxed{\therefore \text{ The given language is not regular}}$ //

Q4) Show that the language $L = \{ a^i b^j \mid i<j \}$ is not regular.

<u>Soln:</u> By pumpping lemma,

$$x = a^n b^{n+1}$$

$$x = a^n b b^n$$

$$|x| = 2n+1 \geq n$$

$$x = \underbrace{aa\ldots a}_{n}\ b\ \underbrace{bb\ldots b}_{n}$$

$$x = \underbrace{aa\ldots a}_{n}\ b\ \underbrace{b\ldots b}_{K}\ \underbrace{bb\ldots b}_{n-K}$$

$$x = a^n b b^K b^{n-K} \longrightarrow \text{case(2)}$$

consider, $u = a^n b$

$v = b^K$

$w = b^{n-K}$

By pumpping lemma we write,

$$x = a^n b (b^K)^i b^{n-K}$$

for $i=0$ we get,

$$x = a^n b b^{n-K}$$

which is a contradiction because,

$$x = a^n b b^{n-K} \notin L$$

In $a^n b b^{n-K}$, the no. of a's are more than the no. of b's as $K \geq 1$.

$\boxed{\therefore \text{ The language is not regular.}}$

Properties of Regular languages:-

Closure properties of Regular Languages

- Union of two RL's is Regular
- Intersection of two RL's is Regular
- Complement of RL is Regular
- closure (or) star of a RL is Regular
- The concatenation of two RL's is Regular
- Difference of two RL's is Regular
- Reversal of a RL is Regular.
- A homomorphism of a RL is Regular
- The inverse homomorphism of a RL is Regular.

* Regular languages are closed under union, concatenation and star:-

Theorem: If L_1 and L_2 are regular then,

$L_1 \cup L_2$

$L_1 . L_2$

L_1^* also denote the regular languages.

Proof: It is given that L_1 and L_2 are regular languages. So there exists a regular expressions R_1 and R_2 such that,

$$L_1 = L(R_1)$$
$$L_2 = L(R_2)$$

By definition of regular expressions we have,

- $R_1 + R_2$ is a RE, denoting the language $L_1 \cup L_2$
- $R_1 . R_2$ is a R.E, denoting the language $L_1 . L_2$

• R_1^* is a R.E denoting the language L_1^*.

Hence it's proved.

*Closure under complementation:-

Theorem: If L is a regular language then, the complement of L is denoted by $\bar{L}$ is also regular.

(OR)

The set of regular languages is closed under complementation.

Proof: Let $M_1 = (Q, \Sigma, \delta, q_0, F)$ be a DFA which accepts the language L.

Since the language is accepted by a DFA, then the language is Regular.

Now, let us define a machine

$M_2 = (Q, \Sigma, \delta, q_0, Q-F)$ which accepts the language $\bar{L}$.

Note that there is no difference between M_1 and M_2 except the final states. The nonfinal states of M_1 are the final states of M_2 and final states of M_1 are the nonfinal states of M_2. So the language which is rejected by M_1 is accepted by M_2 and vice-versa. Thus we have a machine M_2 which accepts all those strings denoted by $\bar{L}$, that are rejected by machine M_1.

So the regular language is closed under complementation.

05/10/18

*Closure under Intersection:-

Theorem: If L_1 and L_2 are regular languages then the regular language is closed under intersection.

Proof: Let us consider $M_1 = (Q_1, \Sigma, \delta_1, q_1, F_1)$ which accepts L_1. and $M_2 = (Q_2, \Sigma, \delta_2, q_2, F_2)$ which accepts L_2. Now, we define $M = (Q, \Sigma, \delta, q, F)$ recognizing $L_1 \cap L_2$ as follows,

- $Q = Q_1 \times Q_2$ where, (p, q) is in Q. where $p \in Q_1$ and $q \in Q_2$.
- Σ is same for both the machines.
- $\delta : Q \times \Sigma \rightarrow Q$ is defined by

$$\delta((p,q), a) = (\delta_1(p,a), \delta_2(q,a))$$

- $q : (q_1, q_2)$ is the start state where q_1 is start state of M_1 and q_2 is start state of M_2.
- $F = \{ (p,q) \mid p \in F_1 \text{ and } q \in F_2 \}$

$\therefore$ The regular language is closed under Intersection.

M_2

*Closure under Difference:-

Theorem: If L_1 and L_2 are regular languages, then regular language is closed under difference

(OR)

If L_1 and L_2 are regular languages then $L_1 - L_2$ is also regular.

Proof: Let us consider $M_1 = (Q_1, \Sigma, \delta_1, q_1, F_1)$ which accepts L_1 and $M_2 = (Q_2, \Sigma, \delta_2, q_2, F_2)$ which accepts L_2.

Now, we define $M = (Q, \Sigma, \delta, q, F)$ recognizing $L_1 - L_2$ as follows,

- $Q = Q_1 \times Q_2$ where (p, q) is in Q where $p \in Q_1$ and $q \in Q_2$.
- Σ is same for both the machines.
- $\delta : Q \times \Sigma \rightarrow Q$ is defined by

$$\delta((p,q), a) = (\delta_1(p,a), \delta_2(q,a))$$

- $q = (q_1, q_2)$ is the start state where q_1 is the start state of M_1 and q_2 is the start state of M_2.

- $F = \{(p,q) \mid p \in F_1 \text{ and } q \notin F_2\}$.

$\therefore$ The regular language is closed under difference.

*

*Closure under Reversal:

Theorem: If L is regular then, L^R is also regular.

Proof: Let L be the regular language corresponding to R.E 'E'. It is required to prove that there is another R.E 'E^R' such that,

$$L(E^R) = (L(E))^R$$

which is read as,

" Language of E^R is the reversal of language of E "

Basic step: By definition of R.E 'E' we have,

- ϕ is R.E
- ε is R.E
- a is R.E

So, the reversal of R.E 'E^R' is given by,

- $\{\phi\}^R = \phi$
- $\{\varepsilon\}^R = \varepsilon$
- $\{a\}^R = a$

Induction step: Again by definition of R.E, if E_1 and E_2 are R.E's then,

- $E_1 + E_2$ is R.E.
- $E_1 \cdot E_2$ is R.E.
- E_1^* is a R.E.

which results in 3 different cases. Now, let us prove ~~each~~ of these cases.

case (1): $E = E_1 + E_2$

If $E = E_1 + E_2$ is a R.E then, $E^R = E_1^R + E_2^R$ is a R.E representing the language, $L(E^R) = L(E_1^R) + L(E_2^R)$

case (2): $E = E_1 \cdot E_2$

If $E = E_1 \cdot E_2$ is a R.E then, $E^R = E_1^R \cdot E_2^R$ is a R.E representing the language, $L(E^R) = L(E_1^R) \cdot L(E_2^R)$

case (3): $E = E_1^*$

If $E = E_1^*$ is a R.E then, $E^R = (E_1^R)^*$ is a R.E denoting the language, $L(E^R) = L(E_1^R)^*$.

* Closure under Homomorphism:

- **Homomorphism:-**

  Let Σ and Γ are set of alphabets. The Homopheric function,

  $$h : \Sigma \rightarrow \Gamma^*$$

  is called homomorphism.

  ie, substitution where, a single letter is replaced by a string.

  if $w = a_1 a_2 \ldots a_n$

  then $h(w) = h(a_1)\, h(a_2) \ldots h(a_n)$

  If L is made up of alphabets Σ then,

  $h(L) = \{ h(w) \mid w \in L \}$ is called homopheric image.

eg(1): Let $\Sigma = \{0, 1\}$, $\Gamma = \{0, 1, 2\}$ and $h(0) = 01$, $h(1) = 112$. What is i) $h(010)$?

(ii) If $L = \{00, 010\}$ what is homophese image of $h(L)$?

Soln:

(i) $h(010) = h(0)\, h(1)\, h(0)$

$= 0111201$

(ii) $L = \{00, 010\}$

$\Rightarrow h(L) = \{ h(00), h(010) \}$

$= \{ h(0)h(0),\ h(0)h(1)h(0) \}$

$\therefore h(L) = \{ 0101,\ 0111201 \}$

eg(2) If $\Sigma = \{0,1\}$, $\Gamma = \{1,2,3\}$, $h(0) = 3122$ and $h(1) = 132$ then find $h((0+1)^*(00)^*)$

Soln:
$$h((0+1)^*(00)^*)$$
$$= (h(0)+h(1))^* (h(0)h(0))^*$$
$$= (3122+132)^* (3122\,3122)^*$$

Limitations of Finite Automata

Note: There are so many problems for which we can not construct a DFA and still we want some solution to solve those problems for example

① check for the matching parenthesis (not possible using DFA)

② count number of a's and no. of b's (not possible using DFA) and so it can not be used as a counter.

Limitations:

① An FA has a finite no of states and so it does not have the capacity to remember arbitrary long amount of information.

② Since it does not have memory, FA can not remember a long string. For example to check for matching parenthesis, check whether the string is a palindrome or not etc... are not possible using FA.

③ Finite automata or finite state machines have trouble in recognizing various types of languages involving counting, calculating, storing the string.

05|

★De

ec

★r

→

05|10|18

Module - 3

CONTEXTFREE GRAMMARS (CFG)

✭ Definition of a Grammar:

A grammar G is 4 tuple (or) quadruple $G = (V, T, P, S)$ where,

- $V \rightarrow$ set of variables (or) non-terminals
- $T \rightarrow$ set of terminals.
- $P \rightarrow$ set of productions (or) rules
  Each production is of the form $\alpha \rightarrow \beta$ where α is in a string $(V \cup T)^+$ and α cannot be ε. and ε cannot occur on left hand side of any production.
  But β is a string in $(V \cup T)^*$ and hence it includes ε also. So ε can occur on right hand side of the production.
- $S \rightarrow$ it is the start symbol.

eg: Vimala ate slowly
noun verb adverb

1) sentence → noun verb adverb
2) noun → vimala
3) verb → ate
4) adverb → slowly.

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

V – {sentence, noun, verb, adverb}

✭ Notations used while constructing the grammars:

→ The following are the terminals,

a) The keywords such as if, for, while, dowhile etc...
b) Digits from 0 to 9.
c) Symbols such as +, −, ∗, % etc...
d) the lowercase letters such as a, b, c, ...
e) the Bold phase letters id

→ The following are the non-terminals

a) The lower case names such as expressions, operators, operands, statements etc..

b) The capital letter such as A, B, C, ...etc.

c) The S is the start symbol

*CHOMSKY HIERARCHY :-

The grammar can be classified as shown below -

types of Grammars →
- → Type 0 Grammar (phrase structured Grammar)
- → Type 1 Grammar (context sensitive Grammar)
- → Type 2 Grammar (contextfree Grammar) (CFG)
- → Type 3 Grammar (Regular Grammar)

→ Type 0 Grammar (Definition) :

A grammar $G = (V, T, P, S)$ said to be type 0 grammar (or) unrestricted grammar (or) phrase structured grammar if all the productions is of the form $\alpha \rightarrow \beta$ where $\alpha \in (V \cup T)^+$ and $\beta \in (V \cup T)^*$.

In this type of grammar there is no restrictions on length of α and β. The only restriction is that α cannot be ε ie, ε can't appear on the left hand side of any production. But ε can appear on righthand side of production.

This is the largest family of grammars, more powerful than all types of grammars. Any language can be obtained from this grammar. Only Turing machines (TM) can recognize this language.

eg: $S \rightarrow aAb \mid \varepsilon$
$aA \rightarrow bAA$
$bA \rightarrow a$

Type 1 Grammar

A grammar $G = (V, T, P, S)$ is said to be Type 1 grammar (or) context sensitive grammar if all the productions are of the form $\alpha \rightarrow \beta$ as in type 0 grammar. But there is a restriction on length of β must be atleast as much of length of α.

ie, $|\beta| \geq |\alpha|$ and $\alpha, \beta \in (V \cup T)^+$ ie, ε cannot appear on the righthand side (or) lefthand side of the productions.

It is an ε free grammar. The linear bounded automata (LBA) can be constructed to recognize the language generated from this grammar.

eg: $S \rightarrow aAb$
$aA \rightarrow bAA$

*** Type 2 Grammar

A grammar $G = (V, T, P, S)$ is said to be Type 2 grammar (or) contextfree grammar (CFG) is all the productions of the grammar are of the form

$A \rightarrow \alpha$ where, $\alpha \in (V \cup T)^*$ and
A is a nonterminal (or) a variable.

The symbol ε can be appear on the righthand side of the any production.

- Pushdown Automata (PDA) can be constructed to recognize the language generated from this grammar.

eg: $S \to aB \mid bA \mid \varepsilon$

$A \to aA \mid b$

$B \to bB \mid a \mid \varepsilon$

Type 3 Grammar (or) Regular Grammar

The grammar $G = (V, T, P, S)$ is said to be type 3 grammar (or) regular grammar if the grammar is right linear (or) left linear.

Right linear grammar: A grammar G is said to be right linear if all the productions are of the form,

$A \to wB$

or

$A \to w$

where, A, B are the variables $(A, B \in V)$ and $w \in T^*$

eg: $S \to aaB \mid bbA \mid \varepsilon$

$A \to aA \mid b$

$B \to bB \mid a \mid \varepsilon$

eg: $S \to aaB$

$\to aabB$

$\to aaba$

Left linear grammar: A grammar G is said to be left linear if all the productions are of the form,

$A \to Bw$

or

$A \to w$

where, $A, B \in V$ and $w \in T^*$

eg: $S \to Baa \mid Abb \mid \varepsilon$

$A \to Aa \mid b$

$B \to Bb \mid a \mid \varepsilon$

eg: $S \to Baa$

$\to Bbaa$

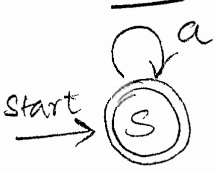
$\to abaa$

Grammar from Finite Automata (FA)

09/10/18

Q1) Obtain a grammar to generate a string consisting of any no. of a's.

DFA



transitions

$\delta(S, a) = S$

S is final state

Grammar

$S \to aS$

$S \to \varepsilon$

∴ The resulting grammar $G = (V, T, P, S)$

$V = \{S\}$

$T = \{a\}$

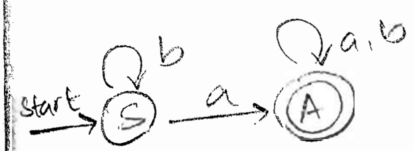
$P = \{ S \to aS \mid \varepsilon \}$

$S = \{S\}$ is the start state.

1) $S \to \varepsilon$
2) $S \to aS$
aaS
aaaS

Q2) Obtain a grammar to generate a string consisting of any no. of a's and b's with atleast one 'a'.

DFA



transitions

$\delta(S, a) = A$

$\delta(S, b) = S$

$\delta(A, a) = A$

$\delta(A, b) = A$

A is finalstate

Grammar

$S \to aA$

$S \to bS$

$A \to aA$

$A \to bA$

$A \to \varepsilon$

∴ The resulting grammar $G = (V, T, P, S)$

$V = \{S, A\}$

$T = \{a, b\}$

$P = \{ S \to aA \mid bS, \ A \to aA \mid bA \mid \varepsilon \}$

$S \to$ start state.

Grammar from Regular expressions :-

Q1) Obtain grammar to generate strings of a's & b's having a substring 'ab'.

$$R.E = (a+b)^* ab (a+b)^*$$

$$\Downarrow$$

$$S \rightarrow AabA$$
$$A \rightarrow \varepsilon \mid aA \mid bA$$

$\therefore$ The resulting grammar $G = (V, T, P, S)$

$V = \{S, A\}$

$T = \{a, b\}$

$P = \{ \quad S \rightarrow AaaA,$
$\quad\quad A \rightarrow \varepsilon \mid aA \mid bA \}$.

$S \rightarrow$ is the start state.

Q2) Obtain a grammar to generate strings of a's & b's ending with string 'ab'.

$$R.E = (a+b)^* ab$$

$$\Downarrow$$

$$S \rightarrow Aab$$
$$A \rightarrow \varepsilon \mid aA \mid bA$$

$\therefore$ The resulting grammar $G = (V, T, P, S)$

$V = \{S, A\}$

$T = \{a, b\}$

$P = \{ \quad S \rightarrow Aab,$
$\quad\quad A \rightarrow \varepsilon \mid aA \mid bA \}$

$S \rightarrow$ is the start state

Q3) Obtain a grammar to generate strings of a's & b's starting with 'ab'.

$$R.E = ab(a+b)^*$$

$$\Downarrow$$

$$S \to abA$$
$$A \to \varepsilon \mid aA \mid bA$$

$\therefore$ The resulting grammar $G = (V, T, P, S)$

$V = \{S, A\}$

$T = \{a, b\}$

$P = \{ S \to abA,$
$A \to \varepsilon \mid aA \mid bA \}$

S is the start state

Q4) Obtain a grammar to obtain the following language $L = \{ w : n_a(w) \bmod 2 = 0$ where $w \in \{a,b\}^* \}$

$$R.E = (b^* a b^* a b^*)^*$$

$$\Downarrow$$

$$S \to \varepsilon \mid AaAaA \mid SS \quad (\text{or}) \quad \varepsilon \mid AaAaAS$$
$$A \to \varepsilon \mid bA$$

$\therefore$ The resulting grammar $G = (V, T, P, S)$

$V = \{S, A\}$

$T = \{a, b\}$

$P = \{ S \to \varepsilon \mid AaAaAS,$
$A \to \varepsilon \mid bA \}$

S is the start state

Derivation:

Consider the following grammar shown below from which any arithmetic expression can be obtained.

$E \rightarrow E+E$
$E \rightarrow E-E$
$E \rightarrow E*E$
$E \rightarrow E/E$
$E \rightarrow id$

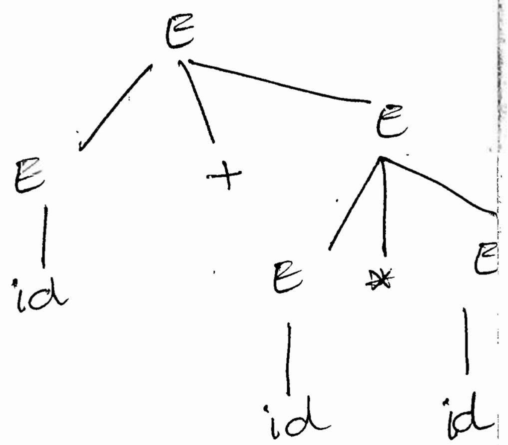
Rule ($\rightarrow$)
Derivation ($\Rightarrow$)

Obtain the string id+id*id and show the derivation for the same.

$E \Rightarrow E+E$
$\Rightarrow id+E$
$\Rightarrow id+E*E$
$\Rightarrow id+id*E$
$\Rightarrow id+id*id$

$\boxed{E \overset{+}{\Rightarrow} id+id*id}$

Derivation tree
(Parse tree)

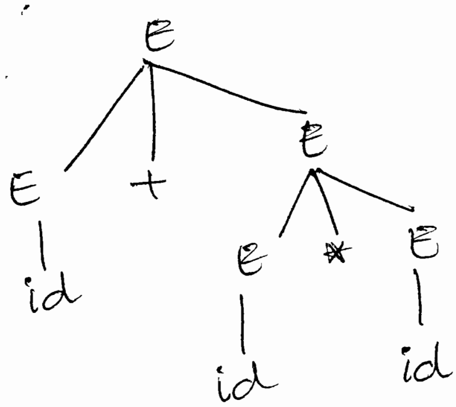


leftmost derivation

$E \Rightarrow E+E$
$\Rightarrow id+E$
$\Rightarrow id+E*E$
$\Rightarrow id+id*E$
$\Rightarrow id+id*id$

$\therefore \boxed{E \underset{lm}{\overset{+}{\Rightarrow}} id+id*id}$

Derivation tree
(Parse tree)

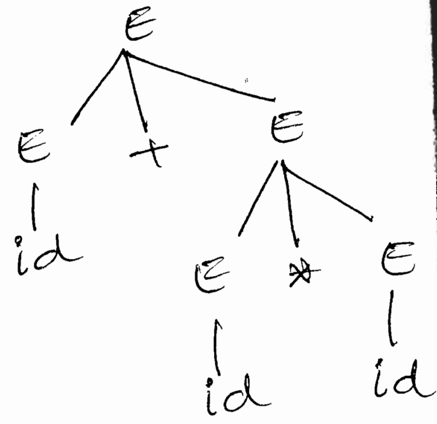


Rightmost derivation

$E \Rightarrow E + E$
$\Rightarrow E + E * E$
$\Rightarrow E + E * id$
$\Rightarrow E + id * id$
$\Rightarrow id + id * id$

$\therefore \boxed{E \overset{+}{\underset{rm}{\Rightarrow}} id + id * id}$

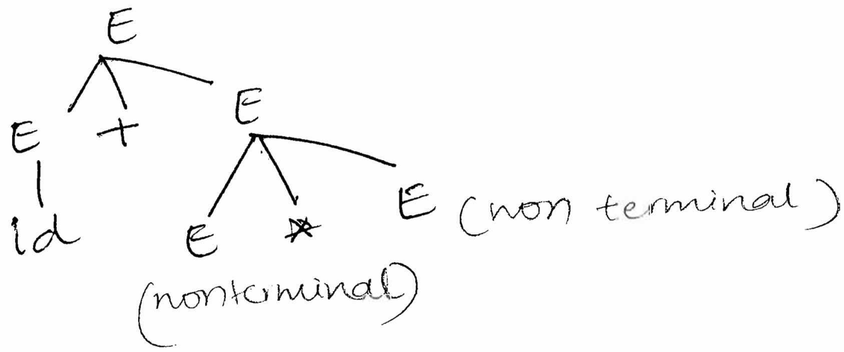
Derivation tree (Parse tree)



Partial derivation tree (Partial parse tree)

when all the leaf nodes are not terminals, then those trees are called partial parse tree.

eg:



Sentence :- Let $G = (V, T, P, S)$ be a context free grammar. A string $w \in (V \cup T)^*$ which is derivable from the start symbol S such that $S \overset{*}{\Rightarrow} w$ is called a sentence (or) sentential form of $\rightarrow G$.

Language: consider the following grammar

$S \rightarrow aCa$
$C \rightarrow aCa \mid b$ what is the language generated by this grammar?

$S \Rightarrow aCa$
$\Rightarrow aaCaa$
$\Rightarrow aaaCaaa$
$\vdots$
$\Rightarrow a^n C a^n$

$\Rightarrow a^n b a^n$

$\therefore \boxed{L(G) = \{ a^n b a^n \mid n \geq 1 \}}$

Ambiguous Grammar:

Note:

① Obtain the leftmost derivation and get a string w, obtain the rightmost derivation and get a string u. For both the derivations construct the parse tree. If there are two different parse trees, then the grammar is ambiguous.

② Obtain the string w by applying leftmost derivation twice and construct the parse tree. If the two parse tree are different, the grammar is ambiguous.

③ Obtain the string w by applying rightmost derivation twice and construct the parse tree. If the two parse trees are different, the grammar is ambiguous.

④ Apply the leftmost derivation and get a string. Apply the leftmost derivation again and get a different string. The parse trees obtained will naturally be different and do not come to the conclusion that the grammar is ambiguous.

Q1) Consider the grammar shown below from which any arithmetic operation can be obtained.

$E \to E+E$
$E \to E-E$
$E \to E*E$
$E \to E/E$
$E \to (E)/E$
$E \to id$

Show that the grammar is ambiguous.

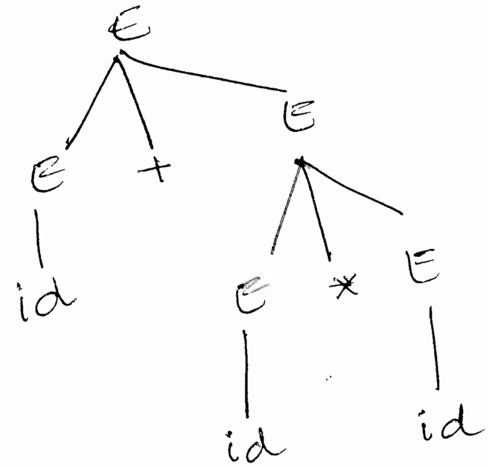
Take string id+id*id

leftmost derivation

$E \Rightarrow E+E$
$\Rightarrow id+E$
$\Rightarrow id+E*E$
$\Rightarrow id+id*E$
$\Rightarrow id+id*id$

$\therefore \boxed{E \overset{+}{\underset{lm}{\Rightarrow}} id+id*id}$

Parse tree

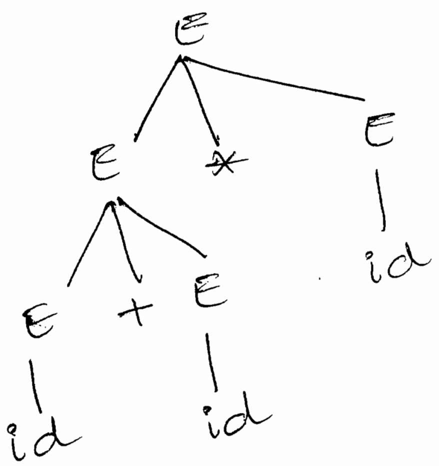


Rightmost derivation

$E \Rightarrow E*E$
$\Rightarrow E*id$
$\Rightarrow E+E*id$
$\Rightarrow E+id*id$
$\Rightarrow id+id*id$

$\therefore \boxed{E \overset{+}{\underset{rm}{\Rightarrow}} id+id*id}$

Parse tree



Since the parse trees are different the given grammar is ambiguous.

10/10/18

Simplification of CFG

① **Substitution method:-**

Q1) Consider the production

$A \to aBa$

$B \to ab \mid b$

Simplify the grammar by substitution method.

Solu^n:

$A \to a(ab \mid b)a$

$\Downarrow$

$A \to aaba \mid aba$

$\therefore$ The resulting grammar $G = (V, T, P, S)$

$V = \{A\}$

$T = \{a, b\}$

$P = \{A \to aaba \mid aba\}$

$S = A$ is the start symbol.

***Left Recursion:**

$A \to A\alpha_1 \mid A\alpha_2 \mid \dots \mid A\alpha_n \mid \beta_1 \mid \beta_2 \mid \dots \mid \beta_m$

$\Downarrow$

$A \to \beta_1 A' \mid \beta_2 A' \mid \dots \mid \beta_m A'$

$A' \to \alpha_1 A' \mid \alpha_2 A' \mid \dots \mid \alpha_n A' \mid \varepsilon$

(OR)

$A \to A\alpha_i \mid \beta_j$

$\Downarrow$

$A \to \beta_j A'$

$A' \to \alpha_i A' \mid \varepsilon$

Q1)

Solu

Q1) Eliminate the left recursion from the following grammar.

$E \rightarrow E+T \mid T$

$T \rightarrow T*F \mid F$

$F \rightarrow (E) \mid id$

Soln:

Given	Substitution	without left recursion
$A \rightarrow A\alpha_i \mid \beta_j$		$A \rightarrow \beta_j A'$ $A' \rightarrow \alpha_i A' \mid \varepsilon$
$E \rightarrow E+T \mid T$	$A = E$ $\alpha_1 = +T$ $\beta_1 = T$	$E \rightarrow TE'$ $E' \rightarrow +TE' \mid \varepsilon$
$T \rightarrow T*F \mid F$	$A = T$ $\alpha_1 = *F$ $\beta_1 = F$	$T \rightarrow FT'$ $T' \rightarrow *FT' \mid \varepsilon$
$F \rightarrow (E) \mid id$	NA	$F \rightarrow (E) \mid id$

∴ The resulting grammar after eliminating left recursion, $G = (V, T, P, S)$

$V = \{E, E', T, T', F\}$

$T = \{+, *, (,), id\}$

$P = \{ E \rightarrow TE', \quad E' \rightarrow +TE' \mid \varepsilon,$
$T \rightarrow FT', \quad T' \rightarrow *FT' \mid \varepsilon,$
$F \rightarrow (E) \mid id \}$

E is the start symbol.

Q2) Eliminate the left recursion from the following grammar

$S \to Ab \mid a$

$A \to Ab \mid Sa$

Solu:

$S \to Ab \mid a$

$A \to Ab \mid (Ab \mid a)a$

$\Downarrow$

$S \to Ab \mid a$

$A \to Ab \mid Aba \mid aa$

Given	Substitution	without left recursion
$A \to A\alpha_i \mid \beta_j$		$A \to \beta_j A'$ $A' \to \alpha_i A' \mid \varepsilon$
$S \to Ab \mid a$	NA	$S \to Ab \mid a$
$A \to Ab \mid Aba \mid aa$	$A = A$ $\alpha_1 = b$ $\alpha_2 = ba$ $\beta_1 = aa$	$A \to aaA'$ $A' \to bA' \mid baA' \mid \varepsilon$

$\therefore$ The resulting grammar after eliminating left recursion, $G = (V, T, P, S)$

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

$T = \{a, b\}$

$P = \{ S \to Ab \mid a,$
$A \to aaA'$
$A' \to bA' \mid baA' \mid \varepsilon \}$

S is the start symbol.

Q3)

11/10/18

Q3) Eliminate the useless symbols in the following grammar

11/10/18

$S \rightarrow aA \mid bB$

$A \rightarrow aA \mid a$

$B \rightarrow bB$

$D \rightarrow ab \mid Ea$

$E \rightarrow aC \mid d$

Let $G = (V, T, P, S)$ be the given grammar.

Step 1: Eliminating useless symbols:-

Old variable (OV)	New variable (NV)	Productions
ϕ	A, D, E	$A \rightarrow a$ $D \rightarrow ab$ $E \rightarrow d$
A, D, E	S, A, D, E	$S \rightarrow aA$ $A \rightarrow aA$ $D \rightarrow Ea$
S, A, D, E	S, A, D, E	—

∴ The resulting grammar, $G_1 = (V_1, T_1, P_1, S)$

$V_1 = \{S, A, D, E\}$

$T_1 = \{a, b, d\}$

$P_1 = \{A \rightarrow a,\ D \rightarrow ab,\ E \rightarrow d,\ S \rightarrow aA,\ A \rightarrow aA,\ D \rightarrow Ea\}$

S is the start symbol.

Now,

$S \rightarrow aA$

$A \rightarrow a \mid aA$

$D \rightarrow ab \mid Ea$

$E \rightarrow d$

Step2: Eliminating useless productions:

$G' = (V', T', P', S)$

P'	T'	V'
—	—	S
$S \to aA$	a	S, A
$A \to aA \mid a$	a	S, A

$\therefore$ The resulting grammar, $G' = (V', T', P', S)$

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

$T' = \{a\}$

$P' = \{S \to aA,$
$A \to aA \mid a\}$

S is the start symbol.

Q2) Simplify the following grammar

$S \to aA \mid a \mid Bb \mid cC$

$A \to aB$

$B \to a \mid Aa$

$C \to cCD$

$D \to ddd$

Let $G = (V, T, P, S)$ be the given grammar.

Step1: Eliminating useless symbols

OV	NV	Productions
ϕ	S, B, D	$S \to a$ $B \to a$ $D \to ddd$
S, B, D	S, A, B, D	$S \to Bb$ $A \to aB$ $\cancel{C \to cCD}$

OV	NV	Productions
S, A, B, D	S, A, B, D	$S \to aA$ $B \to Aa$

$\therefore$ The resulting Grammar is,

$G_1 = (V_1, T_1, P_1, S)$

$V_1 = \{ S, A, B, D \}$

$T_1 = \{ a, b, d \}$

$P_1 = \{ S \to a | Bb | aA,$

$A \to aB$

$B \to a | Aa$

$D \to ddd \}$

S is the start symbol.

<u>Step 2:</u> Eliminating useless productions.

P'	T'	V'
—	—	S
$S \to a \mid Bb \mid aA$	a, b	S, A, B
$A \to aB$	a, b	S, A, B
$B \to a \mid Aa$	a, b	S, A, B

$\therefore$ The resulting grammar, $G' = (V', T', P', S)$

$V' = \{ S, A, B \}$

$T' = \{ a, b \}$

$P' = \{ S \to a | Bb | aA,$

$A \to aB$

$B \to a | Aa \}$

S is the start symbol.

Q1) Eliminate all ε-Productions from the following grammar.

$$S \rightarrow ABCa \mid bD$$
$$A \rightarrow BC \mid b$$
$$B \rightarrow b \mid \varepsilon$$
$$C \rightarrow c \mid \varepsilon$$
$$D \rightarrow d$$

Step1: To find nullable variables.

O.V	N.V	Productions
ϕ	B, C	$B \rightarrow \varepsilon$ $C \rightarrow \varepsilon$
B, C	A, B, C	$A \rightarrow BC$
A, B, C	A, B, C	—

~~∴ The resulting grammar, $G_1 = (V_1, T_1, P_1, S)$~~

∴ The nullable variables are $\{A, B, C\}$

Step2: Find Resulting Productions (P')

Productions (P)	Resulting Productions (P')
$S \rightarrow ABCa \mid bD$	$S \rightarrow ABCa \mid BCa \mid ACa \mid ABa \mid Ca \mid Aa \mid Ba \mid a \mid bD$
$A \rightarrow BC \mid b$	$A \rightarrow BC \mid C \mid B \mid b$
$B \rightarrow b \mid \varepsilon$	$B \rightarrow b$
$C \rightarrow c \mid \varepsilon$	$C \rightarrow c$
$D \rightarrow d$	$D \rightarrow d$

Q2

Ste

∴ The resulting grammar $G' = (V', T', P', S)$

$V' = \{S, A, B, C, D\}$

$T' = \{a, b, c, d\}$

$P' = \{ S \to ABCa \mid BCa \mid ACa \mid ABa \mid Ca \mid Aa \mid Ba \mid a \mid bD,$

$A \to BC \mid C \mid B \mid b$

$B \to b$

$C \to c$

$D \to d \}$

S is the start symbol.

Q2) Eliminate all ε-productions from the grammar

$S \to BAAB$

$A \to 0A2 \mid 2A0 \mid \varepsilon$

$B \to AB \mid 1B \mid \varepsilon$

Step 1: Find nullable variables

O.V	N.V	Productions
ϕ	A, B	$A \to \varepsilon$ $B \to \varepsilon$
A, B	S, A, B	$S \to BAAB$ $B \to AB$
S, A, B.	S, A, B	—

∴ nullable variables are $\{S, A, B\}$

Step 2: Resulting productions (P')

Productions (P)	Resulting productions (P')
$S \rightarrow BAAB$	$S \rightarrow BAAB \mid AAB \mid BAB \mid BAA \mid AB \mid BB \mid BA \mid AA \mid B \mid A$
$A \rightarrow 0A2 \mid 2A0 \mid \varepsilon$	$A \rightarrow 0A2 \mid 02 \mid 2A0 \mid 20$
$B \rightarrow AB \mid 1B \mid \varepsilon$	$B \rightarrow AB \mid A \mid B \mid 1B \mid 1$

$\therefore$ The resulting grammar $G' = (V', T', P', S)$

$V' = \{S, A, B\}$

$T' = \{0, 1, 2\}$

$$P' = \{ S \rightarrow BAAB \mid AAB \mid BAB \mid BAA \mid AB \mid BB \mid BA \mid AA \mid B \mid A,$$

$$A \rightarrow 0A2 \mid 02 \mid 2A0 \mid 20$$

$$B \rightarrow AB \mid A \mid B \mid 1B \mid 1 \}$$

S is the start symbol.

Q1) Eliminate all unit productions from the following grammar.

a variable implies a vari...

$S \rightarrow AB$

$A \rightarrow a$

$B \rightarrow C \mid b$

$C \rightarrow D$

$D \rightarrow E \mid bC$

$E \rightarrow d \mid Ab$

Non-unit productions

$S \to AB$
$A \to a$
$B \to b$
$D \to bC$
$E \to d \mid Ab$

Unit productions

$B \to C$
$C \to D$
$D \to E$

Dependency graph for unit productions

$$B \longrightarrow C \longrightarrow D \longrightarrow E$$

$B \to b$ $\quad$ $C \to bc$ $\quad$ $D \to bc$ $\quad$ $E \to d \mid Ab$
$B \to bc$ $\quad$ $C \to d \mid Ab$ $\quad$ $D \to d \mid Ab$
$B \to d \mid Ab$

$\therefore$ The resulting grammar $G' = (V', T', P', S)$

$V' = \{S, A, B, C, D, E\}$
$T' = \{a, b, d\}$
$P' = \{ S \to AB$
$A \to a$
$B \to b \mid bc \mid d \mid Ab$
$C \to bc \mid d \mid Ab$
$D \to bc \mid d \mid Ab$
$E \to d \mid AB \}$

S is the start symbol.

Q2) Eliminate all unit productions from the grammar

$S \to A0 \mid B$
$B \to A \mid 11$
$A \to 0 \mid 12 \mid B$

Non unit productions

$S \to A0$

$B \to 11$

$A \to 0 | 12$

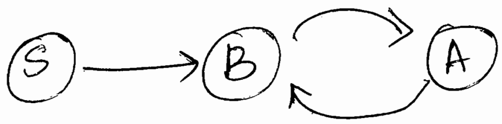
unit productions

$S \to B$

$B \to A$

$A \to B$

Dependency graph



$S \to A0$ $\quad B \to 11$ $\quad A \to 0|12$

$S \to 11$ $\quad B \to 0|12$ $\quad A \to 11$

$S \to 0|12$

$\therefore$ The resulting grammar $G' = (V', T', P', S)$.

$V' = \{S, A, B\}$

$T' = \{0, 1, 2\}$

$P' = \{ S \to A0 | 11 | 0 | 12$

$A \to 0 | 12 | 11$

$B \to 11 | 0 | 12 \}$

S is the start symbol.

Q3) Eliminate all unit productions of the grammar

$S \to Aa | B | Ca$

$B \to aB | b$

$C \to Db | D$

$D \to E | d$

$E \to ab$

Nonunit productions

$S \rightarrow Aa \mid Ca$

$B \rightarrow aB \mid b$

$C \rightarrow Db$

$D \rightarrow d$

$E \rightarrow ab$

Unit productions

$S \rightarrow B$

$C \rightarrow D$

$D \rightarrow E$

Dependency graph

$S \longrightarrow B \qquad C \longrightarrow D \longrightarrow E$

$S \rightarrow Aa \mid Ca$ $\quad B \rightarrow aB \mid b$ $\quad C \rightarrow Db$ $\quad D \rightarrow d$ $\quad E \rightarrow ab$

$S \rightarrow aB \mid b$ $\quad C \rightarrow d$ $\quad D \rightarrow ab$

$C \rightarrow ab$

$\therefore$ The resulting grammar $G' = (V', T', P', S)$

$V' = \{S, A, B, C, D, E\}$

$T' = \{a, b\}$

$P' = \{ S \rightarrow Aa \mid Ca \mid aB \mid b$

$B \rightarrow aB \mid b$

$C \rightarrow Db \mid d \mid ab$

$D \rightarrow d \mid ab$

$E \rightarrow ab \}$

S is the start symbol.

Note:

* All undesirable productions can be eliminated by removing,

- ε - productions
- unit -productions.
- useless symbols & useless productions

12/10/18

Normal forms
- → CHOMSKY Normal Form (CNF)
- → Greibach Normal Form (GNF)

CHOMSKY Normal Form (CNF) :-

Let $G = (V, T, P, S)$ be a CFG. The grammar G is said to be in CNF if all the productions are of the form,

$A \longrightarrow BC$

(or)

$A \longrightarrow a$

where A, B and $C \in V$ and $a \in T$.

Note that if a grammar is in CNF the righthand side of the production should contain 2 symbols (or) one symbol.
If there are 2 symbols on righthand side those 2 symbols must be non terminals and if there is only one symbol that must be a terminal.

Q1) Consider the grammar

$S \rightarrow 0A1 \mid 1B$

$A \rightarrow 0AA \mid 1S \mid 1$

$B \rightarrow 1BB \mid 0S \mid 0$ obtain the grammar in CNF.

Note: In case if the given grammar has ϵ-productions (or) unit-productions perform the following operations one after the other,

- Eliminate all ϵ-productions.
- Eliminate all unit-productions.
- Obtain the grammar in CNF.

Step 1:

$S \rightarrow B_0 A B_1 \mid B_1 B$

$A \rightarrow B_0 A A \mid B_1 S \mid 1$

$B \rightarrow B_1 B B \mid B_0 S \mid 0$

$B_0 \rightarrow 0$

$B_1 \rightarrow 1$

$\therefore$ The resulting grammar

$G_1 = (V_1, T_1, P_1, S)$

$V_1 = \{S, A, B, B_0, B_1\}$

$T_1 = \{0, 1\}$

$P_1 = \{S \rightarrow B_0 A B_1 \mid B_1 B$

$A \rightarrow B_0 A A \mid B_1 S \mid 1$

$B \rightarrow B_1 B B \mid B_0 S \mid 0, \; B_0 \rightarrow 0, \; B_1 \rightarrow 1\}$

S is the start symbol

Step 2:

$S \rightarrow B_0 D_1 \mid B_1 B$

$A \rightarrow B_0 D_2 \mid B_1 S \mid 1$

$B \rightarrow B_1 D_3 \mid B_0 S \mid 0$

$B_0 \rightarrow 0$

$B_1 \rightarrow 1$

$D_1 \rightarrow A B_1$

$D_2 \rightarrow A A$

$D_3 \rightarrow B B$

$\therefore$ The resulting grammar $G' = (V', T', P', S)$

$V' = \{S, A, B, B_0, B_1, D_1, D_2, D_3\}$

$T' = \{0, 1\}$

$P' = \{S \rightarrow B_0 D_1 \mid B_1 B, \; A \rightarrow B_0 D_2 \mid B_1 S \mid 1$

$B \rightarrow B_1 D_3 \mid B_0 S \mid 0, \; B_0 \rightarrow 0, \; B_1 \rightarrow 1$

$D_1 \rightarrow A B_1, \; D_2 \rightarrow A A, \; D_3 \rightarrow B B\}$

S is the start symbol

Greibach Normal Form (GNF) :-

Let $G = (V, T, P, S)$ be a CFG. The CFG is said to be in GNF if all the productions are of the form,

$$\boxed{A \rightarrow a\alpha}$$

where $a \in T$ and $\alpha \in V^*$ ie,
the first symbol on the righthand side of the production must be a terminal and it can be followed by zero (or) more variables.

Procedure to obtain the grammar in GNF :-

Step 1: Obtain the grammar in CNF.

Step 2: Rename the nonterminals to $A_1, A_2, A_3, \ldots$

Step 3: using substitution method obtain the productions to the form,

$$A_i \rightarrow A_j \alpha \quad \text{for} \quad i < j$$

$$\alpha \in V^*.$$

Note: If all the productions are in this manner, the no. of steps will be reduced while converting.

Step 4: After substitution if a grammar has left recursion we should eliminate left recursion.

Step 5: It may be necessary to apply step 3 and/or step 4 more than once to get the grammar in GNF.

13/10/18

Q1) Consider the grammar,

$$S \rightarrow AB1 \mid 0$$
$$A \rightarrow 00A \mid B$$
$$B \rightarrow 1A1$$

i) Eliminate all ε- productions
ii) Eliminate all unit productions
iii) Obtain the grammar in CNF
iv) Finally convert grammar into GNF

Eliminate unit-productions

$S \rightarrow AB1 \mid 0$
$A \rightarrow 00A \mid 1A1$ $\qquad (B \rightarrow 1A1)$
$B \rightarrow 1A1$

Converting into CNF

Step1:
$S \rightarrow ABA_1 \mid 0$
$A \rightarrow A_0A_0A \mid A_1AA_1$
$B \rightarrow A_1AA_1$
$A_0 \rightarrow 0$
$A_1 \rightarrow 1$

∴ the resulting grammar

$G_1 = (V_1, T_1, P_1, S)$

$V_1 = \{S, A, B, A_0, A_1\}$

$T_1 = \{0, 1\}$

$P_1 = \{ S \rightarrow ABA_1 \mid 0$
$A \rightarrow A_0A_0A \mid A_1AA_1$
$B \rightarrow A_1AA_1$
$A_0 \rightarrow 0$
$A_1 \rightarrow 1 \}$

S is the start symbol.

Step2:

$S \rightarrow AD_1 \mid 0$
$A \rightarrow A_0D_2 \mid A_1D_3$
$B \rightarrow A_1D_3$
$A_0 \rightarrow 0$
$A_1 \rightarrow 1$
$D_1 \rightarrow BA_1$
$D_2 \rightarrow A_0A$
$D_3 \rightarrow AA_1$

$\therefore$ The resulting grammar

$G' = (V', T', P', S)$

$V' = \{S, A, B, A_0, A_1, D_1, D_2, D_3\}$

$T' = \{0, 1\}$

$P' = \{S \rightarrow AD_1 | 0, \quad A \rightarrow A_0D_2 | A_1D_3$

$B \rightarrow A_1D_3, \quad A_0 \rightarrow 0, \quad A_1 \rightarrow 1,$

$D_1 \rightarrow BA_1, \quad D_2 \rightarrow A_0A, \quad D_3 \rightarrow AA_1\}$

S is the start symbol

Let $S = A_1$

$A = A_2$

$B = A_3$

$A_0 = A_4$

$A_1 = A_5$

$D_1 = A_6$

$D_2 = A_7$

$D_3 = A_8$

Now, the Grammar can be re-written as

$A_1 \rightarrow A_2A_6 | 0$

$A_2 \rightarrow A_4A_7 | A_5A_8$

$A_3 \rightarrow A_5A_8$

$A_4 \rightarrow 0$

$A_5 \rightarrow 1$

$A_6 \rightarrow A_3A_5$

$A_7 \rightarrow A_4A_2$

$A_8 \rightarrow A_2A_5$

Consider A_3-Production

$A_3 \rightarrow A_5 A_8$

$\therefore \boxed{A_3 \rightarrow 1A_8}$

Consider A_2-Productions

$A_2 \rightarrow A_4 A_7 \mid A_5 A_8$

$\therefore \boxed{A_2 \rightarrow 0A_7 \mid 1A_8}$

Consider A_1-Productions

$A_1 \rightarrow A_2 A_6 \mid 0$

$\therefore A_1 \rightarrow (0A_7 \mid 1A_8) A_6 \mid 0$

$\therefore \boxed{A_1 \rightarrow 0A_7A_6 \mid 1A_8A_6 \mid 0}$

Consider A_6-Productions

$A_6 \rightarrow A_3 A_5$

$\therefore \boxed{A_6 \rightarrow 1A_8A_5}$

Consider A_7-Productions

$A_7 \rightarrow A_4 A_2$

$\therefore \boxed{A_7 \rightarrow 0A_2}$

Consider A_8-Productions

$A_8 \rightarrow A_2 A_5$

$A_8 \rightarrow (0A_7 \mid 1A_8) A_5$

$\therefore \boxed{A_8 \rightarrow 0A_7A_5 \mid 1A_8A_5}$

∴ Final grammar which is in GNF is,

$G' = (V', T', P', S)$

$V' = \{A_1, A_2, A_3, A_4, A_5, A_6, A_7, A_8\}$

$T' = \{0, 1\}$

$P' = \{ A_1 \rightarrow 0A_7A_6 \mid 1A_8A_6 \mid 0,$

$A_2 \rightarrow 0A_7 \mid 1A_8,$

$A_3 \rightarrow 1A_8,$

$A_4 \rightarrow 0$

$A_5 \rightarrow 1$

$A_6 \rightarrow 1A_8A_5$

$A_7 \rightarrow 0A_2$

$A_8 \rightarrow 0A_7A_5 \mid 1A_8A_5 \}.$

A_1 is the start symbol.

Q2) Convert the following grammar into GNF:

$A \rightarrow BC$

$B \rightarrow CA \mid b$

$C \rightarrow AB \mid a$

Here, there is no ε-productions, unit-productions and it is in CNF.

Let $A = A_1$
$B = A_2$
$C = A_3$
} Renaming variables, starting with A_1.

Now, $A_1 \rightarrow A_2A_3$

$A_2 \rightarrow A_3A_1 \mid b$

$A_3 \rightarrow A_1A_2 \mid a$

Consider A_3 Productions

$A_3 \rightarrow A_1 A_2 \mid a$

$\rightarrow A_2 A_3 A_2 \mid a$

$\rightarrow (A_3 A_1 \mid b) A_3 A_2 \mid a$

$\therefore \boxed{A_3 \rightarrow A_3 \underbrace{A_1 A_3 A_2}_{\alpha_1} \mid \underbrace{b A_3 A_2}_{\beta_1} \mid \underbrace{a}_{\beta_2}}$

Left Recursion

$\boxed{A \rightarrow A\alpha_i \mid \beta_j}$

$\Downarrow$

$A \rightarrow \beta_j Z$

$Z \rightarrow \alpha_i Z \mid \varepsilon$

$\Downarrow$

$$\boxed{\begin{array}{l} A \rightarrow \beta_j Z \mid \beta_j \\ Z \rightarrow \alpha_i Z \mid \alpha_i \end{array}}$$

Now, $A_3 \rightarrow b A_3 A_2 Z \mid b A_3 A_2 \mid aZ \mid a$

$Z \rightarrow A_1 A_3 A_2 Z \mid A_1 A_3 A_2$

$A_2 \rightarrow (b A_3 A_2 Z \mid b A_3 A_2 \mid aZ \mid a) A_1 \mid b$

$A_2 \rightarrow b A_3 A_2 Z A_1 \mid b A_3 A_2 A_1 \mid aZA_1 \mid aA_1 \mid b$

$A_1 \rightarrow (b A_3 A_2 Z A_1 \mid b A_3 A_2 A_1 \mid aZA_1 \mid aA_1 \mid b) A_3$

$A_1 \rightarrow b A_3 A_2 Z A_1 A_3 \mid b A_3 A_2 A_1 A_3 \mid aZA_1A_3 \mid aA_1A_3 \mid bA_3$

$\therefore Z \rightarrow (b A_3 A_2 Z A_1 A_3 \mid b A_3 A_2 A_1 A_3 \mid aZA_1A_3 \mid aA_1A_3 \mid bA_3 \mid b A_3 A_2 Z \mid b A_3 A_2 \mid aZ \mid a \mid b A_3 A_2 Z A_1 \mid b A_3 A_2 A_1 \mid aZA_1 \mid aA_1 \mid b) Z \mid (b A_3 A_2 Z A_1 A_3 \mid b A_3 A_2 A_1 A_3 \mid aZA_1A_3 \mid aA_1A_3 \mid bA_3 \mid$ complete

$\therefore Z \rightarrow (bA_3A_2ZA_1A_3 |$

31/10/18

*Pushdown Automata (PDA)

$$M = (Q, \Sigma, \Gamma, \delta, q_0, z_0, F)$$

$$\delta : Q \times \Sigma \times \Gamma \longrightarrow Q \times \Gamma^*$$

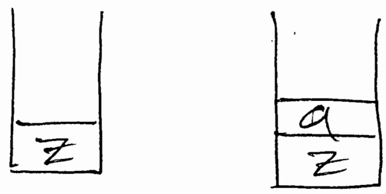
$$\delta(q_0, a, z_0) \Longrightarrow (q_1, az_0 | \epsilon | z_0 | r)$$

Q1) $\delta(p, a, z) = (q, az)$

$$p \xrightarrow{(a,z)|az} q$$

(Push operation)

Before After



Q2) $\delta(p, a, z) = (q, \epsilon)$

$$p \xrightarrow{(a,z)|\epsilon} q$$

(pop operation)

Before After

Q3) $\delta(p, a, z) = (q, r)$

$$p \xrightarrow{(a,z)|r} q$$

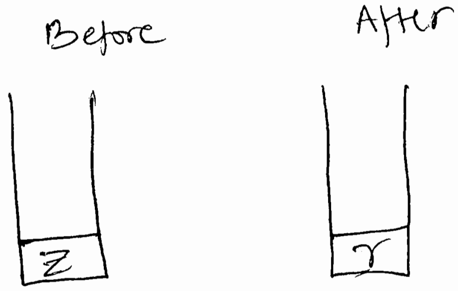
Before After

(replacing z by r)

Q4) $\delta(p, \varepsilon, z) = (q, r)$

$$p \xrightarrow{(\varepsilon, z) \mid r} q$$

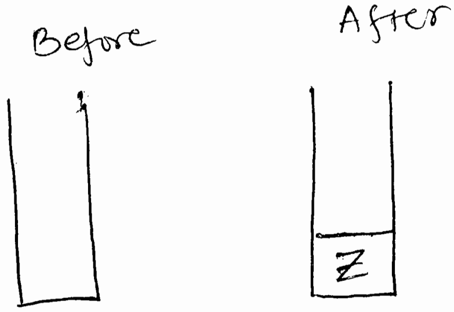
(replacing z by r)



Q5) $\delta(p, \varepsilon, \varepsilon) = (q, z)$

$$p \xrightarrow{(\varepsilon, \varepsilon) \mid z} q$$

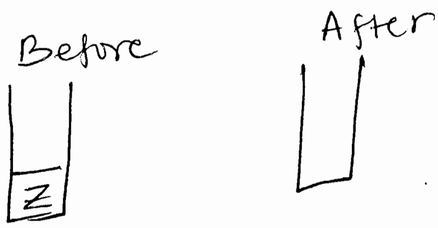
(pushing z)



Q6) $\delta(p, \varepsilon, z) = (q, \varepsilon)$

$$p \xrightarrow{(\varepsilon, z) \mid \varepsilon} q$$

(poping z)



*Acceptance of language PDA:-

There are 2 cases where in a string w is accepted by a PDA,

i) Get the final state from the start state.

ii) Get an empty stack from the start state.

Q1) Obtain a PDA to accept the language $L(M) = \{ wcw^R \mid w \in \{a, b\}^* \}$ where w^R is reverse of w by a final state?

Solution:

$\omega = aab$
$\omega^R = baa$
$\omega c \omega^R = aabcbaa$

$\delta(q_0, a, z_0) = (q_0, az_0)$
$\delta(q_0, b, z_0) = (q_0, bz_0)$

$\delta(q_0, a, a) = (q_0, aa)$
$\delta(q_0, b, a) = (q_0, ba)$
$\delta(q_0, a, b) = (q_0, ab)$
$\delta(q_0, b, b) = (q_0, bb)$

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

$\delta(q_0, c, z_0) = (q_1, z_0)$

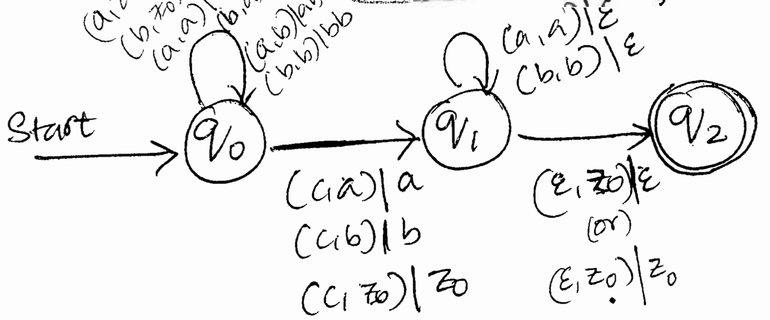
$\delta(q_1, a, a) = (q_1, \varepsilon)$
$\delta(q_1, b, b) = (q_1, \varepsilon)$

$\delta(q_1, \varepsilon, z_0) = (q_2, z_0)$ (or) (q_2, ε)

$\therefore$ The resulting PDA is, $M = (Q, \Sigma, \Gamma, \delta, q_0, z_0, F)$

$Q = \{q_0, q_1, q_2\}$
$\Sigma = \{a, b, c\}$
$\Gamma = \{z_0, a, b\}$

transition diagram.



$q_0 \in Q$ is the start state.

$Z_0 \in \Gamma$ is the start symbol of stack. (Initial stack symbol)

$F \subseteq Q = \{q_2\}$ is the final state

15/11/18

To accept the string: aabcbaa

Initial ID

$(q_0, aabcbaa, Z_0) \vdash (q_0, abcbaa, aZ_0)$
$\vdash (q_0, bcbaa, aaZ_0)$
$\vdash (q_0, cbaa, baaZ_0)$
$\vdash (q_1, baa, baaZ_0)$
$\vdash (q_1, aa, aaZ_0)$
$\vdash (q_1, a, aZ_0)$
$\vdash (q_1, \varepsilon, Z_0)$
$\vdash (q_2, \varepsilon, Z_0)$
Final ID

$\therefore$ The string aabcbaa is accepted by PDA.

To reject the string: aabcbab

Initial ID

$(q_0, aabcbab, Z_0) \vdash (q_0, abcbab, aZ_0)$
$\vdash (q_0, bcbab, aaZ_0)$
$\vdash (q_0, cbab, baaZ_0)$
$\vdash (q_1, bab, baaZ_0)$
$\vdash (q_1, ab, aaZ_0)$
$\vdash (q_1, b, aZ_0)$
Struct

not possible to reach final state

Hence string aabcbab is rejected by PDA.

16/4/18

Q2) Obtain a PDA to accept the language $L = \{a^n b^n \mid n \geq 1\}$ by a final state.

Soln:

$\delta(q_0, a, Z_0) = (q_0, aZ_0)$

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

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

$\delta(q_1, b, a) = (q_1, \varepsilon)$

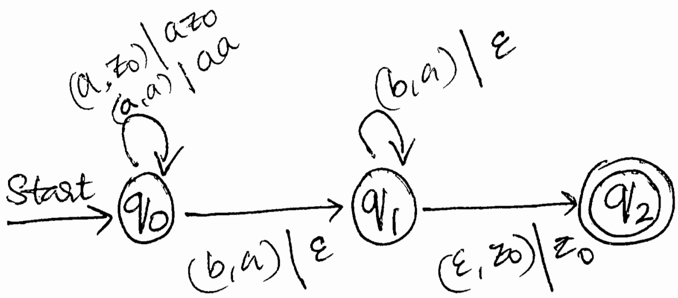
$\delta(q_1, \varepsilon, Z_0) = (q_2, Z_0)$

$\therefore$ The resulting PDA is, $M = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$

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

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

$\Gamma = \{a, Z_0\}$



$q_0 \in Q$ is the start state.

$Z_0 \in \Gamma$ is the start symbol of stack.

$F \subseteq Q = \{q_2\}$ is the final state.

To accept the string : aabb

Initial ID

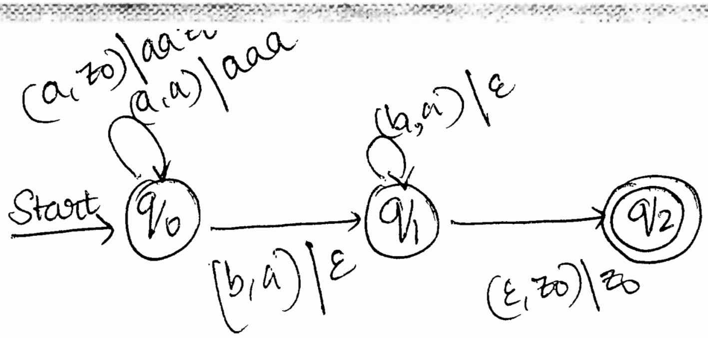
$(q_0, aabb, Z_0) \vdash (q_0, abb, aZ_0)$
$\vdash (q_0, bb, aaZ_0)$
$\vdash (q_1, b, aZ_0)$
$\vdash (q_1, \varepsilon, Z_0)$
$\vdash (q_2, Z_0)$ Final ID

∴ The string aabb is accepted by PDA.

Q3) Obtain a PDA to accept the language $L = \{a^n b^{2n} \mid n \geq 1\}$ by a final state.

Soln:
$\delta(q_0, a, Z_0) = (q_0, aaZ_0)$
$\delta(q_0, a, a) = (q_0, aaaa)$
$\delta(q_0, b, a) = (q_1, \varepsilon)$
$\delta(q_1, b, a) = (q_1, \varepsilon)$
$\delta(q_1, \varepsilon, Z_0) = (q_2, Z_0)$

∴ The resulting PDA is $M = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$
$Q = \{q_0, q_1, q_2\}$
$\Sigma = \{a, b\}$
$\Gamma = \{a, Z_0\}$



$q_0 \in Q$ is the start state.

$Z_0 \in \Gamma$ is the start symbol of stack.

$F \subseteq Q = \{q_2\}$ is the final state.

To accept string: aabbbb

Initial ID

$(q_0, aabbbb, Z_0) \vdash (q_0, abbbb, aaZ_0)$

$\vdash (q_0, bbbb, aaaaZ_0)$

$\vdash (q_1, bbb, aaaZ_0)$

$\vdash (q_1, bb, aaZ_0)$

$\vdash (q_1, b, aZ_0)$

$\vdash (q_1, \varepsilon, Z_0)$

$\vdash (q_2, Z_0)$ Final ID

$\therefore$ The string aabbbb is accepted by PDA.

Q4) Obtain a PDA to accept the language $L = \{a^{2n} b^n \mid n \geq 1\}$ by a final state.

Soln:

$\delta(q_0, a, Z_0) = (q_1, aZ_0)$

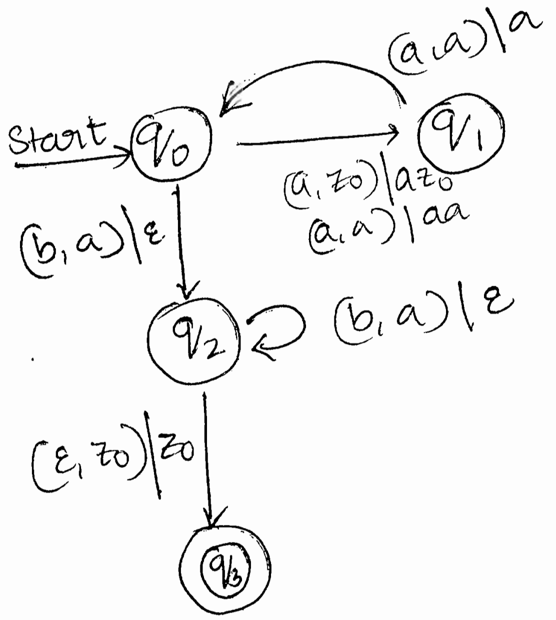
$\delta(q_1, a, a) = (q_0, a)$

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

$\delta(q_0, b, a) = (q_2, \varepsilon)$

$\delta(q_2, b, a) = (q_2, \varepsilon)$

$\delta(q_2, \varepsilon, Z_0) = (q_3, Z_0)$



∴ The resulting PDA

$M = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$

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

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

$\Gamma = \{a, Z_0\}$

$q_0 \in Q$ is the start state

$Z_0 \in \Gamma$ is the start symbol of stack.

$F \subseteq Q = \{q_3\}$ is the final state.

To accept string: aab

Initial ID

$(q_0, aab, z_0) \vdash (q_1, ab, az_0)$

$\vdash (q_0, b, aaz_0)$

$\vdash (q_2, \varepsilon, z_0)$

$\vdash (q_3, z_0)$ Final ID

$\therefore$ The string aab is accepted by PDA

Q5) Obtain a PDA to accept the language $L(M) = \{w \mid w \in \{a, b\}^*$ and $n_a(w) = n_b(w)\}$ by a final state?

Solu:
$\delta(q_0, a, z_0) = (q_0, az_0)$

$\delta(q_0, b, z_0) = (q_0, bz_0)$

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

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

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

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

$\delta(q_0, \varepsilon, z_0) = (q_1, z_0)$

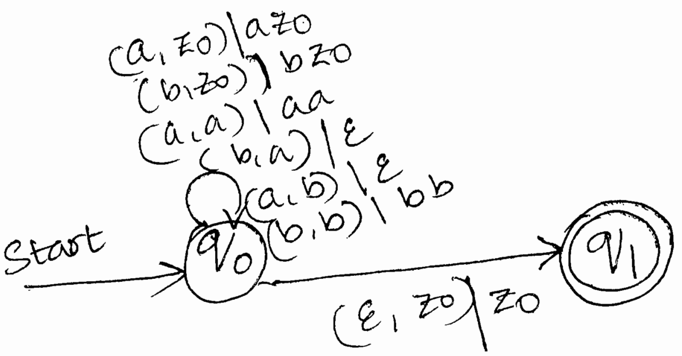
∴ The resulting PDA,

$M = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$

$Q = \{q_0, q_1\}$

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

$\Gamma = \{a, b, Z_0\}$



$q_0 \in Q$ is the start state.

$Z_0 \in \Gamma$ is the start symbol of stack.

$F \subseteq Q = \{q_1\}$ is the final state.

a

To accept string: bbaa

Initial ID

$(q_0, bbaa, Z_0) \vdash (q_0, baa, bZ_0)$

$\vdash (q_0, aa, bbZ_0)$

$\vdash (q_0, a, bZ_0)$

$\vdash (q_0, \varepsilon, Z_0)$

$\vdash (q_1, Z_0)$ Final ID

∴ The string bbaa is accepted by PDA.

Q6) Obtain a PDA to accept language $L(M) = \{w \mid w \in \{a,b\}^* \text{ and } n_a(w) > n_b(w)\}$ by a final state?

Soln:

$$\delta(q_0, a, z_0) = (q_0, az_0)$$
$$\delta(q_0, b, z_0) = (q_0, bz_0)$$

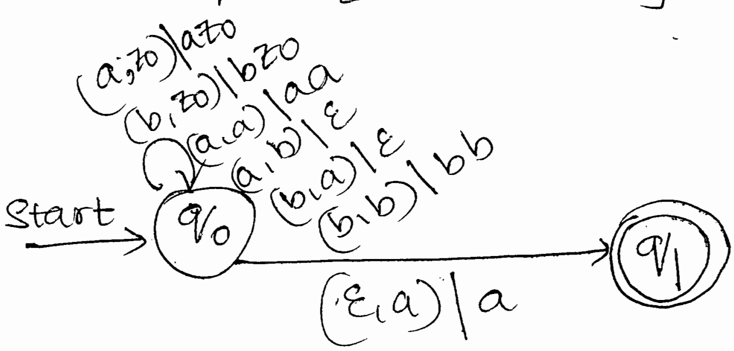
$$\delta(q_0, a, a) = (q_0, aa)$$
$$\delta(q_0, b, a) = (q_0, \varepsilon)$$
$$\delta(q_0, a, b) = (q_0, \varepsilon)$$
$$\delta(q_0, b, b) = (q_0, bb)$$
$$\delta(q_0, \varepsilon, a) = (q_1, a)$$

$\therefore$ The resulting PDA is $M = (Q, \Sigma, \Gamma, \delta, q_0, z_0, F)$

$Q = \{q_0, q_1\}$

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

$\Gamma = \{a, b, z_0\}$



$q_0 \in Q$ is the start state

$z_0 \in \Gamma$ is the start symbol of stack

$F \subseteq Q = \{q_1\}$ is the final state.

Q7) Obtain a PDA to accept the string of balanced paranthesis. The paranthesis is to be considered are (,), [,] .

ε is the min. string

Solu:

$$\delta(q_0, (, Z_0) = (q_1, (Z_0)$$
$$\delta(q_0, [, Z_0) = (q_1, [Z_0)$$

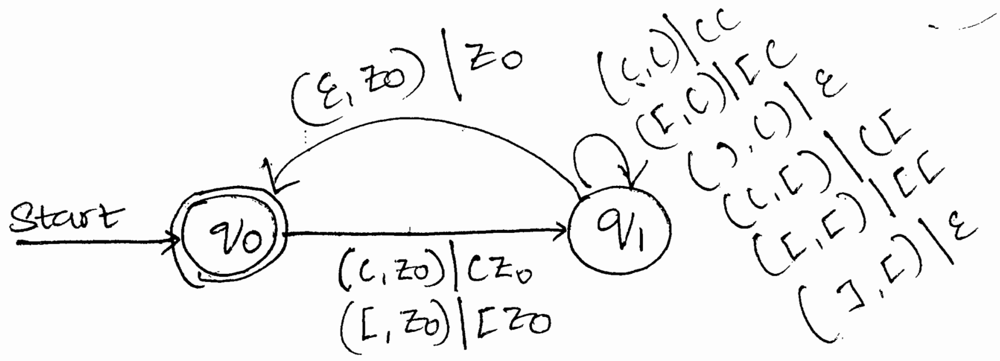
$$\delta(q_1, (, () = (q_1, (()$$
$$\delta(q_1, [, () = (q_1, [()$$
$$\delta(q_1,), () = (q_1, \varepsilon)$$

$$\delta(q_1, (, [) = (q_1, ([)$$
$$\delta(q_1, [, [) = (q_1, [[)$$
$$\delta(q_1,], [) = (q_1, \varepsilon)$$

$$\delta(q_1, \varepsilon, Z_0) = (q_0, Z_0)$$

$\therefore$ The resulting PDA is $M = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$

$$Q = \{q_0, q_1\}$$
$$\Sigma = \{(, [,),]\}$$
$$\Gamma = \{Z_0, (, [\}$$



$q_0 \in Q$ is the start state.
$Z_0 \in \Gamma$ is the start symbol of stack.
$F \subseteq Q = \{q_0\}$ is the final state.

PDA → DPDA

PDA → NDPDA

CNF

A → BC

or

A → a

GNF

A → aα

Definition :-

Let $M = (Q, \Sigma, \Gamma, \delta, q_0, z_0, F)$ in PDA, the PDA is deterministic if,

(i) $\delta(q, a, Z)$ has only one element

(ii) $\delta(q, \varepsilon, Z)$ is not empty

then $\delta(q, a, Z)$ should be empty.

Both the conditions should be satisfied for the PDA to be deterministic. If one of the conditions fails the PDA is non deterministic.

CFG to PDA

(1) For the Grammar

$S \rightarrow aABC$

$A \rightarrow aB \mid a$

$B \rightarrow bA \mid b$

$C \rightarrow a$

Obtain the corresponding PDA.

Note:

- To obtain CFG to PDA, we must convert the given CFG which is in GNF to PDA using some steps.
- CFG must be in GNF to convert CFG to PDA.

solution:

Step 1:

$\delta(q_0, \varepsilon, Z_0) = (q_1, SZ_0)$

Step 2:

Productions	Transitions
$A \to a\alpha$	$\delta(q_1, a, A) = (q_1, \alpha)$
$S \to aABC$	$\delta(q_1, a, S) = (q_1, ABC)$
$A \to aB$	$\delta(q_1, a, A) = (q_1, B)$
$A \to a$	$\delta(q_1, a, A) = (q_1, \varepsilon)$
$B \to bA$	$\delta(q_1, b, B) = (q_1, A)$
$B \to b$	$\delta(q_1, b, B) = (q_1, \varepsilon)$
$C \to a$	$\delta(q_1, a, C) = (q_1, \varepsilon)$

Step 3:

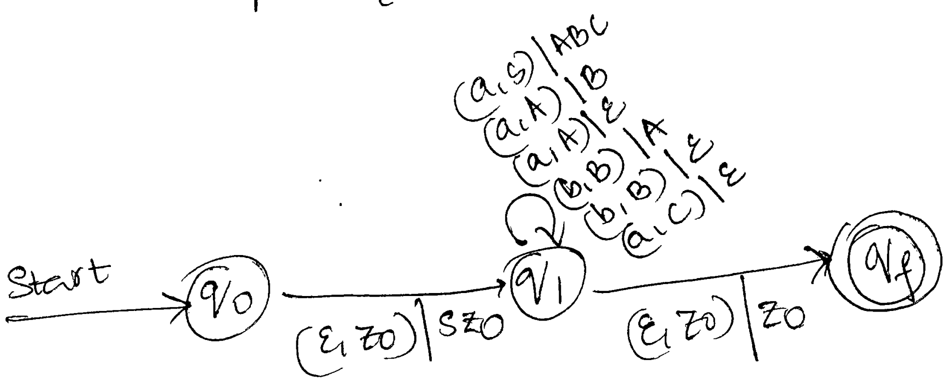
$\delta(q_1, \varepsilon, Z_0) = (q_f, Z_0)$

$\therefore$ The resulting PDA is $M = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$

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

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

$\Gamma = \{Z_0, S, A, B, C\}$



$q_0 \in Q$ is the start state

$Z_0 \in \Gamma$ is the start symbol of stack.

$F \subseteq Q = \{q_f\}$ is the final state.

Derivation

$$S \Rightarrow aABC$$
$$\Rightarrow aaBBC$$
$$\Rightarrow aabBC$$
$$\Rightarrow aabbC$$
$$\Rightarrow aabba$$

$$\therefore \boxed{S \overset{+}{\Rightarrow} aabba}$$

To accept : aabba

Initial ID

$(q_0, aabba, Z_0) \vdash (q_1, aabba, SZ_0)$

$\vdash (q_1, abba, ABCZ_0)$

$\vdash (q_1, bba, BBCZ_0)$

$\vdash (q_1, ba, BCZ_0)$

$\vdash (q_1, a, CZ_0)$

$\vdash (q_1, \varepsilon, Z_0)$

$\vdash (q_f, \varepsilon, Z_0)$

is the final ID.

$\therefore$ the string aabba is accepted by PDA.

(2) For the Grammar

$S \to aABB \mid aAA$

$A \to aBB \mid a$

$B \to bBB \mid A$

$C \to a$

Obtain the corresponding PDA.

soln: Replace production of B,

$S \to aABB \mid aAA$

$A \to aBB \mid a$

$B \to bBB \mid aBB \mid a$

$C \to a$

Step 1:

$\delta(q_0, \varepsilon, Z_0) = (q_1, SZ_0)$

Step 2:

$\delta(q_1, a, S) = (q_1, ABB)$

$\delta(q_1, a, S) = (q_1, AA)$

$\delta(q_1, a, A) = (q_1, BB)$

$\delta(q_1, a, A) = (q_1, \varepsilon)$

$\delta(q_1, b, B) = (q_1, BB)$

$\delta(q_1, a, B) = (q_1, BB)$

$\delta(q_1, a, B) = (q_1, \varepsilon)$

$\delta(q_1, a, C) = (q_1, \varepsilon)$

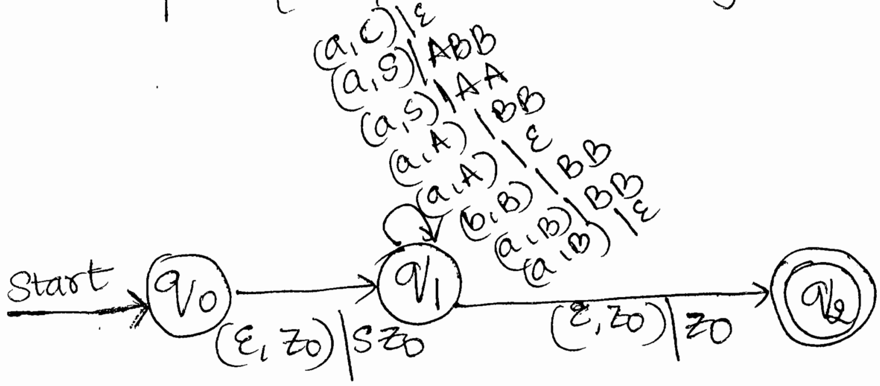
Step 3: $\delta(q_1, \varepsilon, Z_0) = (q_2, Z_0)$

∴ The resulting PDA, $M = (Q, \Sigma, \Gamma, \delta, q_0, z_0, F)$

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

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

$\Gamma = \{S, A, B, C, z_0\}$



$q_0 \in Q$ is the start state

$z_0 \in \Gamma$ is the start symbol of stack.

$F \subseteq Q = \{q_2\}$ is the final state.

Derivation:

$S \Rightarrow aAA$

$\Rightarrow aaBBA$

$\Rightarrow aaABA$

$\Rightarrow aaabBBA$

$\Rightarrow aaabABA$

$\Rightarrow aaabaAA$

$\Rightarrow aaabaaA$

$\Rightarrow aaabaaa$

$\therefore \boxed{S \overset{+}{\Rightarrow} aaabaaa}$

To accept string: aaabaaa

Initial ID.

$(q_0, aaabaaa, z_0) \vdash (q_1, aaabaaa, Sz_0) \vdash (q_1, aabaaa, ABBz_0)$
$\vdash (q_1, abaaa, BBBBz_0) \vdash (q_1, baaa, BBBz_0) \vdash$
$(q_1, aaa, BBBBz_0) \vdash (q_1, aa, BBBz_0) \vdash$
$(q_1, a, BBz_0) \vdash (q_1, \varepsilon, z_0)$
$\vdash (q_2, z_0)$ Final ID

∴ String aaabaaa is accepted by PDA.

Applications of GNF :-

① obtain the PDA to accept the language
$L = \{a^n b^n \mid n \geq 1\}$

Solu:

CFG

$S \rightarrow ab \mid aSb$

CFG to GNF

$S \rightarrow aB$

$S \rightarrow aSB$

$B \rightarrow b$

GNF to PDA : $\delta(q_0, \varepsilon, Z_0) = (q_1, SZ_0)$

$\delta(q_1, a, S) = (q_1, B)$

$\delta(q_1, a, S) = (q_1, SB)$

$\delta(q_1, b, B) = (q_1, \varepsilon)$

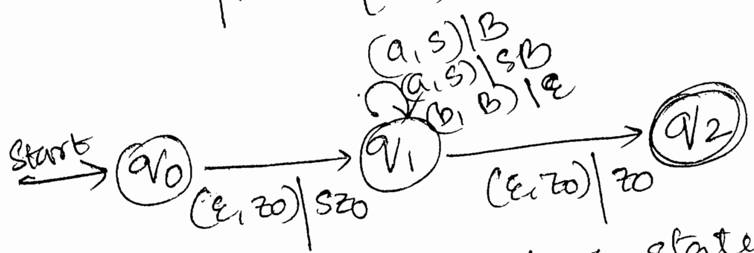
$\delta(q_1, \varepsilon, Z_0) = (q_2, Z_0)$

∴ The resulting PDA, $M = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$

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

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

$\Gamma = \{S, B, Z_0\}$



$q_0 \in Q$ is the start state

$Z_0 \in \Gamma$ is the start symbol of the stack

$F \subseteq Q = \{q_2\}$ is the final state.

② Obtain PDA to accept language
$L = \{ww^R \mid |w| \geq 1$ for $w \in \{a, b\}^*\}$

Soln:

CFG

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

CFG to GNF

$S \rightarrow aA \mid bB \mid aSA \mid bSB$

$A \rightarrow a$

$B \rightarrow b$

GNF to PDA: $\delta(q_0, \varepsilon, Z_0) = (q_1, SZ_0)$

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

$\delta(q_1, b, S) = (q_1, B)$

$\delta(q_1, a, S) = (q_1, SA)$

$\delta(q_1, b, S) = (q_1, SB)$

$\delta(q_1, a, A) = (q_1, \varepsilon)$

$\delta(q_1, b, B) = (q_1, \varepsilon)$

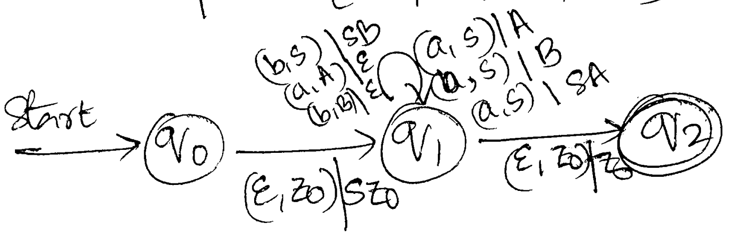
$\delta(q_1, \varepsilon, Z_0) = (q_2, Z_0)$

∴ The resulting PDA is, $M = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$

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

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

$\Gamma = \{S, A, B, Z_0\}$



$q_0 \in Q$ is the start state.

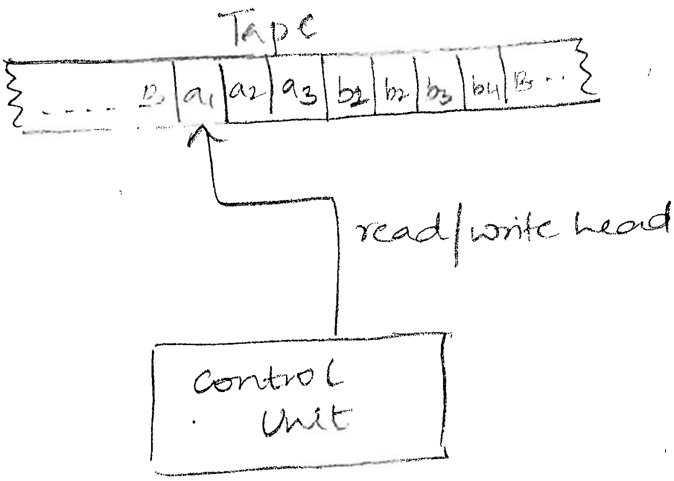
$Z_0 \in \Gamma$ is the start symbol of the stack

$F \subseteq Q = \{q_2\}$ is the final state.

17/11/18

Module - 4

Turing Machines (TM)



Definition:

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

$$\delta : Q \times \Gamma \longrightarrow Q \times \Gamma \times \{L, R\}$$

$$\downarrow$$

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

Q1) Obtain a turing machine to accept the language $L = \{0^n 1^n \mid n \geq 1\}$

B X X 0 0 Y Y 1 1 B

Soln:

$$\delta(q_0, 0) = (q_1, X, R)$$
$$\delta(q_1, 0) = (q_1, 0, R)$$
$$\delta(q_1, Y) = (q_1, Y, R)$$
$$\delta(q_1, 1) = (q_2, Y, L)$$
$$\delta(q_2, Y) = (q_2, Y, L)$$
$$\delta(q_2, 0) = (q_2, 0, L)$$
$$\delta(q_2, X) = (q_0, X, R)$$
$$\delta(q_0, Y) = (q_3, Y, R)$$
$$\delta(q_3, Y) = (q_3, Y, R)$$
$$\delta(q_3, B) = (q_4, B, R)$$

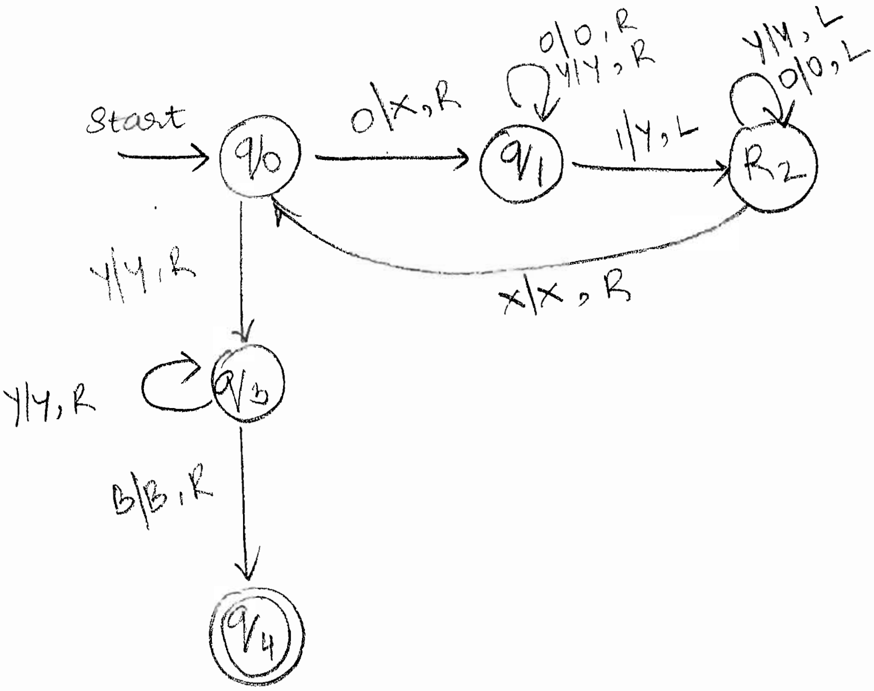
∴ The resulting TM is, $M = (Q, \Sigma, \Gamma, \delta, q_0, B, F)$

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

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

$\Gamma = \{0, 1, X, Y, B\}$

δ: transition function



$q_0 \in Q$ is the start state.

$B \in \Gamma$ is the initial symbol on the tape.

$F \subseteq Q = \{q_4\}$ is the final state.

transition table

δ	0	1	X	Y	B
$\rightarrow q_0$	(q_1, X, R)			(q_3, Y, R)	
q_1	$(q_1, 0, R)$	(q_2, Y, L)		(q_1, Y, R)	
q_2	$(q_2, 0, L)$		(q_0, X, R)	(q_2, Y, L)	
q_3				(q_3, Y, R)	(q_4, B, R)
$* q_4$					

To accept the string: 0011

Initial ID:

$(q_0, 0011) \vdash x q_1 011 \vdash x 0 q_1 11 \vdash$

$x 0 q_2 Y 1 \vdash x q_2 0 Y 1 \vdash$

$x q_0 0 Y 1 \vdash x x q_1 Y 1 \vdash$

$x x Y q_1 1 \vdash x x Y q_2 Y \vdash$

$x x q_2 Y Y \vdash x x q_0 Y Y \vdash$

$x x Y q_3 Y \vdash x x Y Y q_3 B \vdash$

$x x Y Y B q_4$ (Final ID)

∴ the string 0011 is accepted by TM.

To reject string: 001

Initial ID

$(q_0, 001) \vdash x q_1 01 \vdash x 0 q_1 1 \vdash x 0 q_2 Y \vdash$

$x q_2 0 Y \vdash x x q_0 Y \vdash x x Y q_3$

(struct)

∴ The string 001 is rejected (not accepted) by TM.

Q2) Obtain TM to accept the language

$$L = \{0^n 1^n 2^n \mid n \geq 1\}$$

XX00YY11ZZ22
↑
q_0

Soln:

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

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

$\delta(q_1, Y) = (q_1, Y, R)$

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

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

$\delta(q_2, Z) = (q_2, Z, R)$

$\delta(q_2, 2) = (q_3, Z, L)$

$\delta(q_3, Z) = (q_3, Z, L)$

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

$\delta(q_3, Y) = (q_3, Y, L)$

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

$\delta(q_3, X) = (q_0, X, R)$

$\delta(q_0, Y) = (q_4, Y, R)$

$\delta(q_4, Y) = (q_4, Y, R)$

$\delta(q_4, Z) = (q_5, Z, R)$

$\delta(q_5, Z) = (q_5, Z, R)$

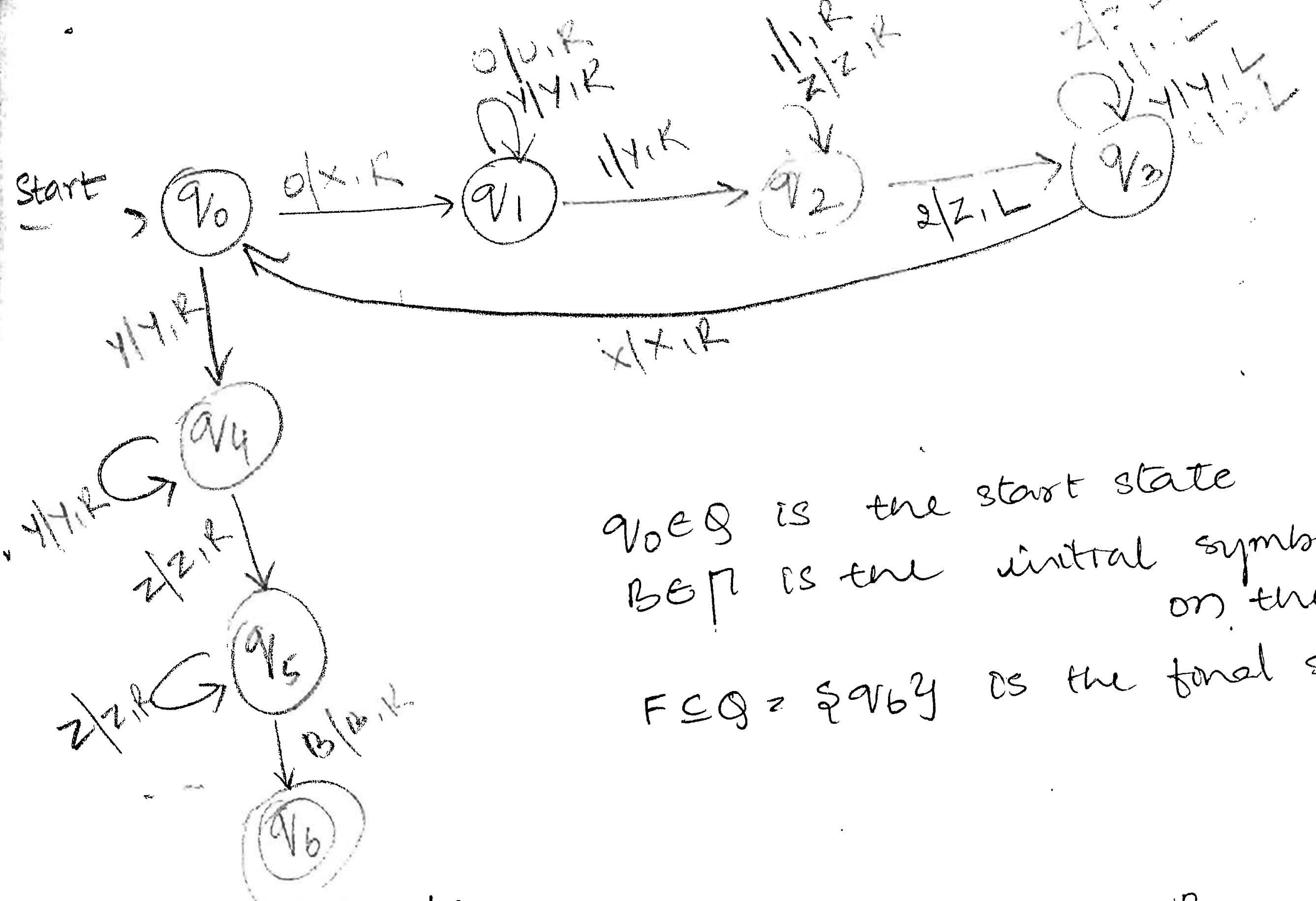
$\delta(q_5, B) = (q_6, B, R)$

$\therefore$ The resulting TM is, $M = (Q, \Sigma, \Gamma, \delta, q_0, B, F)$

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

$\Sigma = \{0, 1, 2\}$

$\Gamma = \{0, 1, 2, X, Y, Z, B\}$



$q_0 \in Q$ is the start state

$B \in \Gamma$ is the initial symbol on the tape

$F \subseteq Q = \{q_6\}$ is the final state.

Transition table

δ	0	1	2	X	Y	Z	B
$\rightarrow q_0$	(q_1, X, R)				(q_4, Y, R)		
q_1	$(q_1, 0, R)$	(q_2, Y, R)			(q_1, Y, R)		
q_2		$(q_2, 1, R)$	(q_3, Z, L)			(q_2, Z, R)	
q_3	$(q_3, 0, L)$	$(q_3, 1, L)$		(q_0, X, R)	(q_3, Y, L)	(q_3, Z, L)	
q_4					(q_4, Y, R)	(q_5, Z, R)	
q_5						(q_5, Z, R)	(q_6, B, R)
$* q_6$							

To accept string: 012

Initial ID

$\delta(q_0, 012) \vdash Xq_1 12 \vdash XYq_2 2 \vdash XYq_3 Z \vdash$
$Xq_3 YZ \vdash Xq_0 YZ \vdash XYq_4 Z \vdash$
$XYZq_5 \vdash XYZq_5 B \vdash XYZBq_6$
(Final ID)

$\therefore$ The string 012 is accepted by TM.

Q3) Obtain a TM to accept the string ω of a's & b's such that $n_a(\omega) = n_b(\omega)$.

Soln:

$\delta(q_0, B) = (q_f, B, R)$

xxaaYYbb

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

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

$\delta(q_1, Y) = (q_1, Y, R)$

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

$\delta(q_2, Y) = (q_2, Y, L)$

$\delta(q_2, a) = (q_2, a, L)$

$\delta(q_2, X) = (q_0, X, R)$

$\delta(q_0, b) = (q_3, X, R)$

xx bbYYaa

$\delta(q_3, b) = (q_3, b, R)$

$\delta(q_3, Y) = (q_3, Y, R)$

$\delta(q_3, a) = (q_4, Y, L)$

$\delta(q_4, Y) = (q_4, Y, L)$

$\delta(q_4, b) = (q_4, b, L)$

$\delta(q_4, X) = (q_0, X, R)$

$\delta(q_0, Y) = (q_0, Y, R)$

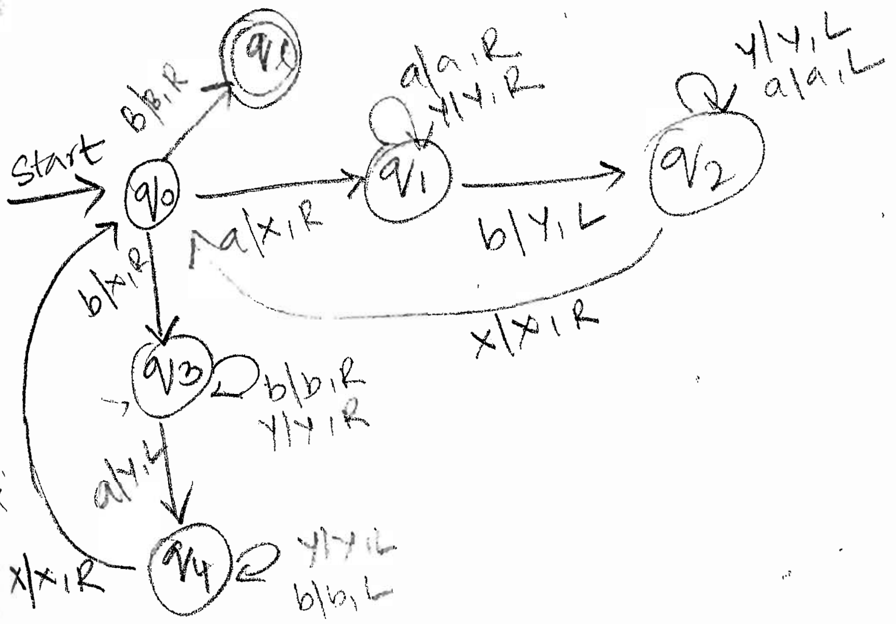
$\delta(q_0, B) = (q_f, B, R)$

$\therefore$ The resulting TM is, $M = (Q, \Sigma, \Gamma, \delta, q_0, B, F)$

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

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

$\Gamma = \{a, b, X, Y, B\}$



Transition table

δ	a	b	X	Y	B
$\rightarrow q_0$	(q_1, X, R)	(q_3, X, R)		(q_0, Y, R)	(q_f, B, R)
q_1	(q_1, a, R)	(q_2, Y, L)		(q_1, Y, R)	
q_2	(q_2, a, L)		(q_0, X, R)	(q_2, Y, L)	
q_3	(q_4, Y, L)	(q_3, b, R)		(q_3, Y, R)	
q_4		(q_4, b, L)	(q_0, X, R)	(q_4, Y, L)	
$* q_f$					

To accept the string: aabb

Initial ID

$(q_0, aabb) \vdash Xq_1abb \vdash Xaq_1bb \vdash Xaq_2Yb \vdash$
$Xq_2aYb \vdash Xq_0aYb \vdash XXq_1Yb \vdash$
$XXYq_1b \vdash XXYq_2Y \vdash XXq_2YY \vdash$
$XXq_0YY \vdash XXYq_0Y \vdash XXYYq_0B \vdash$
$XXYYBq_f$ (Final ID)

$\therefore$ The string aabb is accepted by TM.

84) obtain a TM to accept a palindrome with a's & b's of any length.

Soln:

$\delta(q_0, B) = (q_1, B, R)$

$\delta(q_1, a) = (q_2, B, R)$

right $\delta(q_2, a) = (q_2, a, R)$

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

$\delta(q_2, B) = (q_3, B, L)$

$\delta(q_3, a) = (q_4, B, L)$

* $\delta(q_3, B) = (q_f, B, R)$

when length of string is odd

moving left $\delta(q_4, a) = (q_4, a, L)$

$\delta(q_4, b) = (q_4, b, L)$

$\delta(q_4, B) = (q_1, B, R)$

$\delta(q_1, b) = (q_5, B, R)$

right $\delta(q_5, a) = (q_5, a, R)$

$\delta(q_5, b) = (q_5, b, R)$

$\delta(q_5, B) = (q_6, B, L)$

$\delta(q_6, b) = (q_4, B, L)$

* $\delta(q_6, B) = (q_f, B, R)$

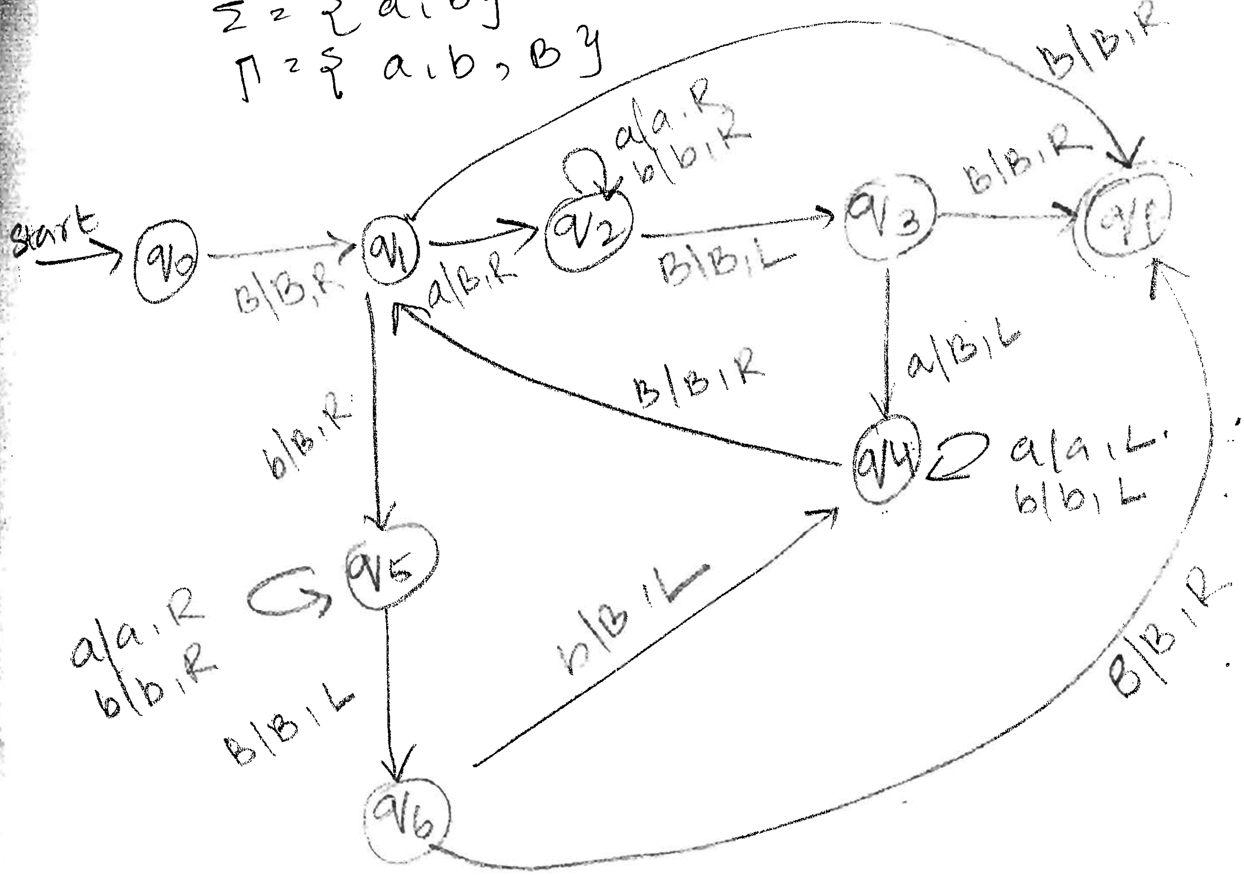
$\delta(q_1, B) = (q_f, B, R)$

∴ The resulting TM is, $M = (Q, \Sigma, \Gamma, \delta, q_0, B, F)$

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

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

$\Gamma = \{a, b, B\}$



Transition table

δ	a	b	B
$\rightarrow q_0$			(q_1, B, R)
q_1	(q_2, B, R)	(q_5, B, R)	(q_f, B, R)
q_2	(q_2, a, R)	(q_2, b, R)	(q_3, B, L)
q_3	(q_4, B, L)		(q_f, B, R)
q_4	(q_4, a, L)	(q_4, b, L)	(q_1, B, R)
q_5	(q_5, a, R)	(q_5, b, R)	(q_6, B, L)
q_6		(q_4, B, L)	(q_f, B, R)
$* q_f$			

Q5) Obtain a TM to accept the language $L = \{ \omega\omega^R \mid \omega \in \{a,b\}^* \}$

Soln:

$\delta(q_0, B) = (q_1, B, R)$

$\delta(q_1, a) = (q_2, B, R)$

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

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

$\delta(q_2, B) = (q_3, B, L)$

$\delta(q_3, a) = (q_4, B, L)$

$\delta(q_4, a) = (q_4, a, L)$

$\delta(q_4, b) = (q_4, b, L)$

$\delta(q_4, B) = (q_1, B, R)$

$\delta(q_1, b) = (q_5, B, R)$

$\delta(q_5, a) = (q_5, a, R)$

$\delta(q_5, b) = (q_5, b, R)$

$\delta(q_5, B) = (q_6, B, L)$

$\delta(q_6, b) = (q_4, B, L)$

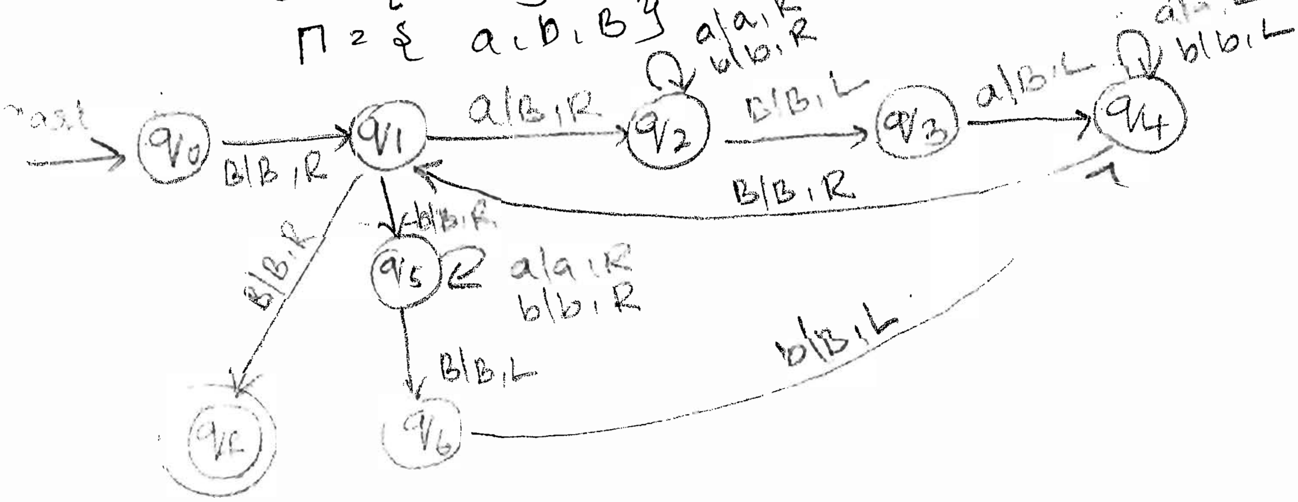
$\delta(q_1, B) = (q_f, B, R)$

$\therefore$ The resulting TM, $M = (Q, \Sigma, \Gamma, \delta, q_0, B, F)$

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

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

$\Gamma = \{a, b, B\}$

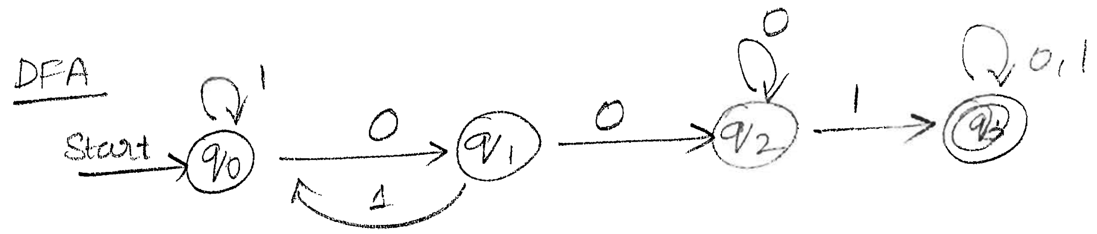


Transition table

δ	a	b	B
→ q_0			(q_1, B, R)
q_1	(q_2, B, R)	(q_5, B, R)	(q_8, B, R)
q_2	(q_2, a, R)	(q_2, b, R)	(q_3, B, L)
q_3	(q_4, B, L)		
q_4	(q_4, a, L)	(q_4, b, L)	(q_1, B, R)
q_5	(q_5, a, R)	(q_5, b, R)	(q_6, B, L)
q_6		(q_4, B, L)	
* q_f			

86) Obtain a TM to accept the language $L = \{w \mid w \in \{0,1\}^*\}$ containing the substring '001'.

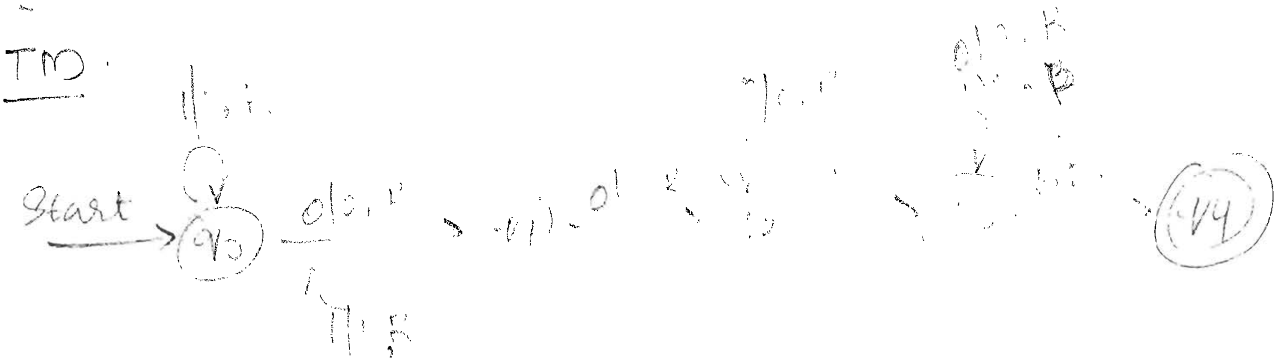
DFA



transition diagram

	0	1
→ q_0	q_1	q_0
q_1	q_2	q_0
q_2	q_2	q_3
* q_3	q_3	q_3

TM.



transition diagram

δ	0	1	B
→ q_0	$(q_1, 0, R)$	$(q_0, 1, R)$	
q_1	$(q_2, 0, R)$	$(q_0, 1, R)$	
q_2	$(q_2, 0, R)$	$(q_3, 1, R)$	
q_3	$(q_3, 0, R)$	$(q_3, 1, R)$	(q_4, B, R)
* q_4			