Finite Automata

Automata Theory :- It is the study of abstract computing devices

→ Why we study automata theory ?

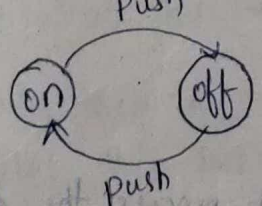
Autometa theory is a model for both hardware and softwar

- Finite Automata is a useful model for many important kinds of hardware and software. Some kinds of automata are
  (a) Software for design and checking the behaviour of digital circuits.
  (b) Lexical Analyzer of compiler that breaks input text into logical units such as keywords, identifiers, punctuations etc
  (c) Software for scan large bodies of text, collection of web pages, to find occurences of words, phrases and other patterns.
  (d) Software for verifying systems of all types that have a finite no. of distinct states.

Note :-

- Finite automata involves states and Transitions. Among states in response to inputs.
- The purpose of state is to remember the relevant portion of a system.

Example :- finite automata for ON/OFF switch

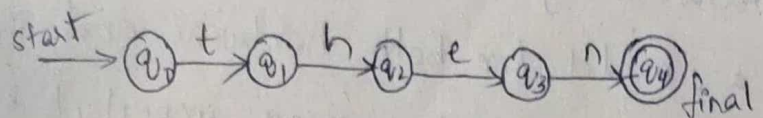


Simple example for F.A

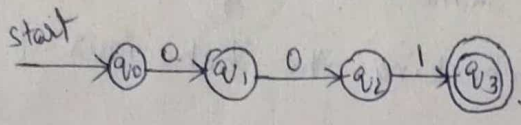
In finite automata, the states are represented by circles. The transitions are enabled by an input. One of the state is designed as start or initial state represented by '→○'.

It is necessary to indicate one or more states as final or accepting states denoted by '◎'.

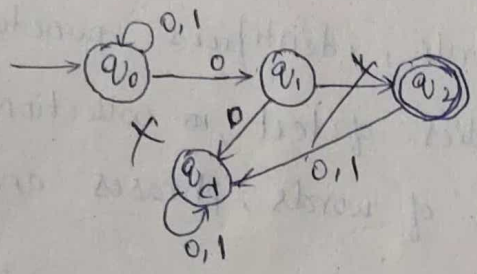
Construct a FA which accepts a word 'then'.

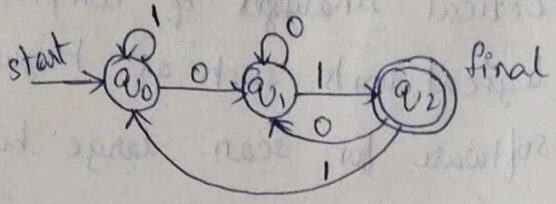


Construct a FA for the string 001

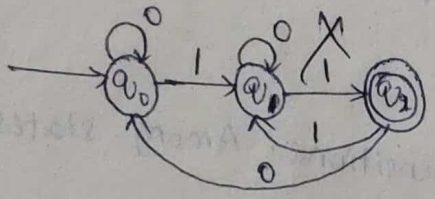


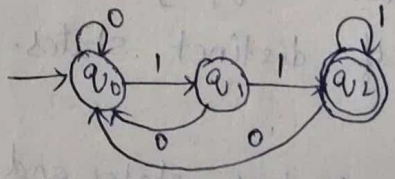
Design DFA which accepts strings ends with 01 over $\Sigma = \{0,1\}$





Design DFA which accepts string ends with 11 over $\Sigma = \{0,1\}$





Structural representation :-

There are two important notations play major role in study of automata and its applications. They are

1. Grammar
2. Regular expression.

Grammar:- Grammar are useful model that process the data recursive structures

Ex:- $E \rightarrow E+E$

$E \rightarrow E*E$

$E \rightarrow id$

Regular Expressions :- These also denotes the structure of data especially text stream. The Unix style regular expression

[A-Z][a-z]*[ ][a-z]*

[A-Z] denotes range of characters from A to Z

[a-z]* denotes any no. of lower case letters from a to z.

(* represents any no. of times)

[ ] denotes space (or) blank

Paranthesis are used to group the contents of expression

Central Concepts of Automata theory :-

1. Alphabet :- It is a finite non-empty collection of symbols denoted by Σ.

Ex:- $\Sigma = \{0,1\}$ → binary alphabet

$\Sigma = \{a, b, \ldots z\}$ → lower case alphabets

2. String :- String is a finite collection of alphabets or a string is a finite sequence of symbols choosen from alphabets

Ex:- 01101 is a string choosen from $\Sigma = \{0,1\}$

aba is a string from $\Sigma = \{a,b\}$

3. Empty string :- It is a string with zero occurence of symbols denoted by ε i.e, $|\varepsilon| = 0$

Length of the string :- No. of symbols in the string gives the length of the string.

Standard notation for length of string w is $|w|$

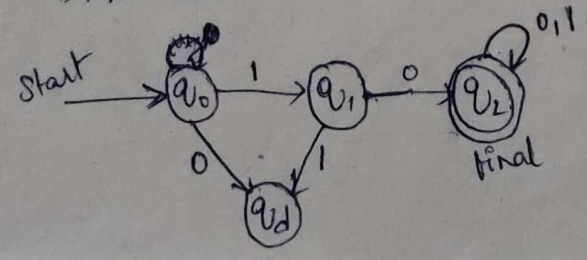
Ex:-

If w = abab then $|w| = 4$

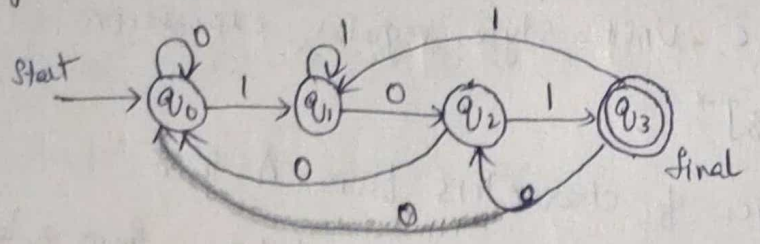
If w = 000101 then $|w| = 6$

If $w = \varepsilon$ then $|w| = 0$

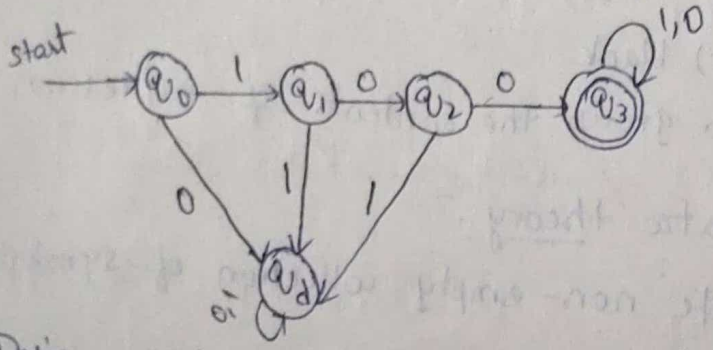
Design DFA which accepts strings that starts with 10 over $\Sigma = \{0,$



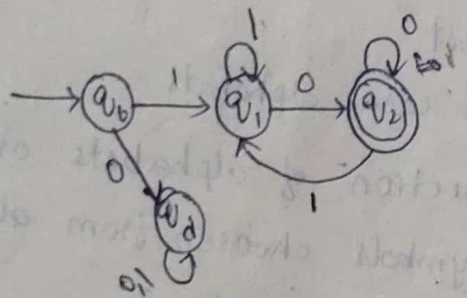
Design DFA which accepts strings ending with 101



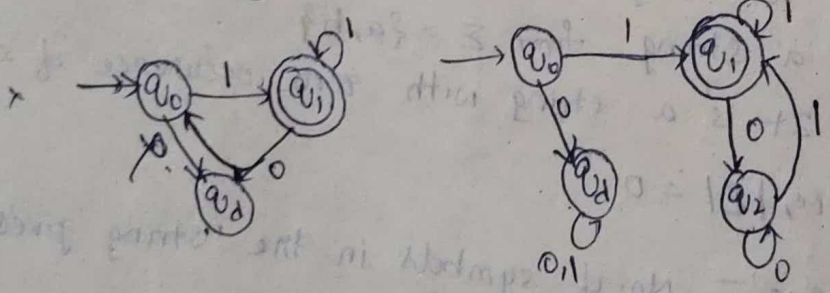
Design DFA which accepts strings starts with 100



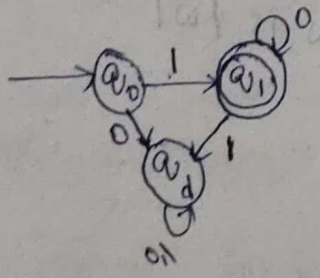
Design DFA string starts with 1 ends with 0



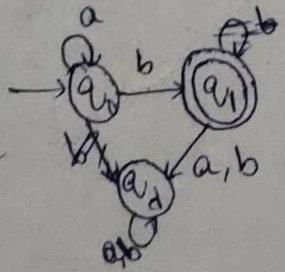
Design DFA string starts with 1 ends with 1



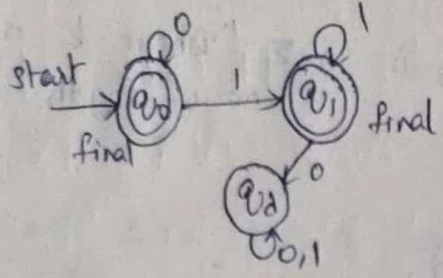
Design DFA which accepts 1 followed by any no of zeros.



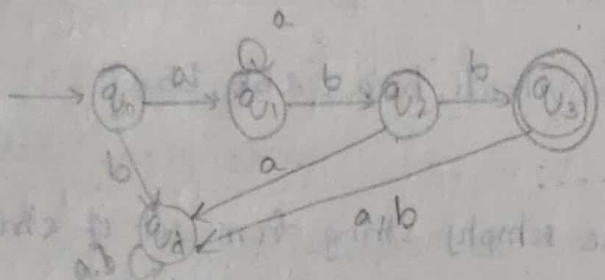
Design DFA which accepts any no. of a's followed by b



Design DFA which accepts strings containing any no. of 0's followed by any no. of 1's

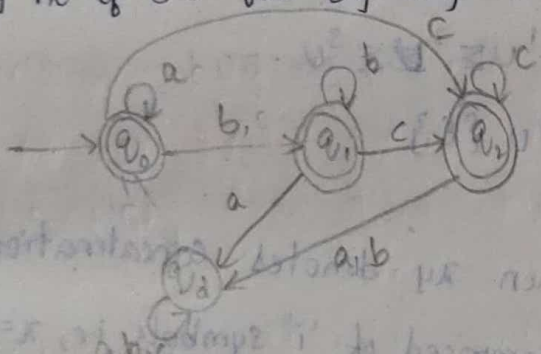


Design DFA which accepts strings atleast one 'a' followed 2 b's over $\Sigma = \{a, b\}$



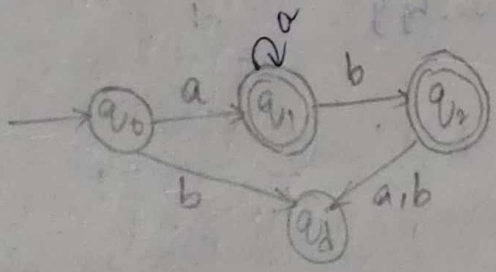
Design DFA which accepts strings contains any no. of a's foll. by any no. of b's foll. by any no. of c's

$L = \{\epsilon, a, b, c, ab, bc, ac\}$



Design DFA which accepts strings atleast one 'a' followed by atmost one b.

$L = \{a, ab, aab, \ldots\}$



Powers of an alphabet :-

If Σ is an alphabet, we define Σ^* to be the set of strings of length k, each of whose symbols is in Σ. where k is any no. of length.

Ex:-

If $\Sigma = \{a, b\}$ then $\Sigma^0 = \{\varepsilon\}$

Here ϵ is only string of length 0.

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

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

$\Sigma^3 = \{aaa, aab, aba, bbb, bab, baa, abb, bba\}$

Note:-

1. $\Sigma^* = \Sigma^0 \cup \Sigma^1 \cup \Sigma^2 \cup \Sigma^3 \cup \ldots$
2. Sometimes we exclude the empty string from set of strings. So, the set of non-empty strings from alphabet Σ
3. Set of non-empty strings from alphabet $\Sigma^+ = \Sigma^* - \varepsilon$

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

$$\Sigma^* = \Sigma^+ \cup \{\varepsilon\}$$

Concatenation of strings :-

Let x, y be two strings then xy denotes concatenation of x and y. If x is a string composed of 'i' symbols i.e, $x = \{a_1, a_2, \ldots a_i\}$ and y is a string composed of j symbols i.e, $y = \{b_1, b_2, \ldots b_j\}$

then $xy = \{a_1, a_2, \ldots a_i, b_1, b_2, \ldots b_j\}$

Ex:- if $x = 0101$ and $y = 11$

then $xy = 010111$

$|xy| = 6$

Note:- For any string w, $\boxed{\varepsilon \cdot w = w \cdot \varepsilon = w}$

Language :- Language is a collection of appropriate strings or Language is a finite ~~sentence~~ set of ~~symbols~~ strings formed from alphabets.

If Σ is an alphabet, then $L \subseteq \Sigma^*$

→ Language of all strings consisting of 0's and 1's with n no. of 0's followed by n no. of 1's where $n \geq 0$

→ Language of all strings containing n no. of 0's followed by n no of 1's where $n \geq 1$

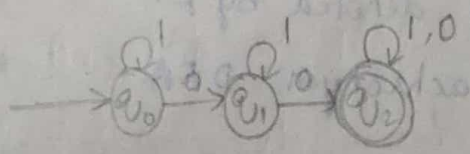
→ Primary language is a set of binary numbers whose values are finite

$L = \{01, 11, 101, 111, 1011, \ldots\}$

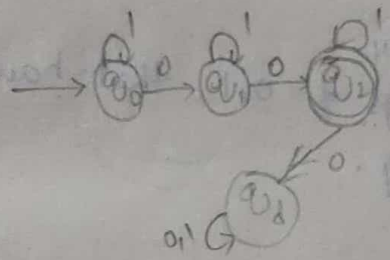
→ $\emptyset$ is an empty language i.e, it has no strings. $\therefore \emptyset = \{\}$

→ $\emptyset \neq \{\varepsilon\}$ because $\emptyset$ has no strings. because ε has one string called empty string.

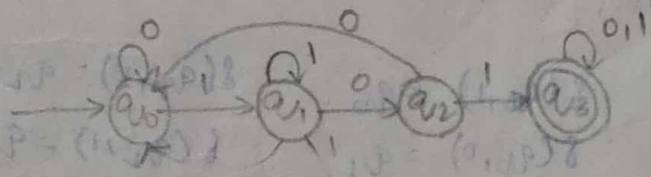
Design DFA which accepts all strings containing two 0's



Design DFA which accepts all strings containing exactly 2 0's.



Design DFA which accepts 101 as substring



Describe language over input alphabet $\Sigma = \{a, b\}$

1. $L = \{a, ab, ab^2, ab^3, \ldots\}$

$L = \{w | w$ is a string a followed by any no. of b's $\}$

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

$L = \{w | w$ is all non empty strings with equal no. of a's and b's $\}$

3. $L = \{a^m b^n | m, n \geq 1\}$

$L = \{w | w$ is atleast one a followed by atleast one b $\}$

4. $L = \{\epsilon, 0, 00, 000, \ldots\}$ over $\Sigma = \{0, 1\}$

$L = \{w \mid w$ is a string with any no. of zeros and no 1's

Deterministic Finite Automaton:-

It consists of

(i) finite set of states defined by Q.

(ii) finite set of input symbols determined by Σ

(iii) the transition function that takes arguments as state and input symbol returns state denoted by δ.

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

(iv) A start state q_0, one of the states in Q

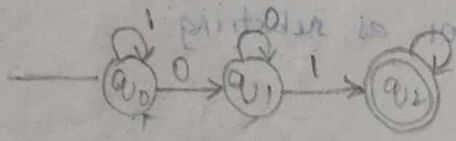
(v) set of final or accepting states denoted by F.

DFAs will be referred by its acronym DFA

five tuple notation of DFA is

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

Construct DFA that accepts all, and only strings of 0,1 that have the sequence of 01 somewhere in the string.



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

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

$q_0 = q_0$

$F = \{q_2\}$

$\delta(q_0, 1) = q_0$

$\delta(q_0, 0) = q_1$

$\delta(q_1, 1) = q_2$

$\delta(q_1, 0) = q_1$

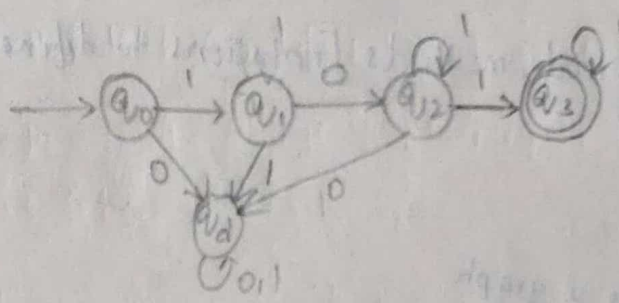
$\delta(q_2, 0) = q_2$

$\delta(q_2, 1) = q_2$

transition table

δ	0	1
$\to q_0$	q_1	q_0
q_1	q_1	q_2
$*q_2$	q_2	q_2

Construct DFA begins with 101



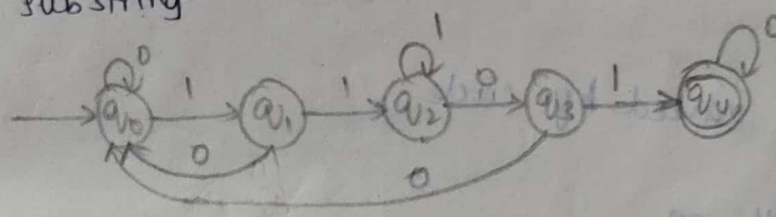
$\Sigma = \{0, 1\}$

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

$F = \{q_3\}$

δ	0	1
→ q_0	q_d	q_1
q_1	q_2	q_d
q_2	q_d	q_3
* q_3	q_3	q_3

✓ Construct DFA which accepts set of all strings contains 1101 as substring



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

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

$F = \{q_4\}$

δ	0	1
→ q_0	q_0	q_1
q_1	q_0	q_2
q_2	q_3	q_2
q_3	q_0	q_4
* q_4	q_4	q_4

Simple Notations for DFA :-

There are two preferred methods / notations to define automata

1. Transition diagram
2. Transition table

1. Transition diagram :- It is a graph contains states and input symbols.

Transition diagram for DFA, $A = (Q, \Sigma, \delta, q_0, F)$ is a graph defined as follows.

(i) for each state in Q there is a node denoted by circle.

(ii) foreach state q_i in Q, foreach input symbol, a in Σ, let $\delta(q_i, a) = p$ then the transition diagram has an arc from state q_i to state p labelled a.

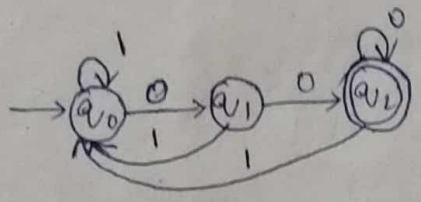
(iii) There is an arrow to the initial state q_0 labelled *

(iv) The nodes corresponding to accepting states (F) are marked by double circle.

(v) States not in F marked by circle.

Ex :-

DFA ends with 00

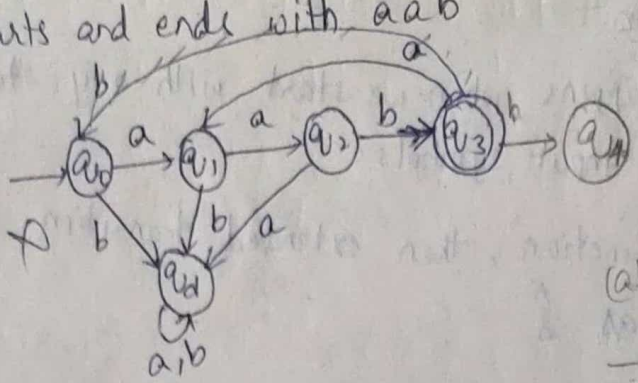


2. Transition table :- A tabular representation of a function δ that takes two arguments - state and symbol and returns state.

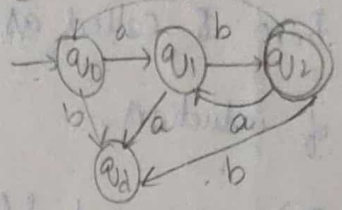
The rows of table corresponds to state and columns to input symbols

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

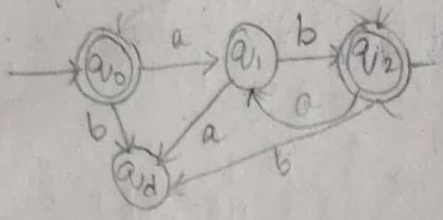
starts and ends with aab

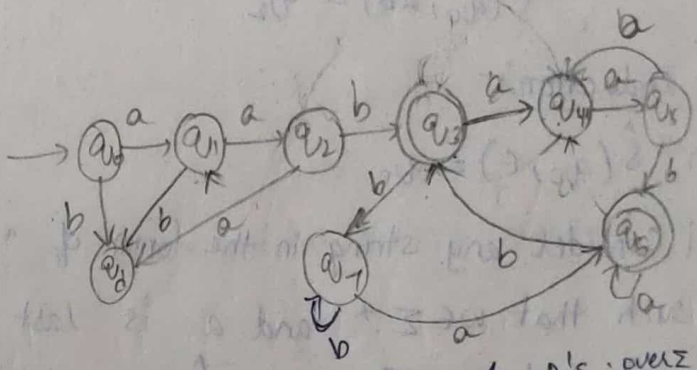


$(ab)^n$, $n \geq 1$

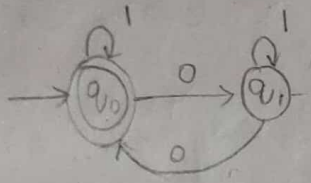


$(ab)^n$, $n \geq 0$



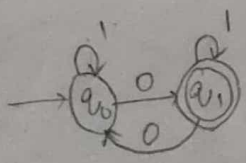


Design DFA which accepts strings containing even no. of 0's. over $\Sigma = \{0,$



	0	1
* → q_0	q_1	q_0
q_1	q_0	q_1

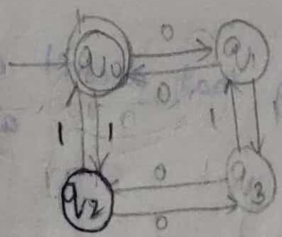
Design DFA which accepts strings containing odd no. of zeros.



	0	1
* q_0	q_1	q_0
→ q_1	q_0	q_1

Design DFA which accepts even no. of 0's and even no. of 1's

ε, 11, 00, 1100, 0



(i) If q_0 is final state, even no. of 0's and even no. of 1's

(ii) If q_1 is final state, odd no. of 0's and even no. of 1's

(iii) If q_3 is final state, odd no. of 0's and odd no. of 1's

(iv) If q_2 is final state, even no. of 0's and odd no. of 1's

Extended Transition function:-

It describes what happens when we start with any state and follow the sequence of input symbols.

If δ is our transition function, then extended transition function constructed from δ called as $\hat{\delta}$

Proof:- Basis of induction

(i) if $\delta(q_0, a) = q_1$ and $\delta(q_1, b) = q_2$ then

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

Induction:-

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

and consider any string in the form of "wa" where w is a string such that $w \in \Sigma^*$ and a is last alphabet of string "wa". such that $a \in \Sigma$ then $\hat{\delta}(q_0, wa) = \delta(\hat{\delta}(q_0, w), a)$

To prove $\hat{\delta}(q_0, ab) = q_2$

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

$= \delta(\hat{\delta}(q_0, a), b) \longrightarrow (1)$

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

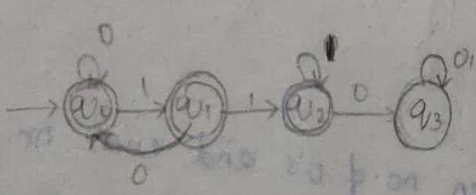
$= \hat{\delta}(\hat{\delta}(q_0, \epsilon), a)$

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

$\hat{\delta}(q_0, ab) = \delta(q_1, b)$

$= q_2$

Design DFA which accepts string that doesn't contain 110, $\Sigma = \{0, 1\}$



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

also checks whether the string 01111 is accepted 1101

$\hat{\delta}(q_0, 01111)$

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

$= \delta(\delta(\delta(\delta(\hat{\delta}(q_0, \epsilon), 0), 1), 1), 1), 1) = q_2$

method 2:

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

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

$= \delta(q_0, 0)$

$= q_0$

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

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

$\hat{\delta}(q_0, 011) = \delta(\hat{\delta}(q_1, \epsilon), 1)$

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

$\hat{\delta}(q_0, 0111) = \delta(\hat{\delta}(q_2, \epsilon), 1)$

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

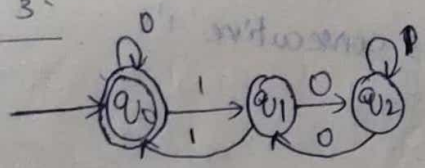
$\hat{\delta}(q_0, 01111) = q_2$

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

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

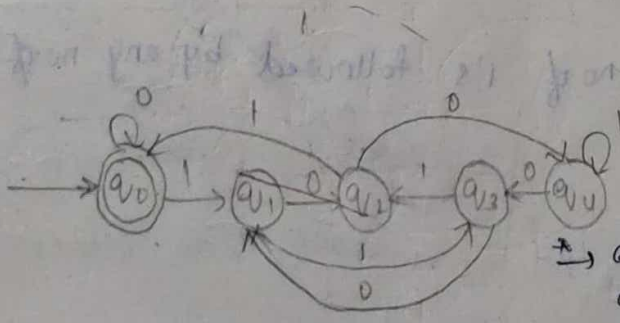
$= \delta(q_0, 0) = q_0$

Design DFA which accepts strings interpreted as binary integer is multiple of 3:



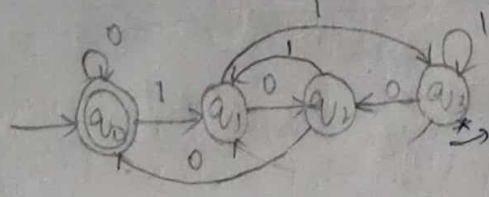
	0	1
→ * q_0	q_0	q_1
q_1	q_2	q_0
q_2	q_1	q_2

Design DFA which accepts strings multiplicple of 5



	0	1
→ * q_0	q_0	q_1
q_1	q_2	q_3
q_2	q_4	q_0
q_3	q_1	q_2
q_4	q_3	q_4

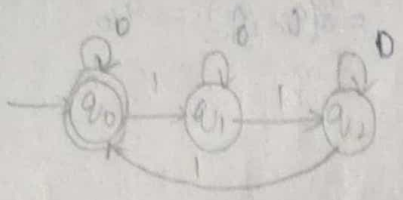
Design DFA multiple of 4



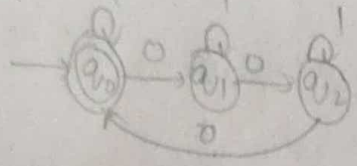
	0	1
→ * q_0	q_0	q_1
q_1	q_2	q_3
q_2	q_0	q_1
q_3	q_1	q_3

Design DFA which accepts all strings contains 0's and 1's which no of 1's is divisible by 3

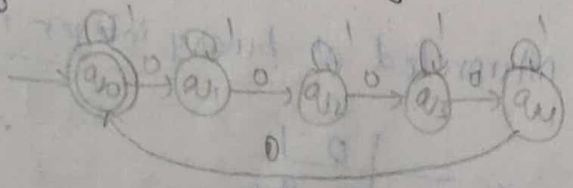
ε, 111, 111111, ...



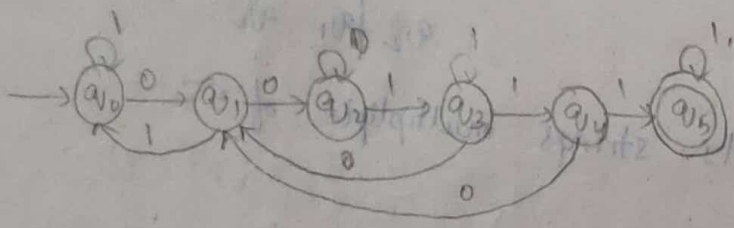
no of 0's is divisible by 5

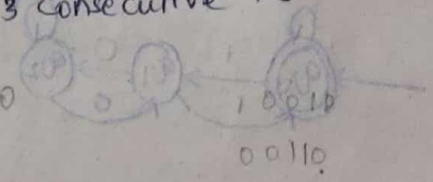


Design DFA which accepts all strings contains 0's and 1's which no of 0's is divisible by 5

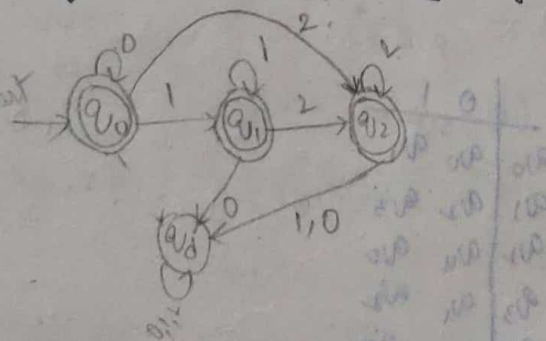


Design DFA 2 consecutive 0's followed 3 consecutive 1's



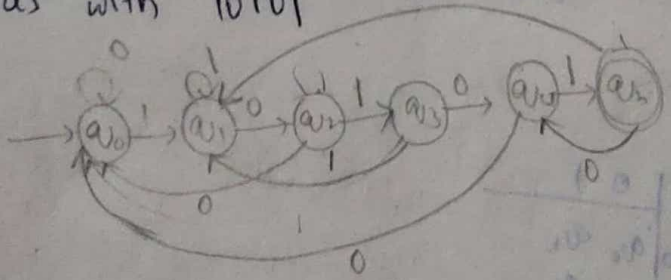


Any no of 0's followed by any no of 1's followed by any no of 2's.



	0	1	2
q_0	q_0	q_1	q_2
q_1	q_3	q_1	q_2
q_2	q_3	q_3	q_2

ends with 10101



1011

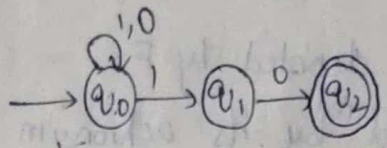
10100101011

	0	1
101010		
101011		

Non- Deterministic finite Automata :-

Concept of NFA is exactly the reverse of DFA. The FA is called NFA when ∃ many paths for specific input from current state to next state.

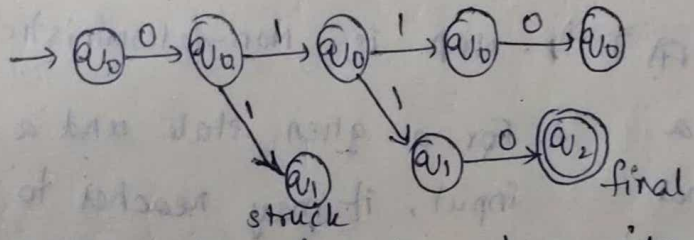
Design NFA which accepts alle strings ends with 10



Proliferation of states :-

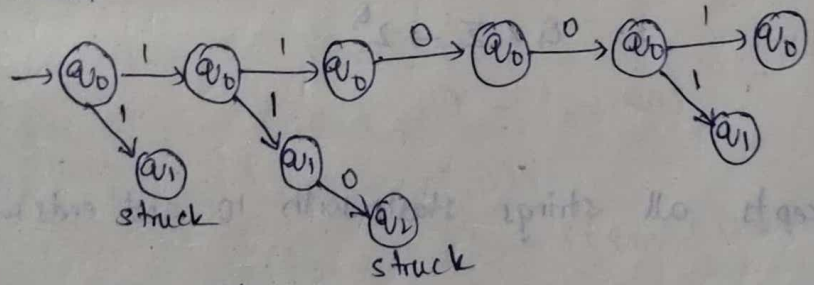
states of NFA is during a process of input sequence

0110

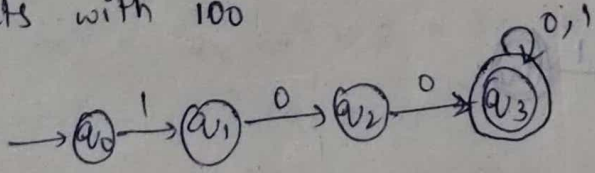


Check whether the following string is NFA or not

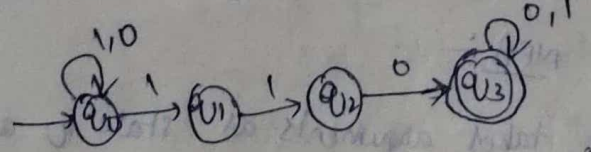
i) 11001



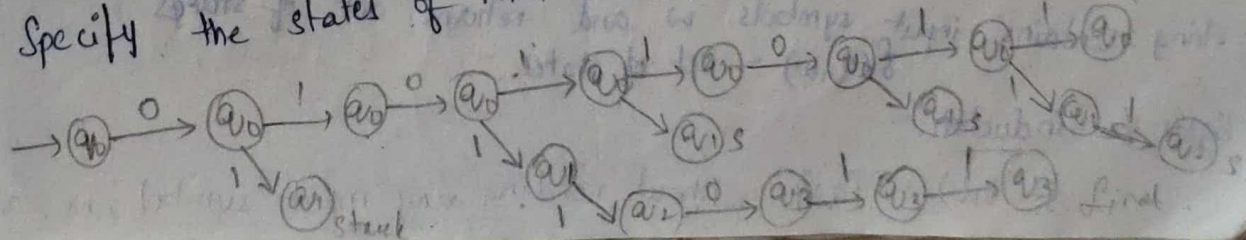
ii) starts with 100



110 as sub string



Specify the states of NFA during the process of 01011011



NFA consists of

(i) finite set of states denoted by Q.

(ii) finite set of input symbols by Σ

(iii) transition function δ takes arguments as a state and input symbol then returns set of states i.e, $Q \times \Sigma \to 2^Q$

(iv) start state q_0, one of the states in Q

(v) set of final or accepting states denoted by F.

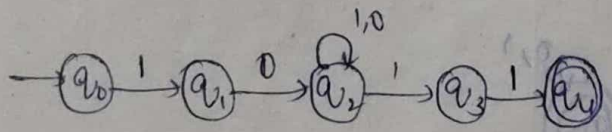
(vi) Non-deterministic FA is differed referred by its acronym NFA

(vii) Five-tuple notation for NFA is $A = (Q, \Sigma, \delta, q_0, F)$

Difference b/w NFA and DFA :-

DFA	NFA
1. DFA is deterministic FA	1. NFA is Non-deterministic FA
2. For a given state on a given input, it reaches to deterministic and unique state.	2. For a given state and a given input, it may reaches to 0 (or) more states.
3. Transition function $Q \times \Sigma \to Q$	3. $Q \times \Sigma \to 2^Q$

Design NFA which accepts all strings starts with 10 and ends with 11



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

Starts with 10, ends with 01

Extended transition function for NFA :-

It is denoted by $\hat{\delta}$ which takes arguments as state q and a string contains input symbols w and returns set of states

$\hat{\delta}(q, w)$ = set of states

Basis of induction :-

$\hat{\delta}(q, t) = \{q\}$ without reading any input symbol, we are

only in state begin with q.

Induction :- Suppose w is of the form $w = xa$ where a is final symbol of w where $a \in \Sigma$ and $x \in \Sigma^*$

if $\hat{\delta}(q_i, x) = (P_1, P_2, \ldots P_k)$ and

$\bigcup_{i=1}^{k} \delta(P_i, a) = \{r_1, r_2, \ldots r_m\}$ then

$\hat{\delta}(q_i, w) = \{r_1, r_2, \ldots r_m\}$

LHS $= \hat{\delta}(q_i, w) = \hat{\delta}(q_i, xa)$

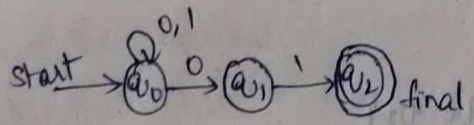
$= \delta(\hat{\delta}(q_i, x), a)$

$= \delta((P_1, P_2, \ldots P_k), a)$

$= \delta(P_1, a) \cup \delta(P_2, a) \cup \ldots \delta(P_k, a)$

$= \bigcup_{i=1}^{k} \delta(P_i, a) = \{r_1, r_2, \ldots r_m\}$

let us use $\hat{\delta}$ to describe the processing of string 00101 for a given NFA



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

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

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

$\hat{\delta}(q_0, 00) = \delta(\hat{\delta}(q_0, 0), 0)$

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

$\hat{\delta}(q_0, 00) = \{q_0, q_1\} \cup \emptyset = \{q_0, q_1\}$

$\hat{\delta}(q_0, 001) = \delta(\hat{\delta}(q_0, 00), 1)$

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

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

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

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

$\hat{\delta}(q_0, 0010) = \delta(\hat{\delta}(q_0, 001), 0)$

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

$= \delta(q_0, 0) \cup \delta(q_2, 0)$

$= \{q_0, q_1\}$

$\hat{\delta}(00101) = \delta(\hat{\delta}(q_0, 0010), 1)$

$= \{q_0, q_2\}$

We obtained final state in the result,

$\therefore$ The string is acceptable.

Specify the states of NFA, 101000

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

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

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

$\hat{\delta}(q_0, 10) = \delta(\hat{\delta}(q_0, 1), 0)$

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

$\hat{\delta}(q_0, 101) = \delta(\hat{\delta}(q_0, 10), 1)$

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

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

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

$\hat{\delta}(q_0, 1010) = \delta(\hat{\delta}(101), 0)$

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

$= \delta(q_0, 0) \cup \delta(q_2, 0)$

$= \{q_0, q_1\} \cup \emptyset = \{q_0, q_1\}$

$\hat{\delta}(q_0, 10100) = \delta(\hat{\delta}(1010), 0)$

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

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

$= \{q_0, q_1\}$

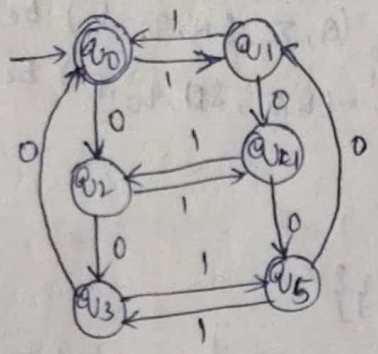
$\hat{\delta}(q_0, 101000) = \delta(\hat{\delta}(10100), 0)$

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

$= \{q_0, q_1\}$

$\therefore$ Since final state is not obtained in result, the string is not acceptable.

Construct DFA which accepts all strings such that no of 1's is even and no of 0's is multiple of 3



- Language of NFA :-

The language of NFA, $A = (Q, \Sigma, \delta, q_0, F)$ is denoted by

$L(A) = \{w \mid \hat{\delta}(q_0, w) \cap F \neq \emptyset\}$

Conversion of NFA to DFA :-

Method of converting NFA to DFA

Let $M = (Q, \Sigma, \delta, q_0, F)$ be NFA which accepts language $L(M)$, there should be equivalent DFA denoted by $M' = (Q', \Sigma', \delta', q_0', F')$ such that $L(M) = L(M')$

step 1:- start state of NFA will be start state of DFA. Hence add q_0 of NFA to Q'

step 2:- for each state $\{q_0, q_1, q_2, \ldots q_i\}$ in Q', the transition for each input symbol in Σ can be obtained as

(i) $\delta'(\{q_0, q_1, \ldots q_i\}, a) = \delta'(q_0, a) \cup \delta'(q_1, a) \cup \ldots \delta'(q_i, a)$

$= \{q_0, q_1, \ldots q_k\}$ (some state)

(ii) Add $\{q_0, q_1, \ldots q_k\}$ to Q' if it is not already in Q'.

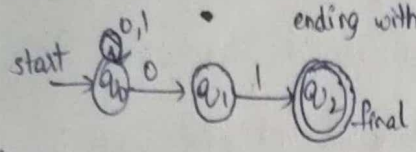
step 3:- Find transitions for every input symbol from Σ for $\{q_0, q_1, \ldots q_k\}$ if we get some states $\{q_0, q_1, \ldots q_n\}$ which is not in Q' of DFA then add $\{q_0, q_1, \ldots q_n\}$ to Q'

step 4:- If there is no new state generated, then rest of the process of finding all transitions can be stopped.

step 5:- For the state $\{q_0, q_1, \ldots q_n\} \in Q'$ of DFA is any one state q_i is the final state of NFA then $\{q_0, q_1, \ldots q_n\}$ becomes final

State of DFA.

1] Convert the following NFA to DFA



ending with 01

let $M = (Q, \Sigma, \delta N, q_0, F)$ be NFA

NFA, $M' = (Q', \Sigma', \delta D, q_0', F')$ be DFA

step1 :-

start state of NFA is q_0

Add q_0 to Q' of DFA. i.e, $Q' = \{\{q_0\}\}$

step2 :- Find transitions for states in Q' for each i/p symbol Σ

$\delta_D(\{q_0\}, 0) = \{q_0, q_1\}$

$\delta_D(\{q_0\}, 1) = \{q_0\}$

Here a new state $\{q_0, q_1\}$ is generated then add it Q' of DFA

i.e, $Q' = \{\{q_0\}, \{q_0, q_1\}\}$

step3 :- Find transition for state $\{q_0, q_1\}$ for each input symbol in Σ

$\delta_D(\{q_0, q_1\}, 0) = \delta_D(q_0, 0) \cup \delta_D(q_1, 0)$

$= \{q_0, q_1\} \cup \emptyset = \{q_0, q_1\}$

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

A new state $\{q_0, q_2\}$ is generated then add it to Q' of DFA

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

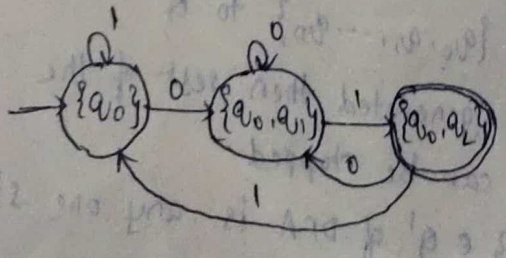
$\delta_D(\{q_0, q_2\}, 0) = \{q_0, q_1\}$

$\delta_D(\{q_0, q_2\}, 1) = \{q_0\}$

There is no new state, so stop the processes.

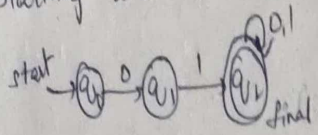
The state $\{q_0, q_2\}$ of Q' contains final state (q_2) of NFA

$\therefore \{q_0, q_2\}$ is final state of DFA



	0	1
→ $\{q_0\}$	$\{q_0, q_1\}$	$\{q_0\}$
$\{q_0, q_1\}$	$\{q_0, q_1\}$	$\{q_0, q_2\}$
* $\{q_0, q_2\}$	$\{q_0, q_1\}$	$\{q_0\}$

2] Starting with 01



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

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

step 1:- start state q_0

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

step 2:-

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

$\delta D(\{q_1\}, 1) = \emptyset$

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

step 3:-

$\delta D(\{q_1\}, 0) = \emptyset$

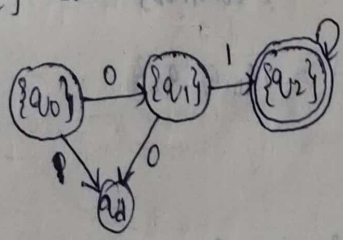
$\delta D(\{q_1\}, 1) = \{q_2\}$

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

$\delta N(\{q_2\}, 0) = \{q_2\}$

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

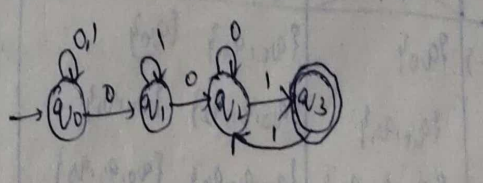
$\{q_2\}$ contains final state (q_2) of NFA.



	0	1
→ {q0}	{q1}	∅
{q1}	∅	{q2}
* {q2}	{q2}	{q2}

3] Convert the following NFA to DFA.

	0	1
→ q_0	$\{q_0, q_1\}$	q_0
q_1	q_2	q_1
q_2	q_2	q_3
* q_3	∅	q_2



start state q_0

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

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

$\delta D(\{q_0\}, 1) = \{q_0\}$

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

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

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

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

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

$= \{q_0\} \cup \{q_1\}$

$= \{q_0, q_1\}$

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

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

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

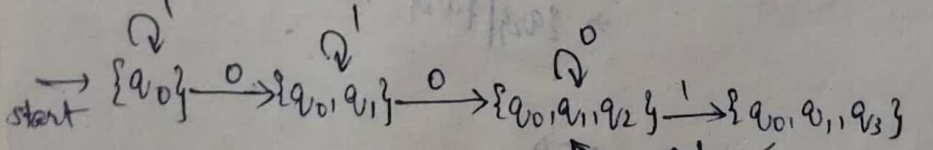
$\delta D(\{q_0, q_1, q_2\}, 1) = \delta D(\{q_0, q_1\}, 1) \cup \delta D(\{q_2\}, 1)$

$= \{q_0, q_1\} \cup \{q_3\} = \{q_0, q_1, q_3\}$

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

$\delta D(\{q_0, q_1, q_3\}, 0) = \{q_0, q_1, q_2\} \cup \emptyset = \{q_0, q_1, q_2\}$

$\delta D(\{q_0, q_1, q_3\}, 1) = \{q_0, q_1\} \cup \{q_2\} = \{q_0, q_1, q_2\}$



δD	0	1	final
$\rightarrow \{q_0\}$	$\{q_0, q_1\}$	$\{q_0\}$	
$\{q_0, q_1\}$	$\{q_0, q_1, q_2\}$	$\{q_0, q_1\}$	
$\{q_0, q_1, q_2\}$	$\{q_0, q_1, q_2\}$	$\{q_0, q_1, q_3\}$	
$* \{q_0, q_1, q_3\}$	$\{q_0, q_1, q_2\}$	$\{q_0, q_1, q_2\}$	

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

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

$q_0' = \{q_0\}$

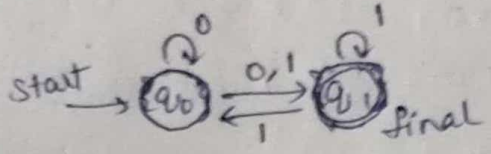
4] Let $M = (\{q_0, q_1\}, \{0,1\}, \delta, q_0, \{q_1\})$ be an NFA where

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

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

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

$\delta(q_1, 1) = \{q_0, q_1\}$ Convert it into DFA.



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

$\delta_D(\{q_0\}, 0) = \{q_0, q_1\}$

$\delta_D(\{q_0\}, 1) = \{q_1\}$

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

$\delta_D(\{q_1\}, 1) = \{q_0, q_1\}$

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

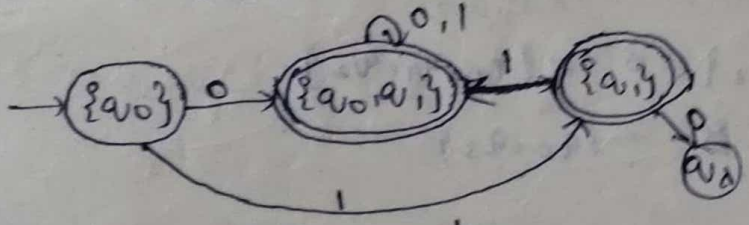
$\delta_D(\{q_0, q_1\}, 0) = \delta_D(\{q_0\}, 0) \cup \delta(\{q_1, 0\})$

$= \{q_0, q_1\} \cup \emptyset = \{q_0, q_1\}$

$\delta_D(\{q_0, q_1\}, 1) = \delta_D(\{q_0\}, 1) \cup \delta_D(\{q_1\}, 1)$

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

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

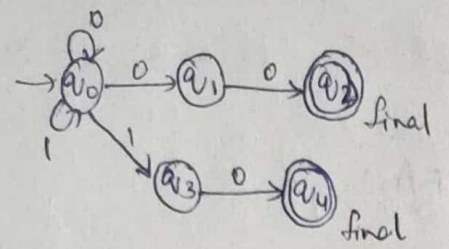


	0	1
→ $\{q_0\}$	$\{q_0, q_1\}$	$\{q_1\}$
* $\{q_0, q_1\}$	$\{q_0, q_1\}$	$\{q_0, q_1\}$
* $\{q_1\}$	$\emptyset$	$\{q_0, q_1\}$

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

$F' = \{\{q_0, q_1\},$

5] Convert to DFA for following NFA



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

$\delta_D(\{q_0\}, 0) = \{q_0, q_1\}$

$\delta_D(\{q_0\}, 1) = \{q_0, q_3\}$

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

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

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

$\delta_D(\{q_0, q_1\}, 1) = \{q_0, q_3\} \cup \emptyset = \{q_0, q_3\}$

$\delta_D(\{q_0, q_3\}, 0) = \{q_0, q_1\} \cup \{q_4\} = \{q_0, q_1, q_4\}$

$\delta_D(\{q_0, q_3\}, 1) = \{q_0, q_3\} \cup \emptyset = \{q_0, q_3\}$

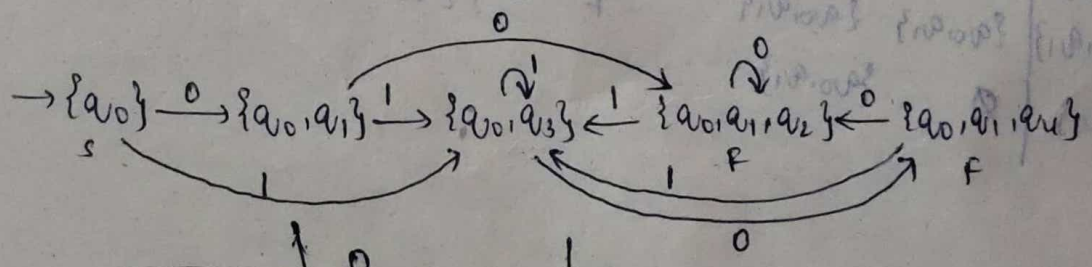
$Q' = \{\{q_0\}, \{q_0, q_1\}, \{q_0, q_3\}, \{q_0, q_1, q_2\}, \{q_0, q_1, q_4\}\}$

$\delta_D(\{q_0, q_1, q_2\}, 0) = \{q_0, q_1, q_2\} \cup \emptyset = \{q_0, q_1, q_2\}$

$\delta_D(\{q_0, q_1, q_2\}, 1) = \{q_0, q_3\} \cup \emptyset = \{q_0, q_3\}$

$\delta_D(\{q_0, q_1, q_4\}, 0) = \{q_0, q_1, q_2\}$

$\delta_D(\{q_0, q_1, q_4\}, 1) = \{q_0, q_3\}$



	0	1
$\{q_0\}$	$\{q_0, q_1\}$	$\{q_0, q_3\}$
$\{q_0, q_1\}$	$\{q_0, q_1, q_2\}$	$\{q_0, q_3\}$
$\{q_0, q_3\}$	$\{q_0, q_1, q_4\}$	$\{q_0, q_3\}$
$\{q_0, q_1, q_2\}$	$\{q_0, q_1, q_2\}$	$\{q_0, q_3\}$
$\{q_0, q_1, q_4\}$	$\{q_0, q_1, q_2\}$	$\{q_0, q_3\}$

Convert following NFA to DFA

NFA	0	1
→ p	{p,q}	{p}
q	{r,s}	{t}
* r	{p,r}	{t}
* s	∅	∅
* t	∅	∅

start state is {p}

$Q^1 = \{\{p\}\}$

$\delta_D(\{p\},0) = \{p,q\}$

$\delta_D(\{p\},1) = \{p\}$

$Q^1 = \{\{p\},\{p,q\}\}$

$\delta_D(\{p,q\},0) = \{p,q\} \cup \{r,s\}$

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

$\delta_D(\{p,q\},1) = \{p\} \cup \{t\}$

$= \{p,t\}$

$Q^1 = \{\{p\},\{p,q\},\{p,q,r,s\},\{p,t\}\}$

$\delta_D(\{p,q,r,s\},0) = \{p,q,r,s\} \cup \{p,r\} \cup \emptyset$

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

$\delta_D(\{p,q,r,s\},1) = \{p,t\} \cup \{t\} \cup \emptyset$

$= \{p,t\}$

$\delta_D(\{p,t\},0) = \{p,q\} \cup \emptyset = \{p,q\}$

$\delta_D(\{p,t\},1) = \{p\} \cup \emptyset = \{p\}$

$Q^1 = \{\{p\},\{p,q,r,s\},\{p,t\},\{p,q\}\}$

	0	1
→ p	{p,q}	{p}
{p,q}	{p,q,r,s}	{p,t}
* {p,q,r,s}	{p,q,r,s}	{p,t}
* {p,t}	{p,q}	{p}

*

Convert NFA to DFA and describe the language it accepts

	0	1
→p	{p,q}	{p}
q	{r}	{r}
r	{s}	∅
*s	{s}	{s}

$Q' = \{\{p\}\}$

$\delta_D(\{p\},0) = \{p,q\}$

$\delta_D(\{p\},1) = \{p\}$

$Q' = \{\{p\},\{p,q\}\}$

$\delta_D(\{p,q\},0) = \{p,q,r\}$

$\delta_D(\{p,q\},1) = \{p,r\}$

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

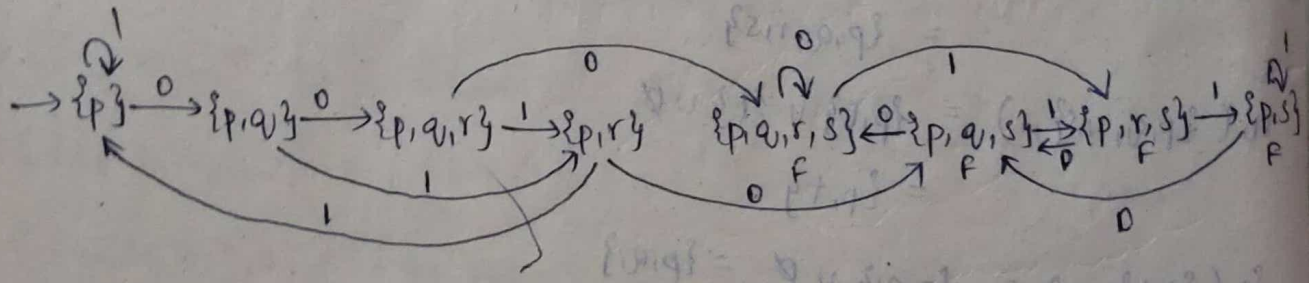
$\delta_D(\{p,q,r,s\},0) = \{p,q,r,s\}$

$\delta_D(\{p,q,r,s\},1) = \{p,r,s\}$

$\delta_D(\{p,s\},0) = \{p,q,s\}$

etc

	0	1
→ {p}	{p,q}	{p}
{p,q}	{p,q,r}	{p,r}
{p,q,r}	{p,q,r,s}	{p,r}
{p,r}	{p,q,s}	{p}
* {p,q,r,s}	{p,q,r,s}	{p,r,s}
* {p,q,s}	{p,q,r,s}	{p,r,s}
* {p,r,s}	{p,q,s}	{p,s}
* {p,s}	{p,q,s}	{p,s}



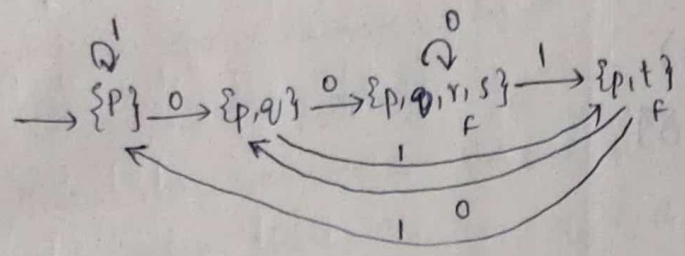
L = contains

accepts all strings ~~start with~~ 000, 010, 0010

Convert NFA to DFA

	0	1
→ p	{p,q}	{p}
q	{r,s}	{t}
r	{p,r}	{t}
* s	∅	∅
* t	∅	∅

	0	1
{p}	{p,q}	{p}
{p,q}	{p,q,r,s}	{p,t}
* {p,q,r,s}	{p,q,r,s}	{p,t}
* {p,t}	{p,q}	{p}



ending with 00, 01

Theorem :-

If $D = (Q_D, \delta_D, \Sigma_D, \{q_0\}, F_D)$ is constructed from $N = (Q_N, \delta_N, \Sigma_N, q_0, F_N)$

then $L(D) = L(N)$

Proof :-

Let us consider a string w of length 'k+1' is of the form $w = xa$ where a is a finite alphabet and x is rest of string w

Basis of induction :-

To prove $L(D) = L(N)$ is true for $|w| = 0$ i.e, $w = \varepsilon$

we know that $\hat{\delta}_D(\{q_0\}, \varepsilon) = \{q_0\}$ $\therefore \hat{\delta}_D(\{q_0\}, \varepsilon) = \hat{\delta}_N(q_0, \varepsilon) = \{q_0\}$

$\hat{\delta}_N(q_0, \varepsilon) = \{q_0\}$ $L(D) = L(N)$ is true for $|w| = 0$

Inductive hypothesis :-

Assume that $L(D) = L(N)$ is true

for $|w| = k$ i.e, $w = x$

$$\hat{\delta}_D(\{q_0\}, x) = \hat{\delta}_N(q_0, x) = \{P_1, P_2, \ldots P_k\} \longrightarrow (1)$$

Inductive step :-

To prove $L(D) = L(N)$ is true for $|w| = k+1$ i.e, $w = xa$

To prove

$$\hat{\delta}_D(\{q_0\}, xa) = \hat{\delta}_N(q_0, xa) \longrightarrow$$

The ~~construction~~ of NFA

basic definition

$$\hat{\delta}_N(q_0, w) = \bigcup_{i=1}^{k} \delta_n((P_i, P_2, \ldots P_k), a) \longrightarrow (2) = \bigcup_{i=1}^{k} \delta_n(P_i, a)$$

Subset construction on the other hand

$$\hat{\delta}_D(\{P_1, P_2, \ldots P_k\}, a) = \bigcup_{i=1}^{k} \hat{\delta}_N(P_i, a) \longrightarrow (3)$$

LHS

$$\hat{\delta}_D(\{q_0\}, xa) = \delta_D(\hat{\delta}_D(\{q_0\}, x), a)$$

$$= \delta_D(P_1, P_2, \ldots P_k), a)$$

$$= \bigcup_{i=1}^{k} \delta_N(P_i, a) \quad [\text{eq } (3)]$$

$$= \hat{\delta}_N(q_0, w)$$

$$= \hat{\delta}_N(q_0, w) \quad [\text{eq } (2)]$$

$$= \hat{\delta}_N(q_0, xa)$$

$$= \text{RHS}$$

Language of NFA

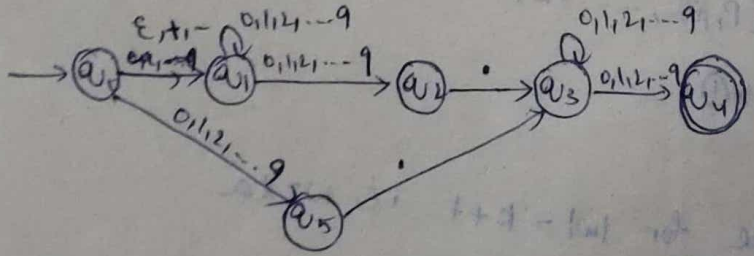
FA with ε-transition:-

This is another extension of FA. The new feature is that allows a transition on ε. i.e, an empty string.

NFA is allowed to make transition without receiving any input symbol.

Ex1:-

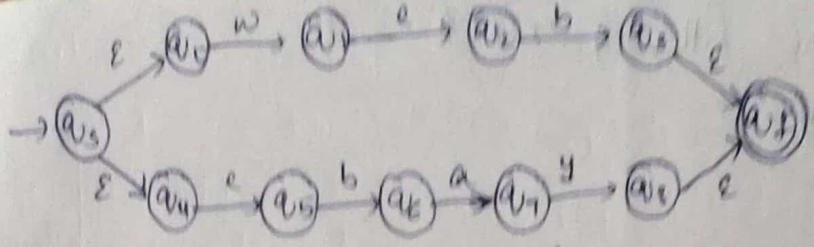
ε-NFA that accepts decimal number consisting of an(i) optional sign + or - (ii) a string of digits 0, 1, ... 9 (iii) a dot operator (iv) other string of digits.



Here $\Sigma = \{\varepsilon, +, -, ., 0, 1, 2, \ldots 9\}$

State q_1 represents a sign

Ex 2:- ε-NFA that recognise set of keywords web, ebay.



Formal definition of ε-NFA:-

ε-NFA can be defined as

1. set of states denoted by Q.
2. finite set of symbols denoted by Σ.
3. Transition function δ defined as $Q \times \Sigma \cup \{\varepsilon\} \rightarrow 2^Q$
4. The start state q_0 i.e, $q_0 \in Q$
5. A set of final states denoted by F
6. Five tuple notation for ε-NFA is $A = (Q, \Sigma, \delta, q_0, F)$

Definition for ε-closure (q) :-

ε-closure of q is set of all states which are reachable for state q on ε-transitions such that

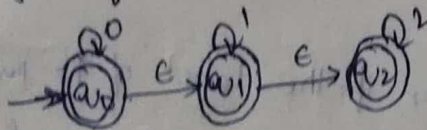
(i) ε-closure (q) q = {q} ∪ ε-transitions from q.

(ii) If ∃ ε-closure (q) = r and ε-closure (r) = s then

∴ ε-closure (q) = {r, s}

Ex:-

ε-NFA for the strings containing any no. of 0's followed by any no. of 1's followed by any no. of 2's.



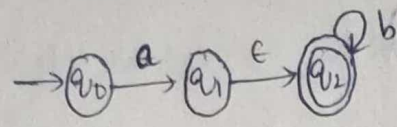
Find ε-closures for all states.

ε-closure (q_0) = {q_0} ∪ {q_1} ∪ {q_2} = {q_0, q_1, q_2}

ε-closure (q_1) = {q_1, q_2}

ε-closure (q_2) = {q_2}

Ex: Find ε-closures for the following ε-NFA

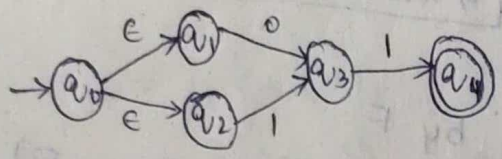


ε-closure $(q_0) = \{q_0\}$

ε-closure $(q_1) = \{q_1\} \cup \{q_2\} = \{q_1, q_2\}$

ε-closure $(q_2) = \{q_2\}$

Ex: Find ε-closures of all states in the following diagram



ε-closure $(q_0) = \{q_0\} \cup \{q_1, q_2\}$

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

ε-closure $(q_1) = \{q_1\}$

ε-closure $(q_2) = \{q_2\}$

ε-closure $(q_3) = \{q_3\}$

ε-closure $(q_4) = \{q_4\}$

Extended transition function for ε-NFA :-

Suppose $E = (Q, \Sigma, \delta, q_0, F)$ is an ε-NFA then extended transition function ε-NFA is denoted by $\hat{\delta}$ is defined as Basis.

* Basis :- $\hat{\delta}(q_0, \varepsilon) = \varepsilon$-closure (q_0) i.e, if the label of the path ε then we can follow only ε-labelled arcs extending from state q_0.

Induction :- Suppose w is a string of the form 'xa', where a is final alphabet of string w and x is rest of the string. then compute $\hat{\delta}(q_0, w)$ as follows.

(i) Let $\{P_1, P_2, \ldots P_k\}$ be $\hat{\delta}(q_0, x)$ i.e, all P_i's are the states that we can reach from the state q_0 following a path 'x'.

(ii) $\bigcup_{i=1}^{n} \delta(P_i, a) = \{r_1, r_2, \ldots r_m\}$ then

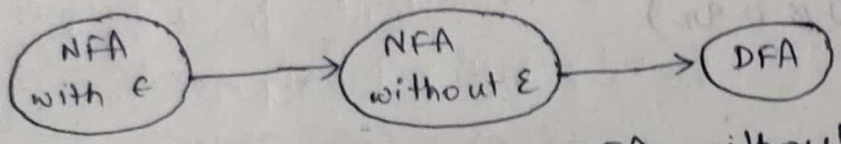
(iii) $\hat{\delta}(q_0, w) = \{r_1, r_2, \ldots r_m\}$

Note :-

1. The symbol for empty string ε is not a member of Σ
2. In ε-NFA, $\hat{\delta}_E(q_0, w) = \varepsilon$-closure $(\delta, \hat{\delta}(q_0, x), a)$

Conversion of ε-NFA to DFA :-

NFA with ε can be converted to NFA without ε which in turn can be converted to DFA. The following diagram represent conversion procedure



Conversion of ε-NFA to NFA without ε :-

(i) Find out all ε-transitions for each state in Q that would be called as ε-closure (q_i) where $q_i \in Q$

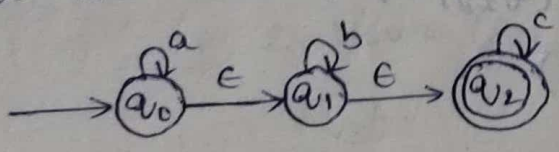
(ii) Then find $\hat{\delta}$ transitions i.e, ε-closure moves on δ.

(iii) Step 2 is repeated for each i/p symbol and each state in given NFA

(iv) Using the resultant state, the transition table for equivalent NFA without ε can be built.

Ex:-

Convert the following ε-NFA to NFA without ε.



Sol

step 1 :-

ε-closure $(q_0) = \{q_0\} \cup \{q_1\} \cup \{q_2\} = \{q_0, q_1, q_2\}$

ε-closure $(q_1) = \{q_1\} \cup \{q_2\} = \{q_1, q_2\}$

ε-closure $(q_2) = \{q_2\}$

step 2 :-

find extended transitions for each state and for each i/p symbol

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

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

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

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

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

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

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

$= \varepsilon\text{-closure}(\emptyset \cup \{q_1\} \cup \emptyset)$

$= \{q_1, q_2\}$

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

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

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

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

$= \{q_2\}$

$\hat{\delta}(q_1, a) = \varepsilon\text{-closure}(\delta(\hat{\delta}(q_1, \varepsilon), a)$

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

$= \varepsilon\text{-closure}(\emptyset \cup \emptyset)$

$= \emptyset$

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

$= \varepsilon\text{-closure}(q_1 \cup \emptyset)$

$= \{q_1, q_2\}$

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

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

$= \{q_2\}$

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

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

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

$= \emptyset$

$\hat{\delta}(q_2, b) = \emptyset$

$\hat{\delta}(q_2, c) = \{q_2\}$

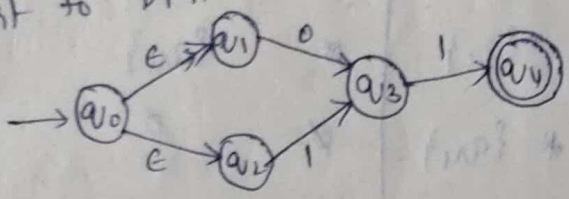
	a	b	c
$\rightarrow *q_0$	$\{q_0, q_1, q_2\}$	$\{q_1, q_2\}$	$\{q_2\}$
$*q_1$	$\emptyset$	$\{q_1, q_2\}$	$\{q_2\}$
$*q_2$	$\emptyset$	$\emptyset$	$\{q_2\}$

Here the states q_0, q_1 & q_2 are final states of NFA without ε because ε-closure (q_0), ε-closure (q_1), ε-closure (q_2) contains the q_2, that is final state of given ε-NFA

DFA

	a	b
→ q_0	$\{q_0, q_1, q_2\}$	$\{q_1, q_2\}$
* $\{q_0, q_1, q_2\}$	$\{q_0, q_1, q_2\}$	$\{q_1, q_2\}$
* q_2	$\emptyset$	$\{q_2\}$
* $\{q_1, q_2\}$	$\emptyset$	$\{q_2\}$

→ Convert the following ε-NFA to NFA without ε and then convert to DFA



step 1:

ε-closure $(q_0) = \{q_0, q_1, q_2\}$

ε-closure $(q_1) = \{q_1\}$

ε-closure $(q_2) = \{q_2\}$

ε-closure $(q_3) = \{q_3\}$

ε-closure $(q_4) = \{q_4\}$

step 2:

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

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

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

$= \varepsilon\text{-closure}(\{q_3\})$

$= \{q_3\}$

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

$= \varepsilon\text{-closure}(q_3)$

$= \{q_3\}$

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

$= \{q_3\}$

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

$= \emptyset$

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

$\hat{\delta}(q_2, 1) = \varepsilon\text{-closure}(\delta(\{q_2\}), 1) = \{q_3\}$

$\hat{\delta}(q_3, 0) = \varepsilon\text{-closure}(\delta(\{q_3\}, 0) = \emptyset$

$\hat{\delta}(q_3, 1) = \{q_4\}$

$\hat{\delta}(q_4, 1) = \varepsilon\text{-closure}(\delta(\{q_4\}, 1)$

$= \emptyset$

$\hat{\delta}(q_4, 0) = \emptyset$

NFA without ε	0	1
$\rightarrow q_0$	$\{q_3\}$	$\{q_3\}$
q_1	$\{q_3\}$	$\emptyset$
q_2	$\emptyset$	$\{q_3\}$
q_3	$\emptyset$	$\{q_4\}$
$* q_4$	$\emptyset$	$\emptyset$

DFA	0	1
$\rightarrow q_0$	$\{q_3\}$	$\{q_3\}$
$\{q_3\}$	$\emptyset$	$\{q_4\}$
$* \{q_4\}$	$\emptyset$	$\emptyset$

	ϵ	a	b	c
$\rightarrow$ p	$\emptyset$	$\{p\}$	$\{q\}$	$\{r\}$
q	$\{p\}$	$\{q\}$	$\{r\}$	$\emptyset$
* r	$\{q\}$	$\{r\}$	$\emptyset$	$\{p\}$

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

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

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

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

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

$= \{p\}$

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

$= \{p, q\}$

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

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

$= \varepsilon\text{-closure}(\{q\} \cup \{p\})$

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

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

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

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

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

$\hat{\delta}(r, a) = \{p, q, r\}$

$\hat{\delta}(r, b) = \{p, q, r\}$

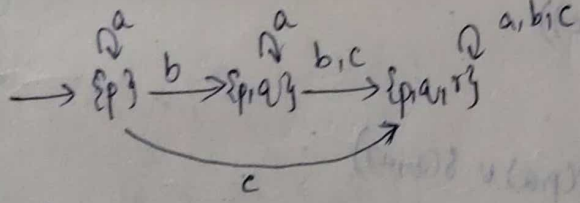
$\hat{\delta}(r, c) = \{p, q, r\}$

NFA without ε

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

DFA

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



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

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

$q_0' = \{p\}$

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

Direct conversion of ε-NFA to DFA :-

1)

	ε	a	b	c
→ p	∅	{p}	{q}	{r}
q	{p}	{q}	{r}	∅
* r	{q}	{r}	∅	{p}

Step 1 :- find ε-closure of each state in ε-NFA

ε-closure(p) = {p}

ε-closure(q) = {p,q}

ε-closure(r) = {p,q,r}

Step 2 :-

Consider start state 'p'.

ε-closure(p) = {p}

Consider $\delta_D(\{p\}, a)$ = ε-closure($\delta(p,a)$)

= ε-closure(p) = {p}

$\delta_D(\{p\}, b)$ = ε-closure($\delta(p,b)$)

= ε-closure(q) = {p,q}

$\delta_D(\{p\}, c)$ = ε-closure($\delta(p,c)$)

= {p,q,r}

$\delta_D(\{p,q\}, a)$ = ε-closure($\delta(p,a) \cup \delta(q,a)$)

= ε-closure(p,q)

= {p,q}

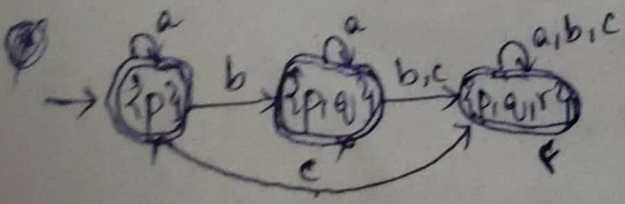
$\delta_D(\{p,q\}, b)$ = {p,q,r}

$\delta_D(\{p,q,r\}, c)$ = {p,q,r}

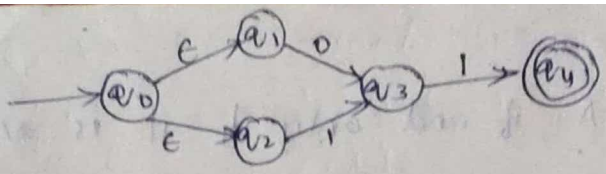
$\delta_D(\{p,q,r\}, a)$ = {p,q}

$\delta_D(\{p,q,r\}, b)$ = {p,q,r}

$\delta_D(\{p,q,r\}, c)$ = {p,q,r}



2]



ε-closure (q_0) = $\{q_0, q_1, q_2\}$

ε-closure(q_1) = $\{q_1\}$

ε-closure (q_2) = $\{q_2\}$

ε-closure (q_3) = $\{q_3\}$

ε-closure (q_4) = $\{q_4\}$

$\{q_0\}$

$\delta_D(\{q_0\}, 0) = \emptyset$

$\delta_D(\{q_0\}, 1) = \emptyset$

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

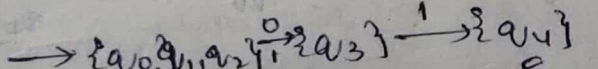
$\delta_D(\{q_0, q_1, q_2\}, 1) = \{q_3\}$

$\delta_D(\{q_3\}, 0) = \emptyset$

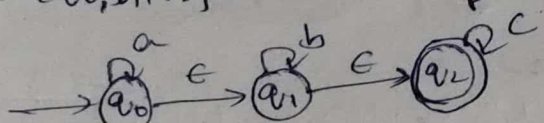
$\delta_D(\{q_3\}, 1) = \{q_4\}$

$\delta_D(\{q_4\}, 0) = \emptyset$

$\delta_D(\{q_4\}, 1) = \emptyset$



3]



ε-closure (q_0) = $\{q_0, q_1, q_2\}$

ε-c(q_1) = $\{q_1, q_2\}$

ε-c(q_2) = $\{q_2\}$

$\delta_D(\{q_0, q_1, q_2\}, a) = \{q_0, q_1, q_2\}$

$\delta_D(\{q_0, q_1, q_2\}, b) = \{q_1, q_2\}$

$\delta_D(\{q_0, q_1, q_2\}, c) = \{q_2\}$

$\delta_D(\{q_1, q_2\}, a) = \{q_0, q_1, q_2\}$

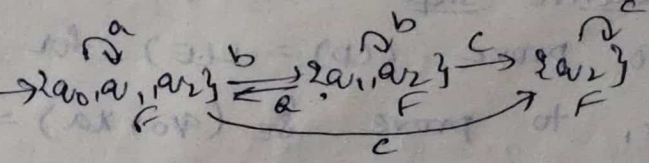
$\delta_D(\{q_1, q_2\}, b) = \{q_1, q_2\}$

$\delta_D(\{q_1, q_2\}, c) = \{q_2\}$

$\delta_D(\{q_2\}, a) = \emptyset$

$\delta_D(\{q_2\}, b) = \emptyset$

$\delta_D(\{q_2\}, c) = \{q_2\}$



Theorem :-

If L is accepted by some ϵ-NFA if and only if it is accepted by some DFA i.e, $L(E) = L(D)$

Proof :-

let $E = (Q_E, \Sigma_E, \delta_E, q_0, F_E)$ be an ϵ-NFA and $D = (Q_D, \Sigma_D, \delta_D, q_D, F_D)$ be a DFA

To prove $L(E) = L(D)$ by assuming that the extended transition functions of E & D are same

$\hat{\delta}_E(q_0, w) = \hat{\delta}_D(q_D, w)$ on the length of the string w by induction.

Basis :- To prove $L(D) = L(E)$ for $|w| = 0$

i.e, $w = \epsilon$

Here w is a string of the form 'xa' where a is last alphabet of w. and x is the rest of the string w.

$\hat{\delta}_E(q_0, \epsilon) = \hat{\delta}_D(q_D, \epsilon)$

ϵ-closure (start state of ϵ-NFA q_0) = start state of DFA q_D

LHS $\hat{\delta}_E(q_0, \epsilon) = \epsilon$-closure (q_0)

RHS $\hat{\delta}_D(q_D, \epsilon) = q_D$

$= \epsilon$-closure (q_0)

LHS = RHS

$L(D) = L(E)$ for $|w| = 0$

Inductive hypothesis :-

Assume $L(D) = L(E)$ for $|w| = k$ i.e, $w = x$

i.e, $\hat{\delta}_E(q_0, x) = \hat{\delta}_D(q_D, x) = \{P_1, P_2, \ldots P_k\} \longrightarrow (1)$

Inductive step :-

To prove $L(D) = L(E)$ for $|w| = k+1$ i.e, $w = xa$

i.e, to prove $\hat{\delta}_E(q_0, xa) = \delta_D(q_D, xa)$

LHS $= \hat{\delta}_E(q_0, xa) = \epsilon$-closure $(\delta_E(\hat{\delta}_E(q_0, x), a))$ [$\because$ eq (1)]

$= \epsilon$-closure $(\delta_E(\{P_1, P_2, P_3 \ldots P_k\}), a)$

$= \epsilon\text{-closure}(r_1, r_2, \ldots r_m)$

$= \{r_1, r_2, \ldots r_m\}$

RHS $= \delta_D(q_D, xa)$

$= \delta_D(\hat{\delta}_D(q_D, x), a)$

$= \delta_D(\{p_1, p_2, \ldots p_k\}, a)$

$= \{r_1, r_2, \ldots r_m\}$

LHS = RHS

$\therefore L(D) = L(E)$ is true for $|w| = k+1$ i.e, $w = xa$

Hence, by M-I $L(D) = L(E)$ is true $\forall$ cases.

Convert the following ϵ-NFA to DFA

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

$\epsilon\text{-closure}(p) = \{p, q, r\}$

$\epsilon\text{-closure}(q) = \{q\}$

$\epsilon\text{-closure}(r) = \{r\}$

$\delta_D(\{p, q, r\}, a) = \epsilon\text{-closure}(\{p\}) = \{p, q, r\}$

$\delta_D(\{p, q, r\}, b) = \{q, r\}$

$\delta_D(\{p, q, r\}, c) = \{p, q, r\}$

$\delta_D(\{q, r\}, a) = \{p, q, r\}$

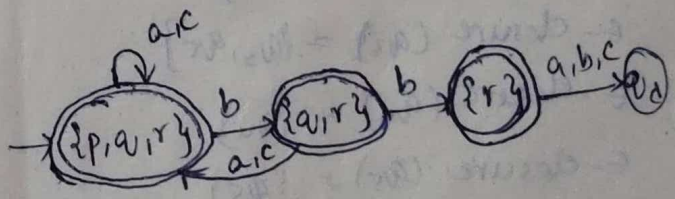
$\delta_D(\{q, r\}, b) = \{r\}$

$\delta_D(\{q, r\}, c) = \{p, q, r\}$

$\delta_D(\{r\}, a) = \emptyset$

$\delta_D(\{r\}, b) = \emptyset$

$\delta_D(\{r\}, c) = \emptyset$



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

1] Convert the following ε-NFA to DFA

	ε	+,-	.	0,1,2,...9
→ q_0	$\{q_1\}$	$\{q_1\}$	∅	∅
q_1	∅	∅	$\{q_2\}$	$\{q_1, q_4\}$
q_2	∅	∅	∅	$\{q_3\}$
q_3	$\{q_5\}$	∅	∅	$\{q_3\}$
q_4	∅	∅	$\{q_3\}$	∅
* q_5	∅	∅	∅	∅

2] For given ε-NFA

	ε	a	b
→ P	$\{r\}$	$\{q\}$	$\{p, r\}$
q	∅	$\{p\}$	∅
* r	$\{p, q\}$	$\{r\}$	$\{p\}$

(a) Compute ε-closure of each state

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

(c) Convert automata to DFA

1] ε-closure $(q_0) = \{q_0, q_1\}$

ε-closure $(q_1) = \{q_1\}$

ε-closure $(q_2) = \{q_2\}$

ε-closure $(q_3) = \{q_3, q_5\}$

ε-closure $(q_4) = \{q_4\}$

ε-closure $(q_5) = \{q_5\}$

Start state $\{q_0, q_1\}$

$\delta_D(\{q_0, q_1\}, +, -) = \{q_1\}$

$\delta_D(\{q_0, q_1\}, .) = \{q_2\}$

$\delta_D(\{q_0, q_1\}, 0,1,2,\ldots 9) = \{q_1, q_4\}$

$\delta_D(\{q_1\}, +, -) = \emptyset$

$\delta_D(\{q_1\}, .) = \{q_2\}$

$\delta_D(\{q_1\}, 0,1,2,\ldots 9) = \{q_1, q_4\}$

$\delta_D(\{q_2\}, +, -) = \emptyset$

$\delta_D(\{q_2\}, \cdot) = \emptyset$

$\delta_D(\{q_2\}, 0,1,2,\ldots 9) = \{q_3, q_5\}$

$\delta_D(\{q_1, q_4\}, +, -) = \emptyset$

$\delta_D(\{q_1, q_4\}, \cdot) = \{q_2, q_3, q_5\}$

$\delta_D(\{q_1, q_4\}, 0,1,2\ldots 9) = \{q_1, q_4\}$

$\delta_D(\{q_3, q_5\}, +, -) = \emptyset$

$\delta_D(\{q_3, q_5\}, \cdot) = \emptyset$

$\delta_D(\{q_3, q_5\}, 0,1,2,\ldots 9) = \{q_3, q_5\}$

$\delta_D(\{q_2, q_3, q_5\}, +, -) = \{q_3, q_5\}\ \emptyset$

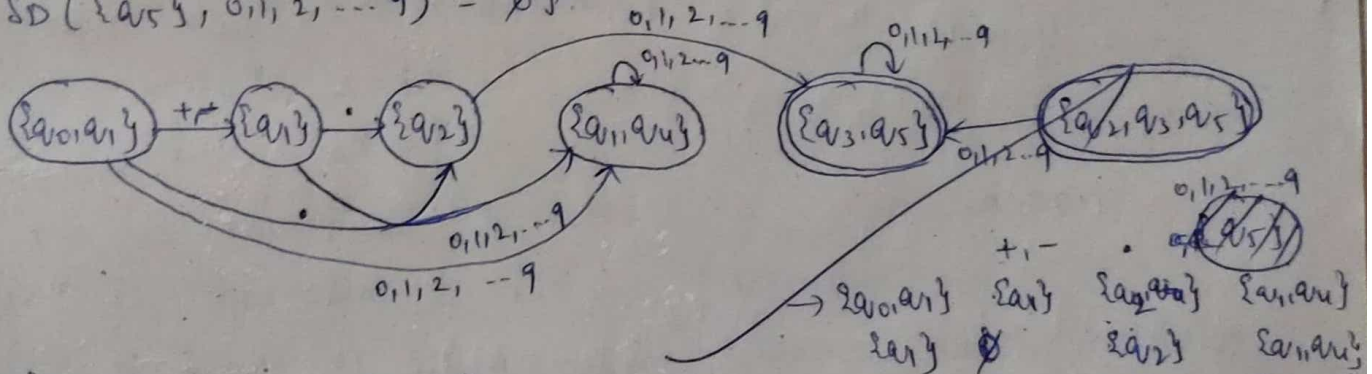
$\delta_D(\{q_2, q_3, q_5\}, \cdot) = \emptyset$

$\delta_D(\{q_2, q_3, q_5\}, 0,1,2,\ldots 9) = \{q_3, q_5\}$

$\delta_D(\{q_5\}, +, -) = \emptyset$

$\delta_D(\{q_5\}, \cdot) = \emptyset$

$\delta_D(\{q_5\}, 0,1,2,\ldots 9) = \emptyset$



	+ , −	.	0,1,2,...9
→ $\{q_0, q_1\}$	$\{q_1\}$	$\{q_2\}$	$\{q_1, q_4\}$
$\{q_1\}$	$\emptyset$	$\{q_2\}$	$\{q_1, q_4\}$
$\{q_2\}$	$\emptyset$	$\emptyset$	$\{q_3, q_5\}$
$\{q_1, q_4\}$	$\emptyset$	$\{q_2, q_3, q_5\}$	$\{q_1, q_4\}$
* $\{q_3, q_5\}$	$\emptyset$	$\emptyset$	$\{q_3, q_5\}$
* $\{q_2, q_3, q_5\}$	$\emptyset$	$\emptyset$	$\{q_3, q_5\}$

(b) $L = \{\epsilon, a, b, ab, ba, aab, abb, bba, baa, aaa, bbb, bab, aba\}$

2] (a) ϵ-closure $(p) = \{p, r\}$

ϵ-closure $(q) = \{q\}$

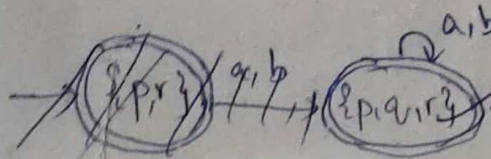
ϵ-Closure $(r) = \{p, q, r\}$

(c) ~~$\delta_D(\{p, r\}, a) = \{p, q, r\}$~~

~~$\delta_D(\{p, r\}, b) = \{p, q, r\}$~~

$\delta_D(\{p, r\}, a) = \{p, q, r\}$

$\delta_D(\{p, q, r\}, b) = \{p, q, r\}$



	a	b
→ ~~$\{p, r\}$~~	~~$\{p, q, r\}$~~	~~$\{p, q, r\}$~~
* $\{p, q, r\}$	$\{p, q, r\}$	$\{p, q, r\}$

Sunday
last 10 min. Regular Expressions & Regular languages

The language accepted by FA are easily described by simple expressions called Regular exp.
The regular exp. is most effective way to represent any values
The language accepted by some regular exp. is called regular language

Building Regular Expressions :-

Let Σ be an alphabet which is used to denote the input set. The regular expression over Σ can be defined as follows

1. $\emptyset$ is regular expression and denotes empty set $\{\}$
2. ϵ_0 is a regular expression and denoted the set $\{\epsilon\}$ and it is null string.
3. foreach 'a' is Σ is regular expression and denotes set $\{a\}$
4. If R and S are regular exp. denoting the languages L_1 and L_2 resp. then R+S is equivalent to $L_1 \cup L_2$ i.e, union

$r = a^*$; $s = b^*$

$r+s = a^*, b^*$

5. r.s is equivalent to $L_1 \cdot L_2$ i.e,. concatenation
6. r^* is equivalent to L_1^* i.e, closure

r^* is known as Kleen closure, closure which indicates occurence of 'r' for infinite no. of times

Formal definition of regular expression :-

Regular exp. is defined to describes any regular language accepted by FA. Regular exp. involves combination of strings of symbol from some alphabets Σ, (,), and the operators +, *, .

where + is used to denote union operation

* is used to denote closure operation or * closure

. is used to denote concatenation operation.

Operations on languages :-

There are three basic operations on languages. They are

1. Union of two languages, L and M denoted by $L \cup M$ is set of strings that are in either of L or M or both

Ex:- If $L = \{0, 01, 110, 1010\}$ and $M = \{\epsilon, 1, 01\}$ then

$L \cup M = \{\epsilon, 0, 1, 01, 110, 1010\}$

2. Concatenation of languages : L & M denoted by $L \cdot M$ is set of strings that can be formed by taking any string in L and concatenating it with any string in M.

→ To denote concatenation of languages L & M either with dot operator (·) or no operator at all i.e, $L \cdot M$ or LM

Ex:- If $L = \{0, 01, 110, 1010\}$, $M = \{\epsilon, 1, 01\}$

$LM = \{0, 01, 110, 1010, 01, \underline{011}, \underline{1101}, 10101, 001, 0101, 11001, 101001\}$

3. Closure (star closure / Kleen closure) of language L is denoted by L^* and represent the set of those strings that can be formed by taking any no. of strings from L.

Ex:-

$$L^* = L^0 \cup L^1 \cup L^2 \cup \dots = \bigcup_{i=0}^{\infty} L^i$$

$L_i^0 = \{\epsilon\}$

if $L = \{a, b\}$ then $L = \{a, b\}$

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

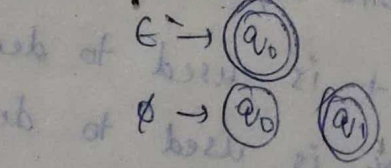
$L^* = \{a, b, ab, ba, \dots\}$

Note :-

1. Closure of language $\emptyset$ is denoted by $\emptyset^*$, $\emptyset = \{\}$

$\emptyset^* = \{\epsilon\}$

2. closure of the empty string ϵ is denoted by $\epsilon^* = \epsilon$



→ Write the regular expressions RE = a^* for language accepting all combination of a's over $\Sigma = \{a\}$ except null string.

$RE = a^+$

→ Write the regular exp. for language accepting all combination of a's over $\Sigma = \{a\}$, $RE = a^*$

→ Design regular exp. for language accepting all strings of all combinations of a's and b's.

$(a^*.b^*)^* = (a+b)^* =$ Any combinations of a's & b's.

→ Design regular exp. for language accepting all strings having any no. of a's & b's except null string.

$(a+b)^+$

→ Construct regular exp for language accepting all strings which are ending with 00. over $\Sigma = \{0,1\}$

$RE = (0+1)^*00$

→ Construct regular exp. for language accepting strings which are starting with 1 ending with 0 over $\Sigma = \{0,1\}$

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

→ Construct RE with L which accepts starting and ending with a and having any combination of b's in b/w.

$RE = ab^*a$

→ Write RE to denote language L over Σ^* where $\Sigma = \{a,b,c\}$ in which every string will be such that any no. of a's followed by any no. of b's followed by any no. of c's.

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

→ Write RE for L accepts every string having atleast one a followed by atleast one b by atleast one b.

$RE = a^+b^+c^+$

→ Find regular Exp. for L which accepts strings such that 3rd character from right end of the string is always a over $\Sigma = \{a,b\}$

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

→ atleast two b's, $\Sigma = \{a, b\}$

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

→ exactly two b's, $\Sigma = \{a, b\}$

$a^* b a^* b a^*$

→ 101 as substring

$(1+0)^* 101 (1+0)^*$

→ two consecutive b's, $\Sigma = \{a, b\}$

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

→ begin or end with either 00 or 11

case 1 begin with 00 or 11

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

case 2 end with 00 or 11

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

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

Identity rules for regular expressions:-

If P, Q & R are regular exp. then the identity rules are given below.

1. $\varepsilon \cdot R = R, \varepsilon = R$
2. $\varepsilon^* = \varepsilon$
3. $\emptyset^* = \varepsilon$
4. $\emptyset \cdot R = R \cdot \emptyset = \emptyset$
5. $\emptyset + R = R + \emptyset = R$
6. $R + R = R$
7. $R \cdot R^* = R^* \cdot R = R^+$
8. $(R^*)^* = R$
9. $\varepsilon + R \cdot R^* = \varepsilon + R^* \cdot R = R^*$
10. $(P+Q) \cdot R = PR + QR$
11. $(P+Q)^* = (P^* Q^*)^* = (P^* + Q^*)^*$
12. $R^*(\varepsilon + R) = (\varepsilon + R) R^* = R^*$
13. $(\varepsilon + R)^* = (R + \varepsilon)^* = R^*$
14. $\varepsilon + R^* = R^*$
15. $(PQ)^* \cdot P = P(QP)^*$

16. $R^*R + R = R^*R$

Prove the following

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

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

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

$= (\varepsilon+1)[\varepsilon + (\varepsilon+1)^+]$

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

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

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

$= 1^* =$ RHS.

2] Simplify $(1+00^*1) + (1+00^*1)(0+10^*1)^* (0+10^*1)$

$= (1+00^*1)(\varepsilon + (0+10^*1)^*(0+10^*1))$

$= (1+00^*1)(0+10^*1)^*$

$= (\varepsilon+00^*)1(0+10^*1)^*$

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

3] Simplify $0+0(\varepsilon+1)^*(\varepsilon+1)$

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

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

$= 01^*$

Conversion of DFA to regular Expressions :-

If $L = L(A)$ for some DFA, A then there is a R.E $\in R$ such that $L = L(R)$

Proof :-

Basis $R_{ij}(k)$ is a R.E whose language is set of strings w such that w is the variable of path from state i to state j in A and the path has no intermediate node whose number is greater than k.

To Construct R.E $R_{ij}(k)$ at $k = 0$ and finally reaching $k = n$

where n indicates no of states.

for k=0, there are only two kinds of paths that meet such a condition

(i) an arc from state i to state j

(ii) path of length '0' that consists of only some node 'i' if i≠j

(a) if there is no symbol, then $R_{ij}^{(0)} = \phi$

(b) if there is exactly one symbol 'a' then $R_{ij}^{(0)} = a$

(c) if there are symbols $a_1, a_2, \ldots a_k$ then $R_{ij}^{(0)} = a_1 + a_2 + \ldots + a_k$

(iii) a path of length '0' that consists of only some node 'i', if i=j

(a) if there is no symbol, then $R_{ij}^{(0)} = \varepsilon + \phi$

(b) if there is exactly one symbol 'a' then $R_{ij}^{(0)} = \varepsilon + a$

(c) if there are symbols $a_1, a_2, \ldots a_k$, $R_{ij}^{(0)} = \varepsilon + a_1 + a_2 + \ldots + a_k$

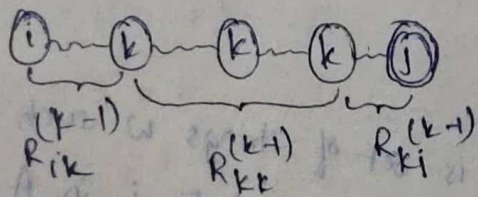
Induction:-

Suppose there is a path from state 'i' to state 'j' that goes through no state higher than k

There are two possible cases to consider

(i) The path that doesn't go through state 'k' at all label of the path is $R_{ij}^{(k-1)}$

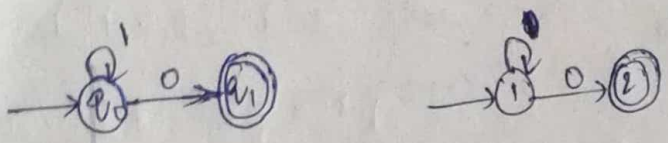
(ii) The path goes through the state 'k' atleast once then we can break the path into several piecel.



To combine the expressions of two types

$$R_{ij}^{(k)} = R_{ij}^{(k-1)} + R_{ik}^{(k-1)} \cdot (R_{kk}^{(k-1)})^{*} \cdot R_{kj}^{(k-1)}$$

To convert the following DFA to regular expression



Sol :-

at $k=0$ | at $k=1$

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

$R_{12}^{(0)} = 0$ | $R_{12}^{(1)} =$

$R_{21}^{(0)} = \phi$ | $R_{21}^{(1)} =$

$R_{22}^{(0)} = \varepsilon + \phi = \varepsilon$ | $R_{22}^{(1)} =$

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

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

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

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

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

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

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

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

$$= 1^* 0$$

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

$$= \phi + \phi = \phi$$

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

$$= \varepsilon + \phi = \varepsilon$$

$k=2$

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

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

$$= 1^*$$

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

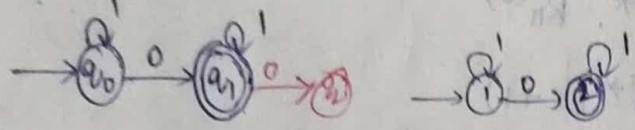
$$= 1^*0 + 1^*0 (\varepsilon)^* \varepsilon$$

$$= 1^*0 [\varepsilon + \varepsilon] = 1^*0$$

Q7

Ans:

Find the RE from following DFA



at k=0

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

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

$R_{21}^{(0)} = \emptyset$

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

at k=1

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

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

$= 1^*$

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

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

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

$= \emptyset + \emptyset \cdot = \emptyset$

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

$= (\varepsilon+1) + \emptyset = \varepsilon + 1$

at k=2

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

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

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

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

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

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

$= 1^*0 \cdot 1^*$

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

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

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

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

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

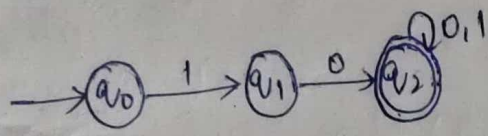
$= (\varepsilon+1)(\varepsilon+1)^*$

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

$= 1^*$

The R.E of DFA is 1^*01^*

Construct R.E for following DFA



K=0

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

$R_{12}^{(0)} = 1$

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

$R_{21}^{(0)} = \emptyset$

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

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

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

$R_{32}^{(0)} = \emptyset$

$R_{33}^{(0)} = \varepsilon + (0+1)^*$

K=1

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

$= \varepsilon + \varepsilon \cdot \varepsilon^* \varepsilon = \varepsilon$

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

$= 1 + \varepsilon (\varepsilon^*) 1 \quad R_{13}^{(1)} = \emptyset$

$= 1 + 1 = 1$

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

$= \emptyset + \emptyset = \emptyset$

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

$= \varepsilon + \emptyset = \varepsilon$

$R_{23}^{(1)} = R_{23}^{(0)} + R_{21}^{(0)} (R_{11}^{(0)})^* R_{13}^{(0)}$

$= 0 + \emptyset = 0$

$$R_{31}^{(1)} = R_{31}^{(0)} + R_{31}^{(0)} (R_{11}^{(0)})^* R_{11}^{(0)}$$

$$= \phi + \phi = \phi$$

$$R_{32}^{(1)} = R_{32}^{(0)} + R_{31}^{(0)} (R_{11}^{(0)})^* R_{12}^{(0)}$$

$$= \phi + \phi = \phi$$

$$R_{33}^{(1)} = R_{33}^{(0)} + R_{31}^{(0)} (R_{11}^{(0)})^* R_{13}^{(0)}$$

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

$$= \varepsilon + 0 + 1$$

<u>k = 2</u>

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

$$= \varepsilon + 1\varepsilon^* \phi = \varepsilon$$

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

$$= 1 + 1 \cdot \varepsilon^* \varepsilon = 1$$

$$R_{13}^{(2)} = R_{13}^{(1)} + R_{12}^{(1)} (R_{22}^{(1)})^* R_{23}^{(1)}$$

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

$$= 1\varepsilon 0$$

$$= 10$$

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

$$= \phi + \varepsilon \cdot \varepsilon^* \phi = \phi$$

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

$$= \varepsilon + \varepsilon \cdot \varepsilon^* \varepsilon = \varepsilon$$

$$R_{23}^{(2)} = R_{23}^{(1)} + R_{22}^{(1)} (R_{22}^{(1)})^* R_{23}^{(1)}$$

$$= 0 + \varepsilon \cdot \varepsilon^* 0$$

$$= 0 + 0 = 0$$

$$R_{31}^{(2)} = R_{31}^{(1)} + R_{32}^{(1)} (R_{22}^{(1)})^* R_{21}^{(1)}$$

$$= \phi + \phi = \phi$$

$$R_{32}^{(2)} = R_{32}^{(1)} + R_{32}^{(1)} (R_{22}^{(1)})^* R_{22}^{(1)}$$

$$= \phi$$

$R_{33}^{(2)} = R_{33}^{(1)} + R_{32}^{(1)} (R_{22}^{(1)})^* R_{23}^{(1)}$

$= \varepsilon + 0 + 1 + \phi$

$= \varepsilon + 0 + 1$

<u>K=3</u>

$R_{13}^{(3)} = R_{13}^{(2)} + R_{13}^{(2)} (R_{33}^{(2)})^* R_{33}^{(2)}$

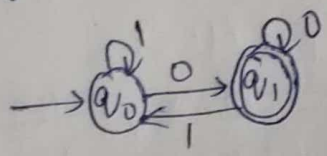
$= 10 + 10 (\varepsilon + 0 + 1)^* (\varepsilon + 0 + 1)$

$= 10 (\varepsilon + (\varepsilon + 0 + 1)^* (\varepsilon + 0 + 1))$

$= 10 (\varepsilon + 0 + 1)^*$

$= 10 (0 + 1)^*$

Design RE for following DFA



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

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

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

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

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

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

$= 1^*$

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

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

$= 1^* 0$

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

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

$= 1 \cdot 1^*$

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

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

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

$= \varepsilon + 1^* 0$

k=2

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

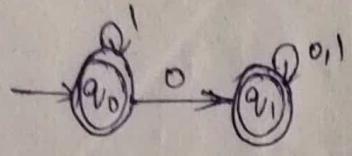
~~$1^*0 + 1^*0 (\varepsilon+1^*0)^* (\varepsilon+1^*0)$~~ $= 1^*0 [\varepsilon + ((\varepsilon+0) + 11^*0))^* ((\varepsilon+0) + 11^*0)]$

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

~~$= 1^*0 (1^*0)^*$~~ $= 1^*0 [((\varepsilon+0) + 11^*0)^*]$

~~$= (1^*0)^*$~~

Design R.E for following DFA.



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

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

$R_{21}^{(0)} = \emptyset$

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

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

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

$= 1^*$

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

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

$= \{1^*0$

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

$= \emptyset + \emptyset = \emptyset$

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

$= \varepsilon + 0 + 1 + \emptyset = \varepsilon + 0 + 1$

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

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

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

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

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

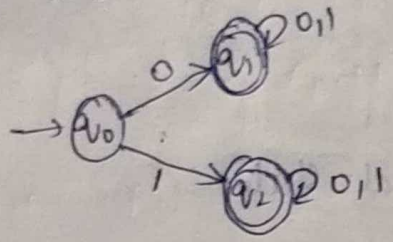
$= 1^*$

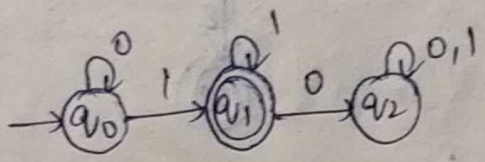
$R.E = R_{11}^{(2)} + R_{12}^{(2)}$

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

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

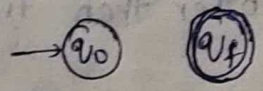
Find R.E for following DFA





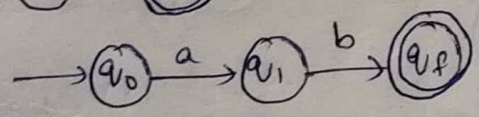
Conversion of FA to RE by elimination of states :-

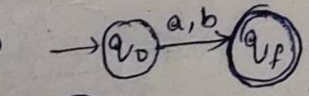
Basic notations for RE to FA :-

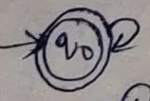
1. RE = $\emptyset$ ⇒ 

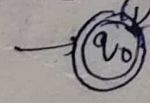
2. RE = ε ⇒ 

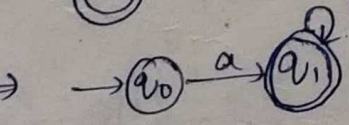
3. RE = a ⇒

4. RE = ab ⇒ 

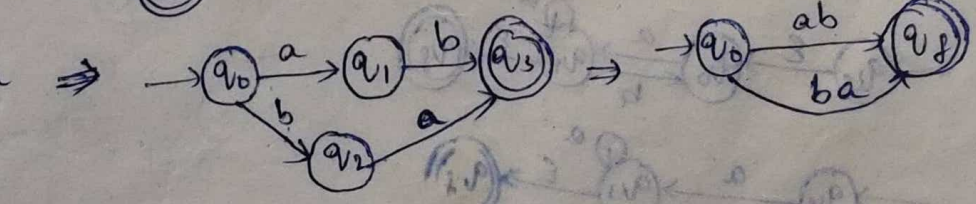
5. RE = a+b ⇒ 

6. RE = a^* ⇒ 

7. RE = $(ab)^*$ ⇒ 

8. RE = ab^* ⇒ 

9. RE = $(a+b)^*$ ⇒ 

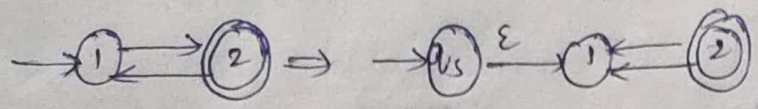
10. RE = ab+ba ⇒ 

Method :-

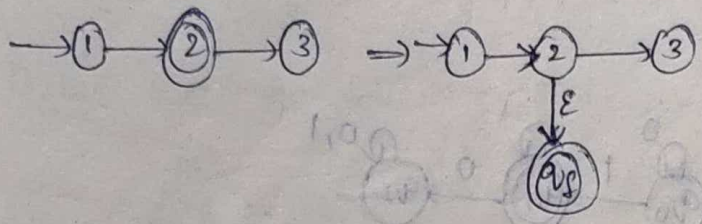
By using state elimination method, we can convert DFA to RE

The following conversion specifies DFA to RE

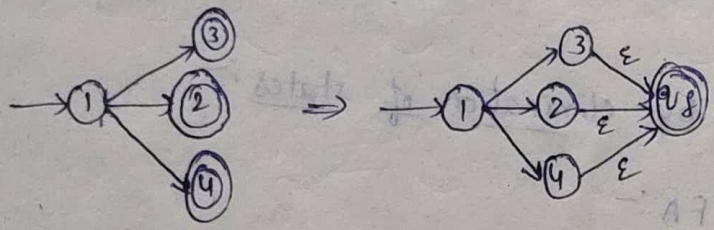
step1:- Initial state donot have any incoming state (edge)



step2 :- (a) For final state, there should not be any outgoing state

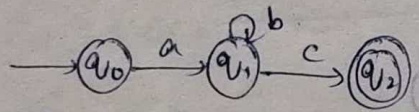


(b) should not have more than one final state

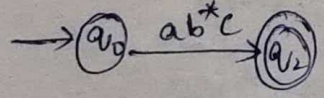


step3 :- Eliminate all remaining states other than final and start states in any order.

Ex:- 1] Convert the following DFA to RE by elimination of states meth

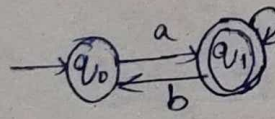


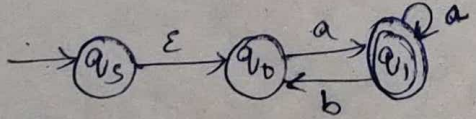
Sol:- Eliminate state q_1

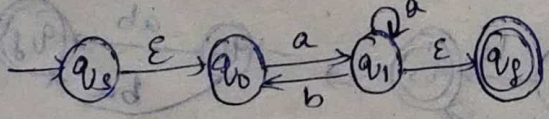


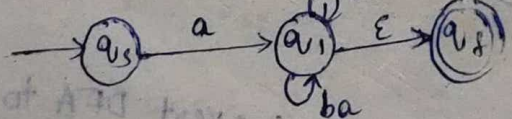
R.E = ab^*c

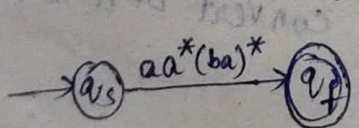
2]



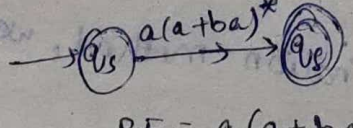






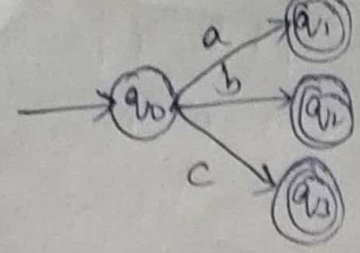


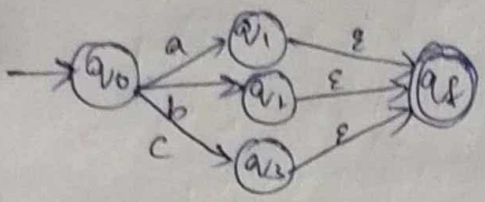
(or)



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

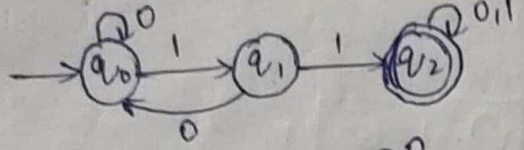
3]

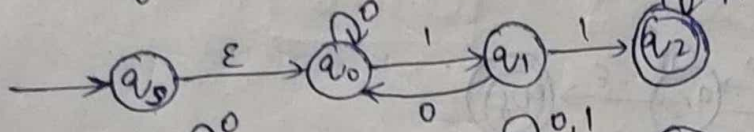


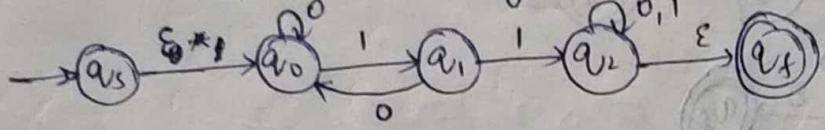


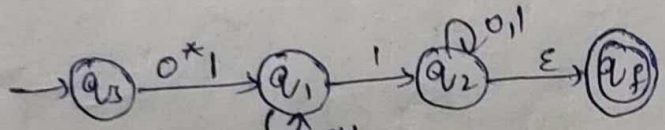
R.E = (a+b+c)

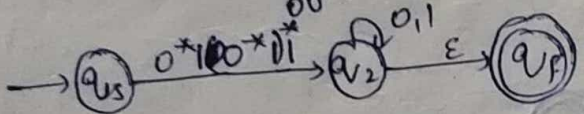
4]

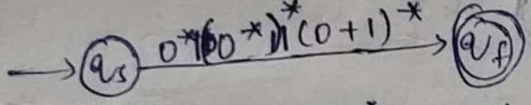






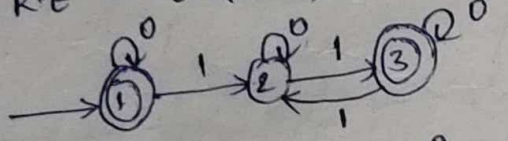


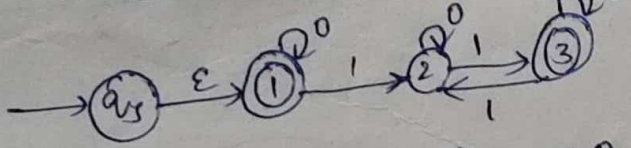


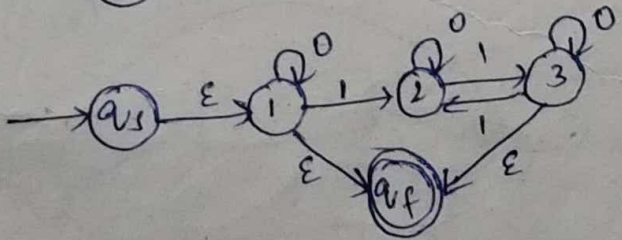


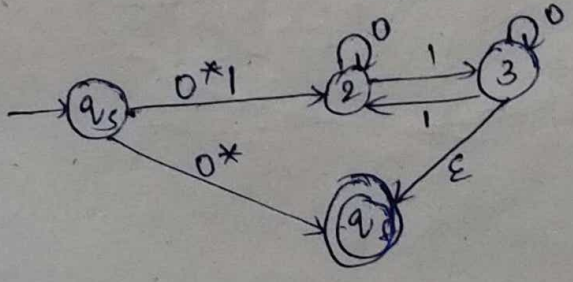
R.E = $0^*1(00^*1)^*(0+1)^*$

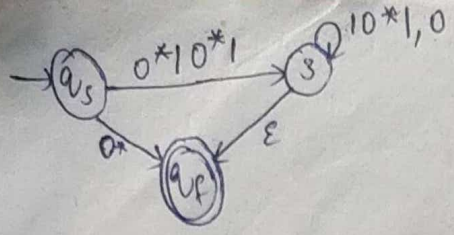
5]

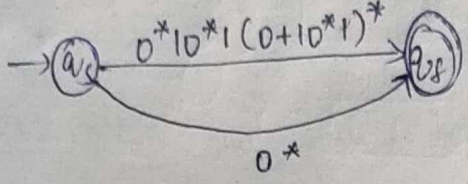






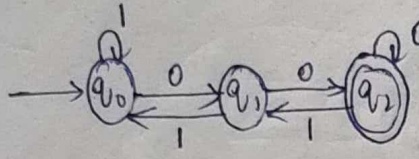


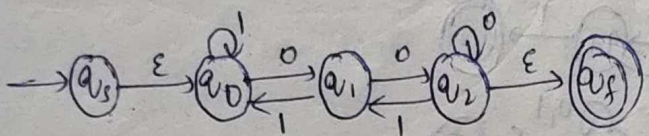


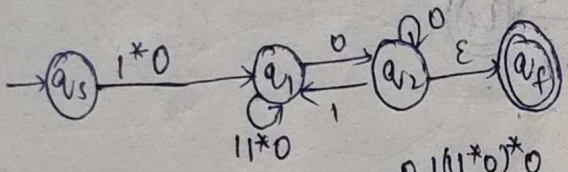


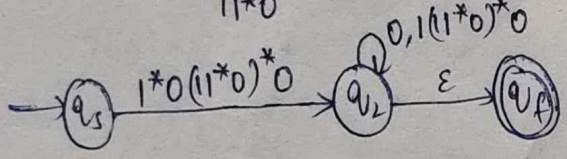
R.E = $0^* + 0^*10^*1\,(10^*1+0)^*$

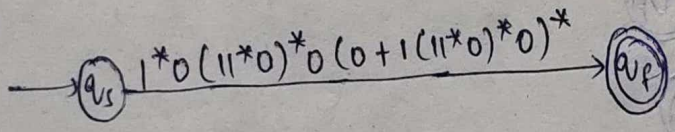
6]





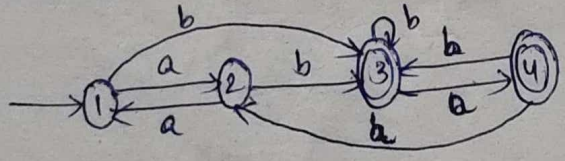


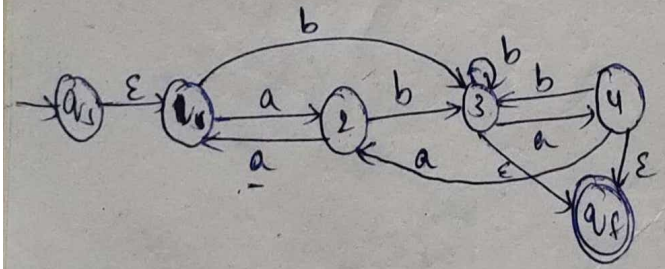


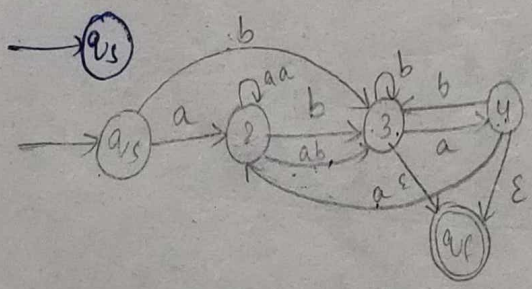


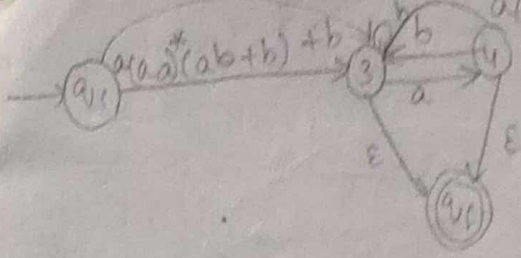
R.E = $1^*0(11^*0)^*0(0+1(11^*0)^*0)^*$

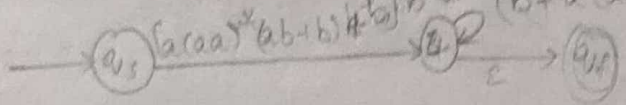
7]











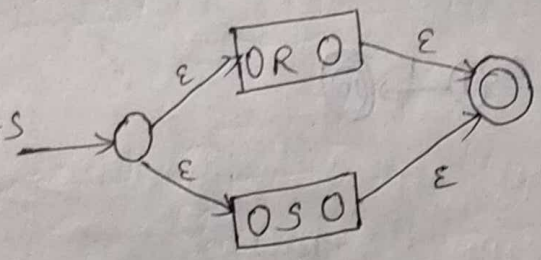
$R.E = [a(aa)^*(ab+b)+b]\, b^*a\, [(b+a(aa)^*(ab+b))b^*a]^*$

Construction of ε-NFA from R.E :-

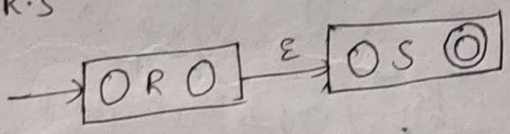
Every language

Some of the basic notations from R.E is as follows

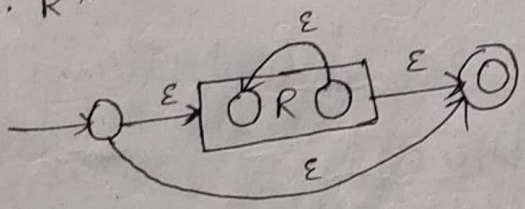
1. R+S



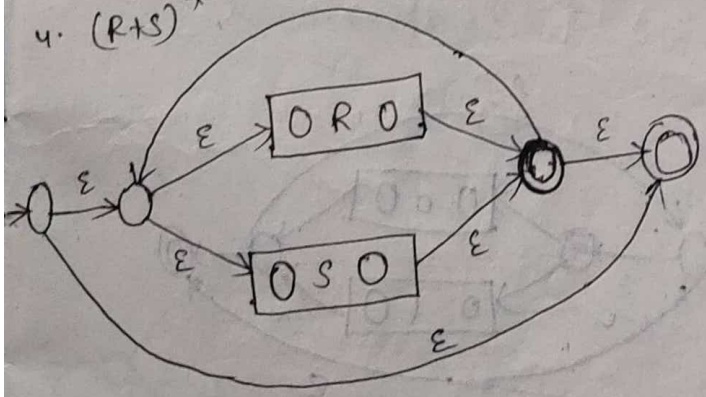
2. R.S



3. R*



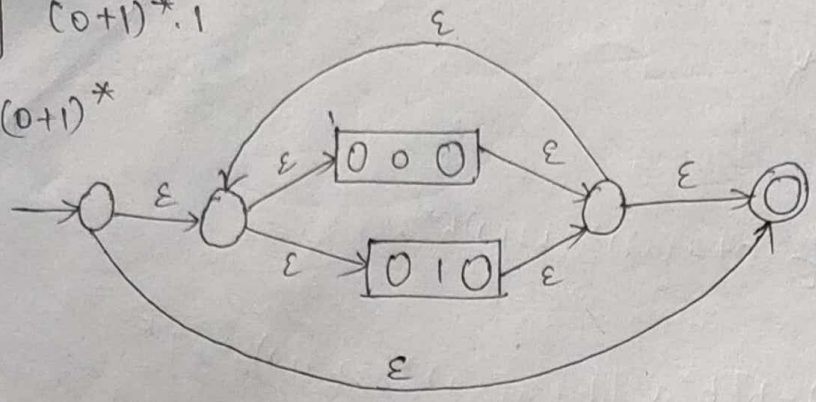
4. (R+S)*



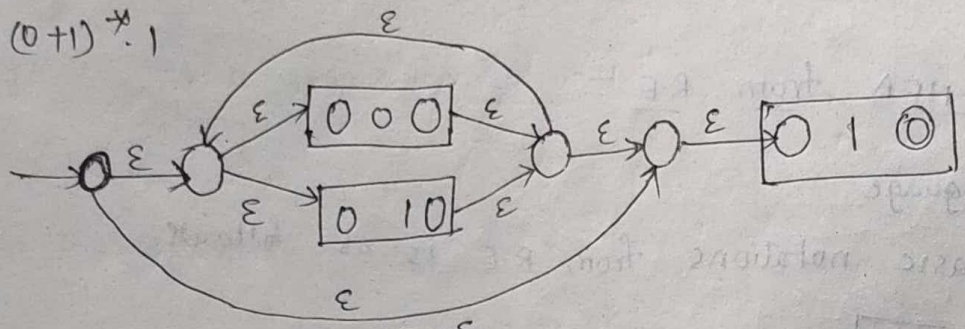
Construct ε-NFA for following RE

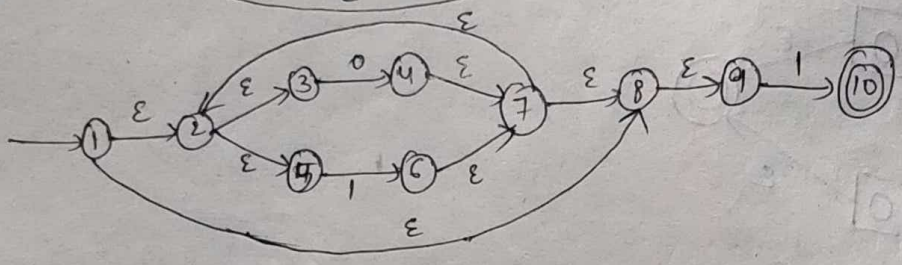
1] $(0+1)^*.1$

$(0+1)^*$



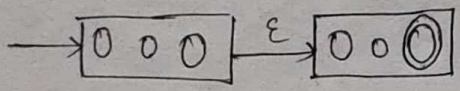
$(0+1)^*.1$



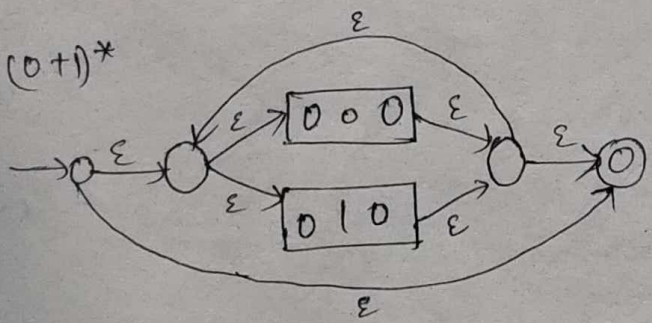


2] $00(0+1)^*$

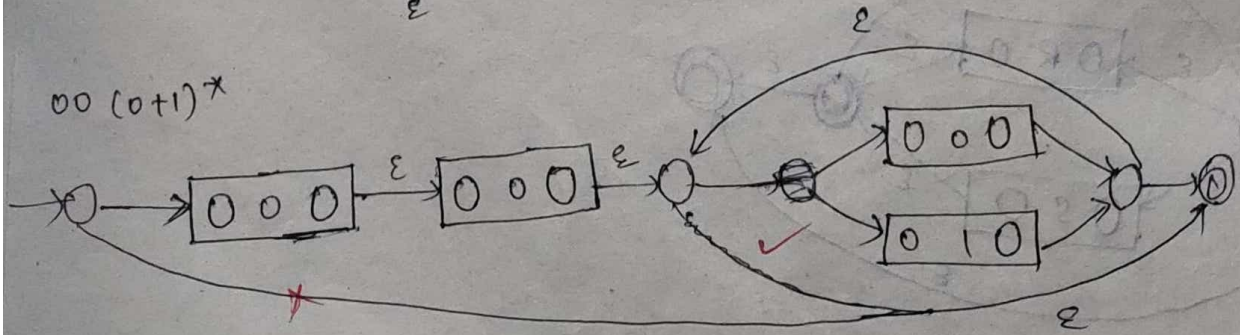
00

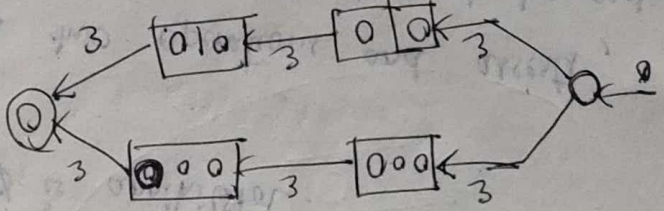


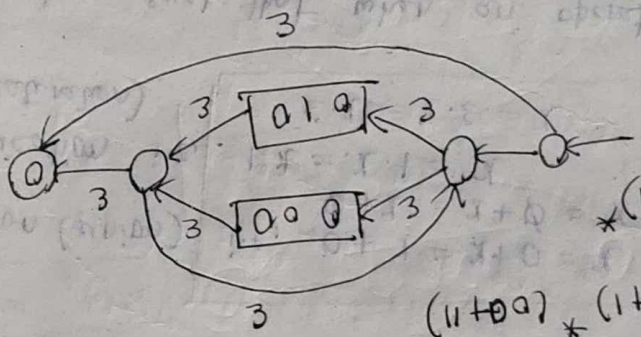
$(0+1)^*$

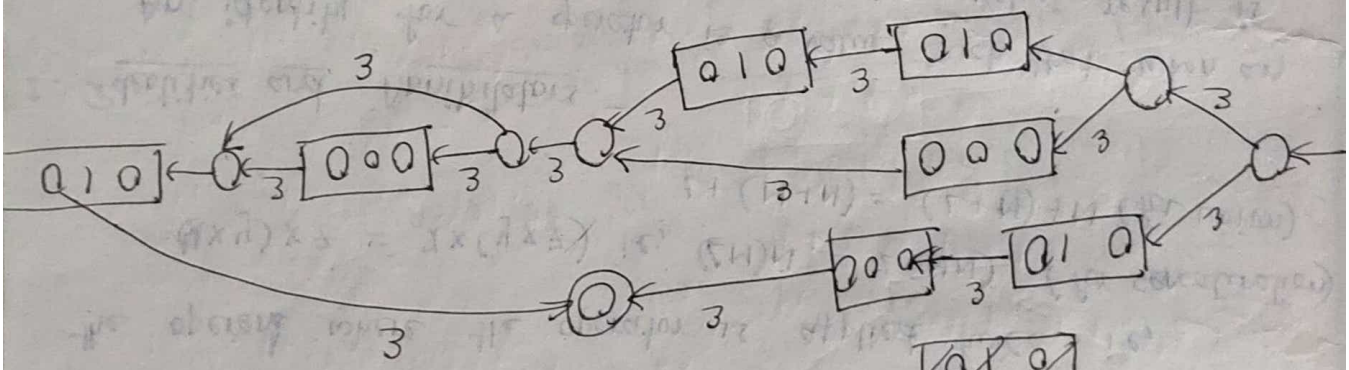


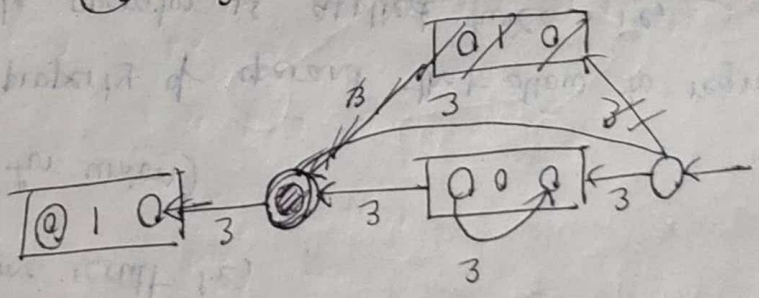
$00(0+1)^*$

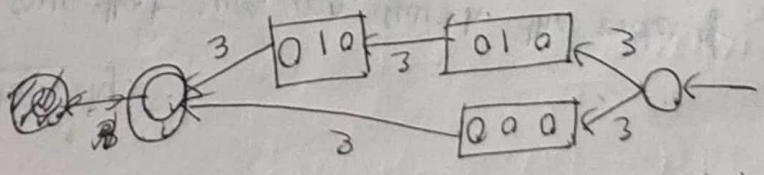


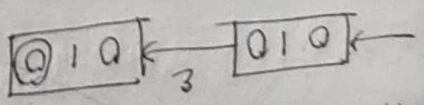

3
3
3
3
3
3
3

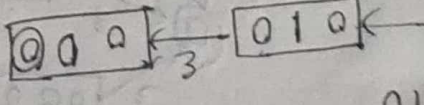

3
3
3
3
3
3
3


3
3
3
3
3
3
3
3
3
3
3

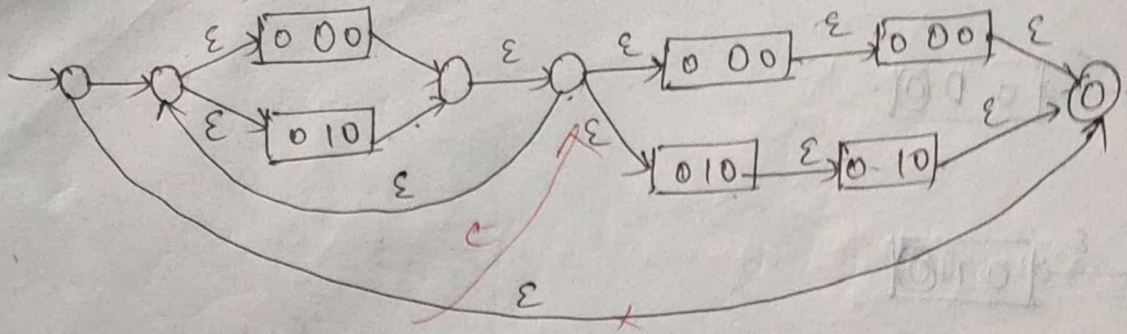

3
3
3
3
3
3


3
3
3
3


3


3

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



Algebraic laws of R.E :-

1. Associativity and commutativity :-

Commutative property says that we can switch the order of operands components and get same result i.e,

$$x+y = y+x \text{ (for union)}$$

Associativity is a property of operand that allow to regroove the operand where the operator is applied twice, i.e,

$(x \times y) \times z = x \times (y \times z)$ i.e, $(LM)N = L(MN)$ (for concatination)

$L+(M+N) = (L+M)+N$ (for union)

2. Identities and Annihilators :-

An identity for a operator is a value such that when an operator is applied to identity and some other value, result is other value.

'0' is identity for addition (union)

'1' is identity for multiplication (concatination) i.e,

$$0+x = x+0 = x$$
$$\phi+x = x+\phi = x$$
$$1 \cdot x = x \cdot 1 = x$$
$$\varepsilon \cdot x = x \cdot \varepsilon = x$$

Annihilator for a operator such that when an operator is applied to identity, the result is any of the values

Ex :- $L \cdot \phi = \phi \cdot L = \phi$ // ϕ is annihilator.

3. Distributive law :-

Distributive law involves two operations and asserts, one assert is that operators can be pushed down to be applied on each argument or an outer operator individually.

Ex:-

$(L+M)N = LN + MN$

$L(M+N) = LM+LN$

4. Idempotent law:- An operator is said to be idempotent if the result applied on two same values as argument is that value

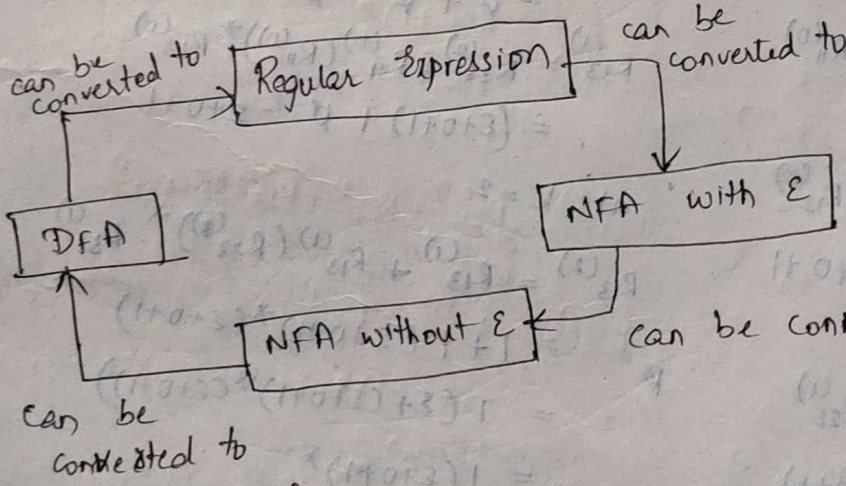
Ex:- $L+L = L$

5. Laws of closures:- There are no. of laws involved in closure operator

1. $(L^*)^* = L$
2. $\emptyset^* = \varepsilon^* = \varepsilon$
3. $L \cdot L^* = L^* \cdot L = L^+$
4. $(L^* M^*)^* = (L+M)^*$
5. $L^* + \varepsilon = L^*$
6. $(\varepsilon + L)^* = \varepsilon + LL^* = L^*$
7. $L^? = \varepsilon + L$

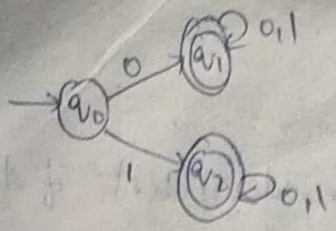
Relation b/w F.A & R.E :-

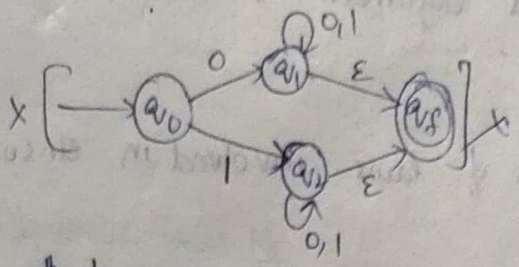
There is a close relationship b/w FA & RE shown below

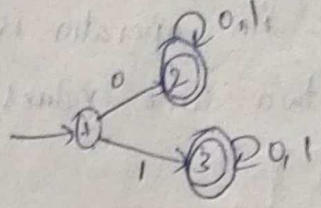


Minimisation of FA :-

1]







at $k=0$

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

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

$R_{21}^{(0)} = \emptyset$

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

at $k=1$

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

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

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

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

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

$= \emptyset + \emptyset = \emptyset$

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

$= (\varepsilon + 0 + 1) + \emptyset = \varepsilon + 0 + 1$

at $k=2$

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

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

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

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

$= 0 (0 + 1)^*$

at $k=0$

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

$R_{13}^{(0)} = 1$

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

$R_{33}^{(0)} = \varepsilon + 0 + 1$

at $k=1$

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

$R_{13}^{(1)} = R_{13}^{(0)} + R_{11}^{(0)} (R_{11}^{(0)})^* R_{13}^{(0)}$

$= 1 + \varepsilon \cdot \varepsilon^* 1 = 1$

$R_{31}^{(1)} = R_{31}^{(0)} + R_{31}^{(0)} (R_{11}^{(0)})^* R_{11}^{(0)}$

$= \emptyset + \emptyset = \emptyset$

$R_{33}^{(1)} = R_{33}^{(0)} + R_{31}^{(0)} (R_{11}^{(0)})^* R_{13}^{(0)}$

$= (\varepsilon + 0 + 1) + \emptyset = \varepsilon + 0 + 1$

at $k=2$

$R_{13}^{(2)} = R_{13}^{(1)} + R_{13}^{(1)} (R_{33}^{(1)})^* R_{33}^{(1)}$

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

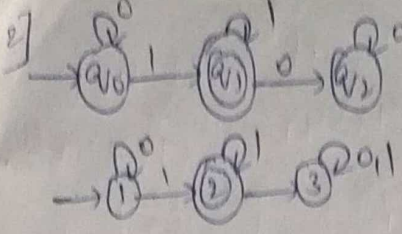
$= 1 (\varepsilon + (\varepsilon + 0 + 1)^* (\varepsilon + 0 + 1))$

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

$= 1 (0 + 1)^*$

R.E of given DFA $= 0 (0+1)^* + 1 (0+1)^*$

2]



at $k=0$

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

$R_{12}^{(0)} = 1$

$R_{21}^{(0)} = \emptyset$

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

at $k=1$

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

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

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

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

$= 0^*$

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

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

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

$= 0^* 1$

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

$= \emptyset + \emptyset = \emptyset$

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

$= \varepsilon + 1 + \emptyset = \varepsilon + 1$

at $k=2$

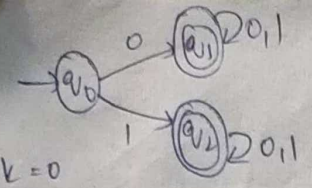
$R_{12}^{(2)} = R_{12}^{(1)} + R_{12}^{(1)} (R_{22}^{(1)})^* R_{22}^{(1)}$

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

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

$= 0^*11^*$

1]



at $k=0$

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

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

$R_{13}^{(0)} = 1$

$R_{21}^{(0)} = \emptyset$

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

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

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

$R_{32}^{(0)} = \emptyset$

$R_{33}^{(0)} = \varepsilon + 0 + 1$

at $k=1$

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

$= \varepsilon$

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

$= 0 + 0 = 0$

$R_{13}^{(1)} = R_{13}^{(0)} + R_{11}^{(0)} (R_{11}^{(0)})^* R_{13}^{(0)}$

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

$= 1$

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

$= \emptyset + \emptyset = \emptyset$

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

$= (\varepsilon + 0 + 1) + \emptyset$

$= \varepsilon + 0 + 1$

$R_{23}^{(1)} = R_{23}^{(0)} + R_{21}^{(0)} (R_{11}^{(0)})^* R_{13}^{(0)}$

$= \emptyset + \emptyset = \emptyset$

$R_{31}^{(1)} = R_{31}^{(0)} + R_{31}^{(0)} (R_{11}^{(0)})^* R_{11}^{(0)}$

$= \emptyset + \emptyset = \emptyset$

$R_{32}^{(1)} = R_{32}^{(0)} + R_{31}^{(0)} (R_{11}^{(0)})^* R_{12}^{(0)}$

$= \emptyset + \emptyset = \emptyset$

$R_{33}^{(1)} = R_{33}^{(0)} + R_{31}^{(0)} (R_{11}^{(0)})^* R_{13}^{(0)}$

$= \varepsilon + 0 + 1$

at $k=2$

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

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

$= \varepsilon$

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

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

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

$= 0(0+1)^*$

$R_{13}^{(2)} = R_{13}^{(1)} + R_{12}^{(1)} (R_{22}^{(1)})^* R_{23}^{(1)}$

$= 1 + \emptyset = 1$

$R_{21}^{(2)} = R_{21}^{(1)} + R_{22}^{(1)} (R_{22}^{(1)})^* \cdot R_{21}^{(1)}$

$= \emptyset$

$R_{22}^{(2)} = R_{22}^{(1)} + R_{22}^{(1)} (R_{22}^{(1)})^* \cdot R_{22}^{(1)}$

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

$= (0+1)^*$

$R_{23}^{(2)} = R_{23}^{(1)} + R_{22}^{(1)} (R_{22}^{(1)})^* \cdot R_{23}^{(1)}$

$= \emptyset$

$R_{31}^{(2)} = R_{31}^{(1)} + R_{32}^{(1)} (R_{22}^{(1)})^* R_{21}^{(1)}$

$= \emptyset$

$R_{32}^{(2)} = R_{32}^{(1)} + R_{32}^{(1)} (R_{22}^{(1)})^* R_{22}^{(1)} = \emptyset$

$R_{33}^{(2)} = R_{33}^{(1)} + \emptyset$

$= \varepsilon + 0 + 1$

at $k=3$

$R_{11}^{(3)} = R_{11}^{(2)} + R_{13}^{(2)} (R_{33}^{(2)})^* R_{31}^{(2)}$

$= \varepsilon + 1 (\varepsilon + 0 + 1) \emptyset$

$= \varepsilon$

$R_{12}^{(3)} = R_{12}^{(2)} + R_{13}^{(2)} (R_{33}^{(2)})^* R_{32}^{(2)}$

$= 0(0+1)^*$

$R_{13}^{(3)} = R_{13}^{(2)} + R_{13}^{(2)} (R_{33}^{(2)})^* R_{33}^{(2)}$

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

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

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

$\therefore$ R.E for DFA is $0(0+1)^* + 1(0+1)^*$

Minimisation of FA (or) Optimisation of FA :-

To minimise the FA, table filling algorithm is used

Table Filling algorithm :- (finding distinguishable pairs)

Table Filling algorithm is recursive distingusible pairs of in DFA. $A = (Q, \Sigma, \delta, q_0, F)$

Basis :-

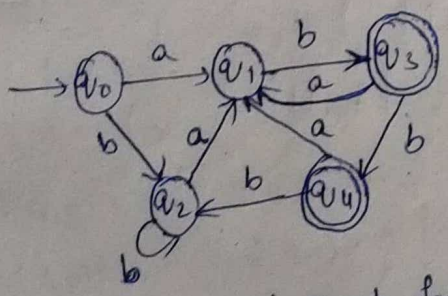
If p is an accepting state and q is non-accepting state then (p, q) is called distinguishable pair.

Induction :-

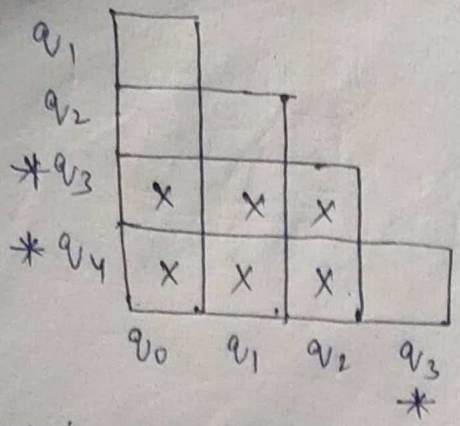
If p, q be two states, such that for some input symbol 'a', $r = \delta(p, a)$ and $s = (q, a)$ then the pair (p, q) are distinguishable

Minimise the following FA.

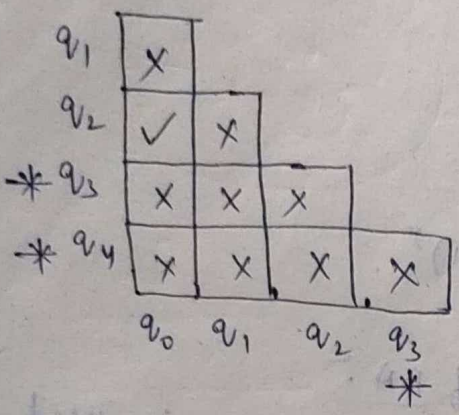
i]



Step 1 :- put 'x' mark for all final and non-final states.



Step 2 :- put 'x' for all unequal pairs



$\delta(q_1, a) = q_1$

$\delta(q_0, a) = q_1$

$\delta(q_1, b) = q_3$

$\delta(q_0, b) = q_2$

$\}$ $\therefore q_0 \neq q_1$ can never equal

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

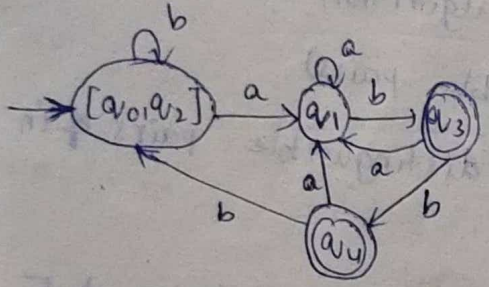
$\delta(q_0, a) = q_1$ $\quad \delta(q_0, b) = q_2$ $\}$ $q_2 = q_0$

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

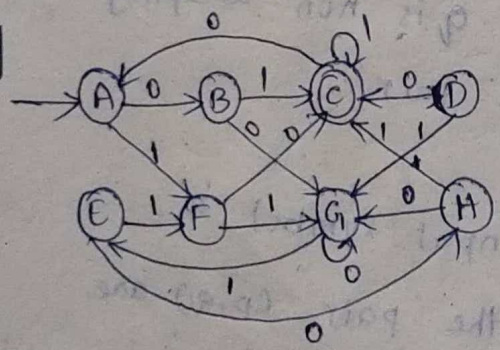
$\delta(q_1, a) = q_1$ $\quad \delta(q_1, b) = q_3$ $\}$ $q_1 \neq q_2$

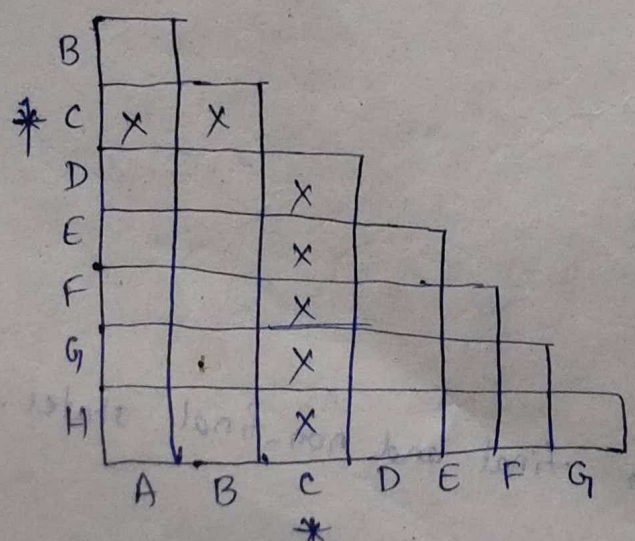
$\delta(q_4, a) = q_1$ $\quad \delta(q_4, b) = q_2$

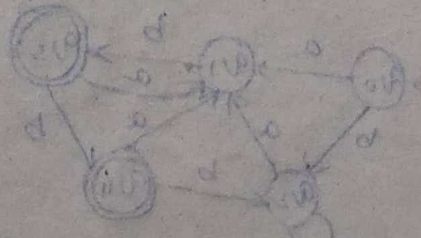
$\delta(q_3, a) = q_1$ $\quad \delta(q_3, b) = q_4$ $\}$ $q_3 \neq q_4$

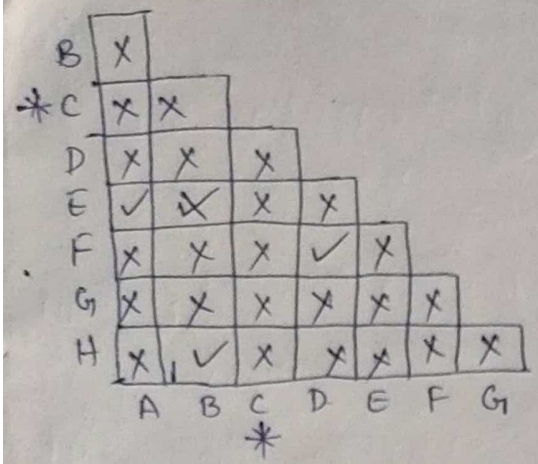


2]









$\delta(B,0) = G$

$\delta(A,0) = B$

$\delta(D,0) = C$

$\delta(A,0) = B$

$\delta(D,0) = C$

$\delta(B,0) = G$

$\delta(E,0) = H$

$\delta(A,0) = B$

$\delta(E,0) = G$

$\delta(B,0) = G$

$\delta(E,0) = G$

$\delta(D,0) = C$

$\delta(F,0) = C$

$\delta(A,0) = B$

$\delta(F,0) = C$

$\delta(B,0) = G$

$\delta(F,0) = C$

$\delta(D,0) = C$

$\delta(F,0) = C$

$\delta(E,0) = G$

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

$\delta(D,1) = G$
$\delta(A,1) = F$ } $A \neq D$

$\delta(D,1) = G$
$\delta(B,1) = C$ } $D \neq B$

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

$\delta(E,1) = F$
$\delta(B,1) = E$ } $B \neq E$

$\delta(E,1) = F$
$\delta(D,1) = G$ } $E \neq D$

$\delta(F,1) = G$
$\delta(A,1) = F$ } $A \neq F$

$\delta(F,1) = G$
$\delta(B,1) = C$ } $B \neq F$

$\delta(F,1) = G$
$\delta(D,1) = G$ } $F = D$

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

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

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

$\delta(G,0) = G$
$\delta(D,0) = C$

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

$\delta(G,1) = E$
$\delta(B,1) = C$ } $G \neq B$

$\delta(G,1) = E$
$\delta(D,1) = G$ } $G \neq D$

$\delta(G,0) = G$
$\delta(E,0) = G$
$\delta(G,0) = G$
$\delta(F,0) = C$
$\delta(H,0) = G$
$\delta(A,0) = B$
$\delta(H,0) = G$
$\delta(B,0) = G$
$\delta(H,0) = B$
$\delta(D,0) = C$
$\delta(H,0) = G$
$\delta(E,0) = G$
$\delta(H,0) = G$
$\delta(F,0) = C$
$\delta(H,0) = G$
$\delta(G,0) = G$

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

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

$\delta(H,1) = C$
$\delta(A,1) = F$ } $H \neq A$

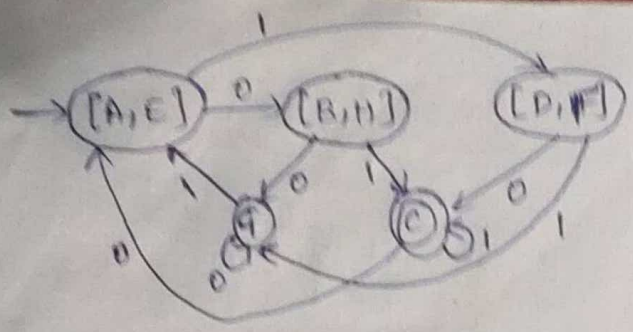
$\delta(H,1) = C$
$\delta(B,1) = C$ } $B = A$

$\delta(H,1) = C$
$\delta(D,1) = G$ } $H \neq D$

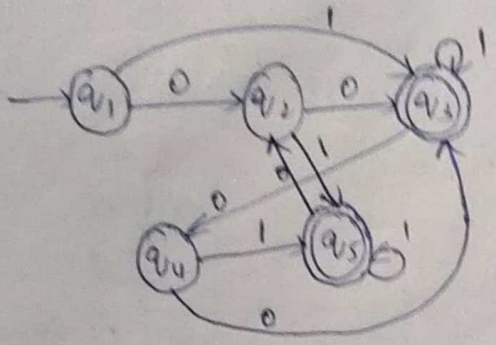
$\delta(H,1) = C$
$\delta(E,1) = F$ } $H \neq E$

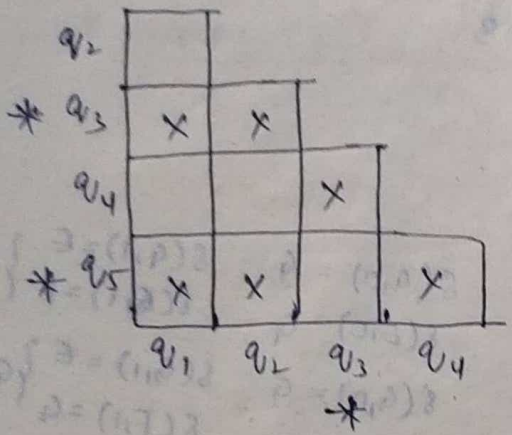
$\delta(H,1) = C$
$\delta(F,1) = G$ } $A \neq F$

$\delta(H,1) = C$
$\delta(G,1) = E$ } $H \neq G$



3]





$\delta(q_2,0) = q_3$ $\quad \delta(q_2,1) = q_5$ } $q_2 \neq q_1$

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

$\delta(q_4,0) = q_3$ $\quad \delta(q_4,1) = q_5$ } $q_4 \neq q_1$

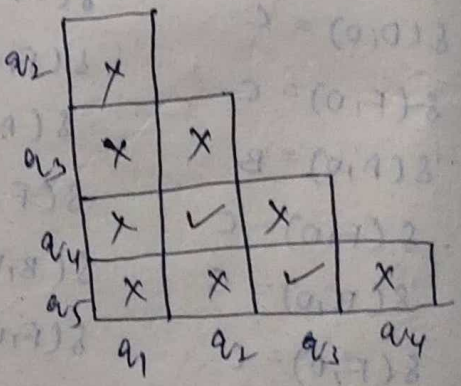
$\delta(q_4,0) = q_2$ $\quad \delta(q_1,1) = q_3$

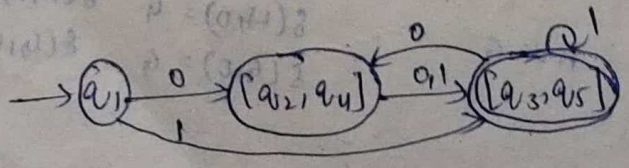
$\delta(q_4,0) = q_3$ $\quad \delta(q_4,1) = q_5$ } $q_4 = q_2$

$\delta(q_2,0) = q_3$ $\quad \delta(q_2,1) = q_5$

$\delta(q_5,0) = q_2$ $\quad \delta(q_5,1) = q_5$ } $q_3 = q_5$

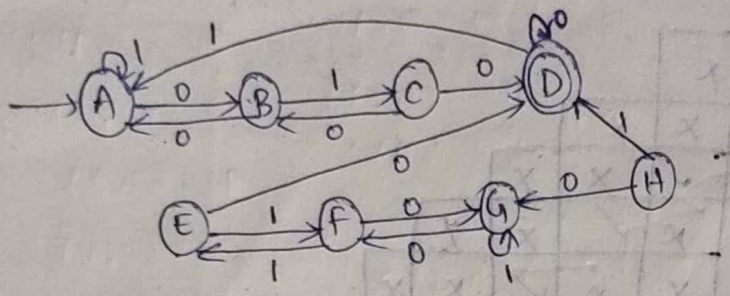
$\delta(q_3,0) = q_4$ $\quad \delta(q_3,1) = q_3$





4]

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



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

*

$\delta(G,0) = F$ $\delta(G,1) = G$ $\delta(H,1) = D$ } $G \neq H$

$\delta(H,0) = G$

$\delta(H,0) = G$ $\delta(H,1) = D$ $\delta(F,1) = E$ } $H \neq F$

$\delta(F,0) = G$

$\delta(H,0) = G$ $\delta(H,1) = D$ $\delta(E,1) = F$ } $H \neq E$

$\delta(E,0) = D$

$\delta(H,0) = G$ $\delta(H,1) = D$ $\delta(C,1) = B$ } $H \neq C$

$\delta(C,0) = D$

$\delta(H,0) = G$ $\delta(H,1) = D$ $\delta(B,1) = C$ } $H \neq B$

$\delta(B,0) = A$

$\delta(H,0) = G$ $\delta(H,1) = D$ $\delta(A,1) = A$ } $H \neq A$

$\delta(A,0) = B$

$\delta(G,0) = F$ $\delta(G,1) = G$ $\delta(F,1) = E$ } $G \neq F$

$\delta(F,0) = G$

$\delta(G,0) = F$ $\delta(G,1) = G$ $\delta(E,1) = F$ } $G \neq E$

$\delta(E,0) = D$

$\delta(G,0) = F$ $\delta(G,1) = G$ $\delta(G,1) = B$ } $G \neq C$

$\delta(C,0) = D$

$\delta(G,0) = F$ $\delta(G,1) = G$ $\delta(B,1) = C$ } $G \neq B$

$\delta(B,0) = A$

$\delta(G,0) = F$ $\delta(G,1) = G$ $\delta(A,1) = A$ } $G \neq A$

$\delta(A,0) = B$

B A
$C \neq A$

E C
$F \neq A$

$C \neq B$

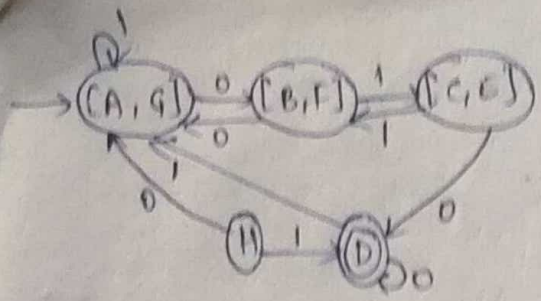
F B

$A \neq E$

$F \neq C$

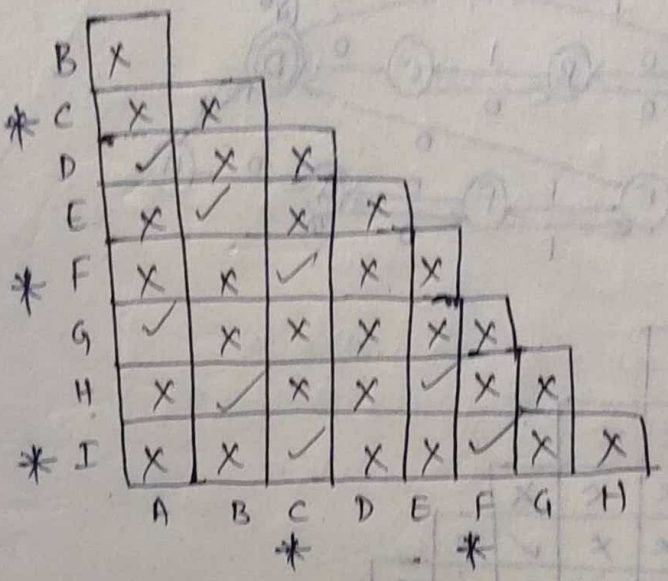
$B \neq E$

A G



3]

	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



$\delta(F,0) = G \quad \delta(F,1) = B$

$\delta(I,0) = A \quad \delta(I,1) = E$

$\delta(F,0) = G \quad \delta(F,1) = B$

$\delta(G,0) = D \quad \delta(G,1) = H$

$\delta(G,0) = H \quad \delta(G,1) = B$

$\delta(H,0) = I \quad \delta(H,1) = C$ } $G \neq H$

$\delta(A,0) = I \quad \delta(A,1) = C$

$\delta(E,0) = F \quad \delta(E,1) = I$ }

$\delta(H,0) = I \quad \delta(H,1) = C$

$\delta(D,0) = E \quad \delta(D,1) = H$ } $H \neq D$

$\delta(H,0) = I \quad \delta(H,1) = C$

$\delta(B,0) = C \quad \delta(B,1) = F$

$\delta(H,0) = I \quad \delta(H,1) = C$

$\delta(A,0) = B \quad \delta(A,1) = E$ } $A \neq H$

$\delta(G,0) = H \quad \delta(G,1) = B$

$\delta(A,0) = B \quad \delta(A,1) = E$

$\delta(E,0) = F \quad \delta(E,1) = I$ } $E \neq D$

$\delta(D,1) = H$

$\delta(G,0) = H \quad \delta(G,1) = B$ } $G \neq B$

$\delta(B,0) = C \quad \delta(B,1) = F$

$\delta(G,0) = H \quad \delta(G,1) = B$

$\delta(D,0) = E \quad \delta(D,1) = H$ } $G \neq D$

$\delta(G,0) = H \quad \delta(G,1) = D$

$\delta(E,0) = F \quad \delta(E,1) = I$ } $G \neq E$

$\delta(F,0) = G \quad \delta(F,1) = B$

$\delta(C,0) = D \quad \delta(C,1) = A$

$\delta(B,0) = C \quad \delta(B,1) = F$

$\delta(A,0) = B \quad \delta(A,1) = E$ } $A \neq B$

$\delta(B,0) = E \quad \delta(B,1) = H$

$\delta(A,0) = B \quad \delta(A,1) = E$

$\delta(D,0) = E \quad \delta(D,1) = H$

$\delta(B,0) = C \quad \delta(B,1) = F$ } $D \neq E$

$\delta(E,0) = F \quad \delta(E,1) = I$

$\delta(A,0) = B \quad \delta(A,1) = E$ } $A \neq E$

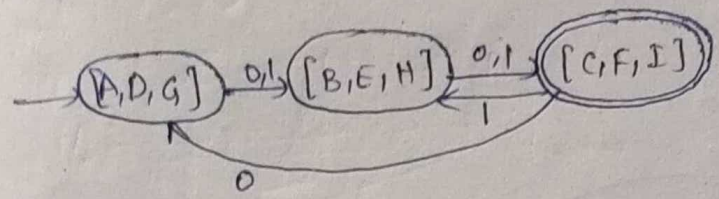
$\delta(E,0) = F \quad \delta(E,1) = I$

$\delta(B,0) = C \quad \delta(B,1) = F$

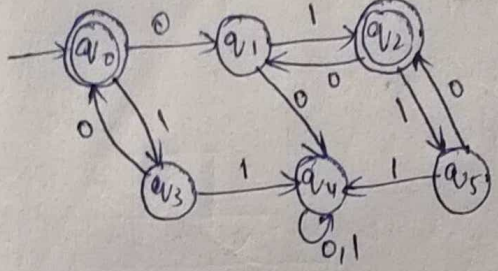
A = D
D = G , G = A → [A, D, G]

E = H
E = B , H = B → [B, E, H]

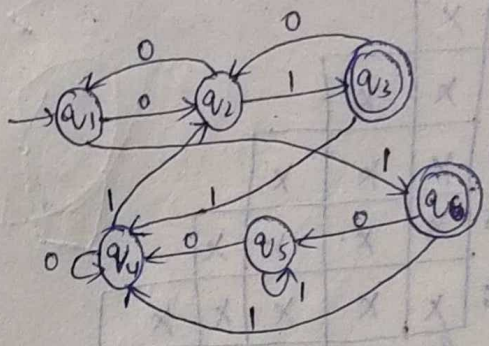
F = I
C = F , C = I → [C, F, I]



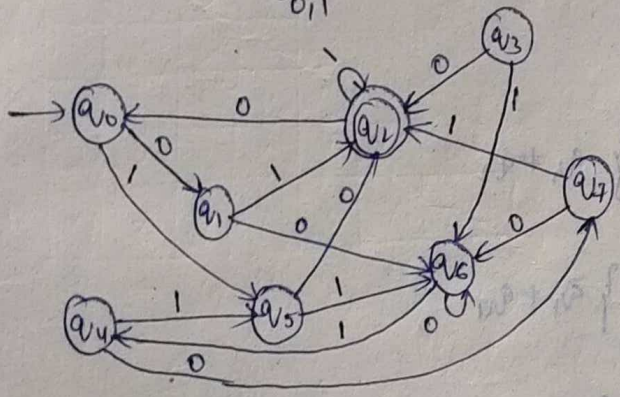
6]



7]



8]



6]

	q_0	q_1	q_2	q_3	q_4
q_1	X				
* q_2	✓	X			
q_3	X	X	X		
q_4	X	X	X	X	
q_5	X	X	X	✓	X

* * *

$\delta(q_2,0) = q_1$ $\delta(q_2,1) = q_5$

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

$\delta(q_3,0) = q_0$ $\delta(q_3,1) = q_4$
$\delta(q_1,0) = q_4$ $\delta(q_1,1) = q_2$ } $q_3 \neq q_1$

$\delta(q_3,0) = q_0$ $\delta(q_3,1) = q_4$
$\delta(q_1,0) = q_4$ $\delta(q_1,1) = q_2$ } $q_3 \neq q_1$

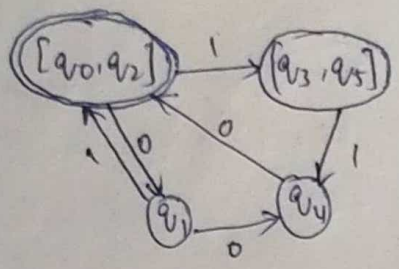
$\delta(q_4,0) = q_4$ $\delta(q_4,1) = q_4$
$\delta(q_1,0) = q_4$ $\delta(q_1,1) = q_2$ } $q_1 \neq q_4$

$\delta(q_4,0) = q_4$ $\delta(q_4,1) = q_4$
$\delta(q_3,0) = q_0$ $\delta(q_3,1) = q_4$ } $q_3 \neq q_4$

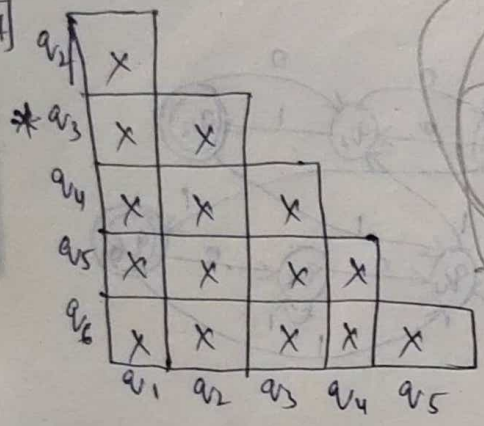
$\delta(q_5,0) = q_2 \quad \delta(q_5,1) = q_4$

$\delta(q_3,0) = q_0 \quad \delta(q_3,1) = q_4$

$\left.\begin{array}{l}\delta(q_5,0) = q_2 \quad \delta(q_5,1) = q_4 \\ \delta(q_4,0) = q_4 \quad \delta(q_4,1) = q_4\end{array}\right\} q_5 \neq q_4$



7]



$\left.\begin{array}{l}\delta(q_2,0) = q_1 \quad \delta(q_2,1) = q_3 \\ \delta(q_1,0) = q_2 \quad \delta(q_1,1) = q_6\end{array}\right\} q_1 \neq q_2$

$\left.\begin{array}{l}\delta(q_1,0) = q_2 \quad \delta(q_1,1) = q_6 \\ \delta(q_4,0) = q_4 \quad \delta(q_4,1) = q_2\end{array}\right\} q_1 \neq q_4$

$\left.\begin{array}{l}\delta(q_4,0) = q_4 \quad \delta(q_4,1) = q_2 \\ \delta(q_2,0) = q_1 \quad \delta(q_2,1) = q_3\end{array}\right\} q_4 \neq q_2$

$\left.\begin{array}{l}\delta(q_1,0) = q_2 \quad \delta(q_1,1) = q_6 \\ \delta(q_5,0) = q_4 \quad \delta(q_5,1) = q_5\end{array}\right\} q_1 \neq q_5$

$\left.\begin{array}{l}\delta(q_5,0) = q_4 \quad \delta(q_5,1) = q_5 \\ \delta(q_2,0) = q_1 \quad \delta(q_2,1) = q_3\end{array}\right\} q_3 \neq q_5$

$\left.\begin{array}{l}\delta(q_5,0) = q_4 \quad \delta(q_5,1) = q_5 \\ \delta(q_4,0) = q_4 \quad \delta(q_4,1) = q_2\end{array}\right\} q_5 \neq q_4$

$\left.\begin{array}{l}\delta(q_3,0) = q_2 \quad \delta(q_3,1) = q_4 \\ \delta(q_6,0) = q_5 \quad \delta(q_6,1) = q_4\end{array}\right\} q_3 \neq q_6$

8]

	q_0	q_1	q_2	q_3	q_4	q_5	q_6
q_1	X						
q_2	X	X					
q_3	X	X	X				
q_4	✓	X	X	X			
q_5	X	X	X	✓	X		
q_6	X	X	X	X	X	X	
q_7	X	✓	X	X	X	X	X

$\delta(q_1,0) = q_6$ $\delta(q_1,1) = q_2$
$\delta(q_0,0) = q_1$ $\delta(q_0,1) = q_5$ } $q_1 \neq q_0$

$\delta(q_3,0) = q_2$ $\delta(q_3,1) = q_6$
$\delta(q_0,0) = q_1$ $\delta(q_0,1) = q_5$ } $q_3 \neq q_0$

$\delta(q_3,0) = q_2$ $\delta(q_3,1) = q_6$
$\delta(q_1,0) = q_6$ $\delta(q_1,1) = q_2$ } $q_1 \neq q_3$

$\delta(q_4,0) = q_7$ $\delta(q_4,1) = q_5$
$\delta(q_0,0) = q_1$ $\delta(q_0,1) = q_5$ } $q_0 = q_4$

$\delta(q_4,0) = q_7$ $\delta(q_4,1) = q_5$
$\delta(q_3,0) = q_2$ $\delta(q_3,1) = q_6$ } $q_4 \neq q_1$

$\delta(q_4,0) = q_7$ $\delta(q_4,1) = q_5$
$\delta(q_3,0) = q_2$ $\delta(q_3,1) = q_6$ } $q_4 \neq q_3$

$\delta(q_0,0) = q_1$ $\delta(q_0,1) = q_5$
$\delta(q_5,0) = q_2$ $\delta(q_5,1) = q_6$ } $q_0 \neq q_5$

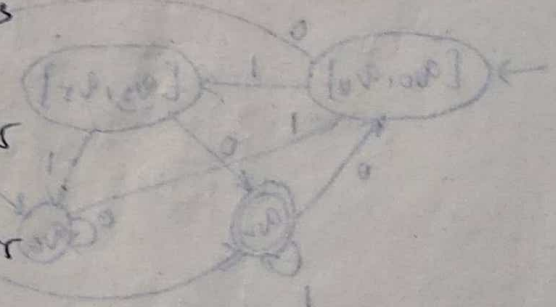
$\delta(q_5,0) = q_2$ $\delta(q_5,1) = q_6$
$\delta(q_1,0) = q_6$ $\delta(q_1,1) = q_2$ } $q_1 \neq q_5$

$\delta(q_5,0) = q_2$ $\delta(q_5,1) = q_6$
$\delta(q_3,0) = q_2$ $\delta(q_3,1) = q_6$ } $q_3 = q_5$

$\delta(q_5,0) = q_2$ $\delta(q_5,1) = q_6$
$\delta(q_4,0) = q_1$ $\delta(q_4,1) = q_5$ } $q_5 \neq q_4$

$\delta(q_6,0) = q_6$ $\delta(q_6,1) = q_4$
$\delta(q_0,0) = q_1$ $\delta(q_0,1) = q_5$ } $q_0 \neq q_6$

$\delta(q_6,0) = q_6$ $\delta(q_6,1) = q_4$
$\delta(q_1,0) = q_6$ $\delta(q_1,1) = q_2$ } $q_1 \neq q_6$



$\delta(q_6,0) = q_6$ $\delta(q_6,1) = q_4$
$\delta(q_3,0) = q_2$ $\delta(q_3,1) = q_6$ } $q_6 \neq q_3$

$\delta(q_6,0) = q_6$ $\delta(q_6,1) = q_4$
$\delta(q_4,0) = q_7$ $\delta(q_4,1) = q_5$ } $q_6 \neq q_4$

$\delta(q_6,0) = q_6$ $\delta(q_6,1) = q_4$
$\delta(q_5,0) = q_2$ $\delta(q_5,1) = q_6$ } $q_6 \neq q_5$

$\delta(q_7,0) = q_6$ $\delta(q_7,1) = q_2$
$\delta(q_0,0) = q_1$ $\delta(q_0,1) = q_5$ } $q_0 \neq q_7$

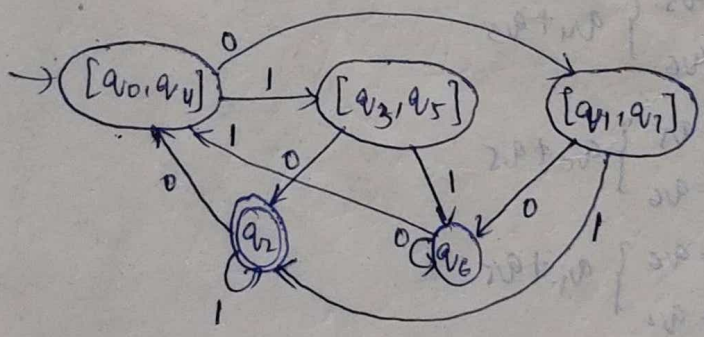
$\delta(q_7,0) = q_6$ $\delta(q_7,1) = q_2$
$\delta(q_1,0) = q_6$ $\delta(q_1,1) = q_2$ } $q_1 = q_7$

$\delta(q_7,0) = q_6$ $\delta(q_7,1) = q_2$
$\delta(q_3,0) = q_2$ $\delta(q_3,1) = q_6$ } $q_7 \neq q_3$

$\delta(q_7,0) = q_6$ $\delta(q_7,1) = q_2$
$\delta(q_4,0) = q_7$ $\delta(q_4,1) = q_5$ } $q_7 \neq q_4$

$\delta(q_7,0) = q_6$ $\delta(q_7,1) = q_2$
$\delta(q_5,0) = q_2$ $\delta(q_5,1) = q_6$ } $q_7 \neq q_5$

$\delta(q_7,0) = q_6$ $\delta(q_7,1) = q_2$
$\delta(q_6,0) = q_6$ $\delta(q_6,1) = q_4$ } $q_7 \neq q_6$



Testing equivalence of regular language :-

Table filling algorithm is used for testing the equivalence of regular language.

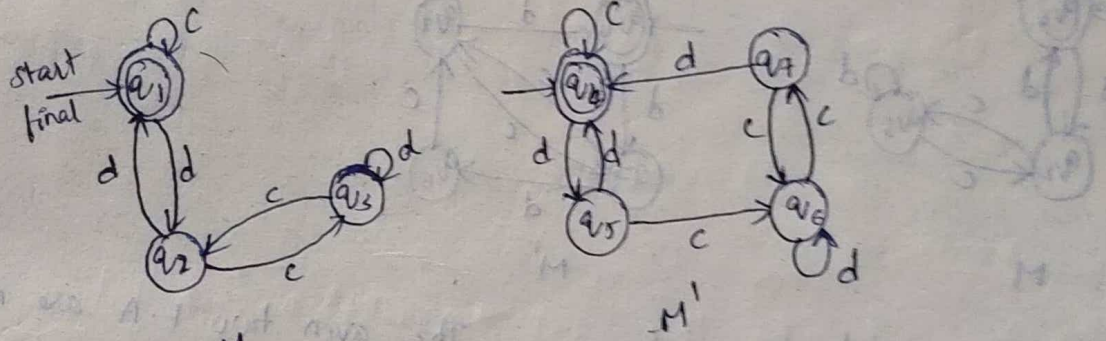
Method for comparing two FA :-

Let M, M' be two FA and Σ is a set of input strings.

(i) we will construct a transition table have pair wise entries (q, q') where $q \in M$ & $q' \in M'$ for each input symbol.

(ii) If we get a pair as one final state and other non-final state, then we can terminate construction of transition table declaring that two FA are not equivalent. Construction of transition table gets terminated when there is no new pair appear in transition table.

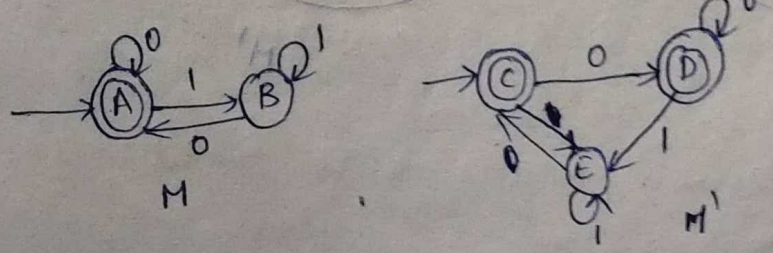
Ex :- 1] Check whether two FA are equivalent or not



	M	
	c	d
→ $\{q_1, q_4\}$	$\{q_1, q_4\}$	$\{q_2, q_5\}$
$\{q_2, q_5\}$	$\{q_3, q_6\}$	$\{q_1, q_4\}$
$\{q_3, q_6\}$	$\{q_2, q_7\}$	$\{q_3, q_6\}$
$\{q_2, q_7\}$	$\{q_3, q_6\}$	$\{q_1, q_4\}$

∴ The given two FA are equivalent because no pair is in combination of final and non-final

2]



	0	1
{A,C}	{A,D}	{B,E}
{A,D}	{A,D}	{B,E}
{B,E}	{A,D}	{B,E}

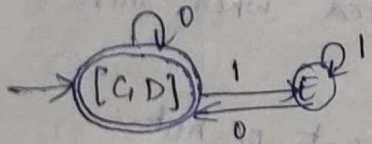
equivalent

Meth II

	C	D
*D	✓	
E	X	X

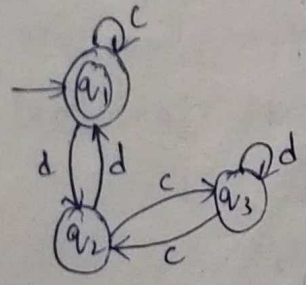
C D
* *

$\delta(C,0) = D \quad \delta(C,1) = E$
$\delta(D,0) = D \quad \delta(D,1) = E$
} C = D

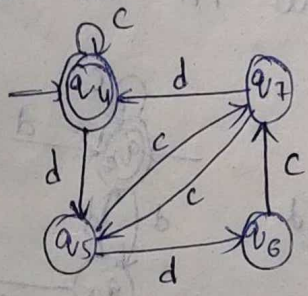


equivalent

3]



M

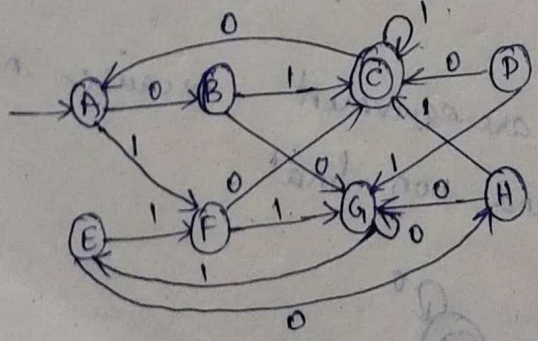


M'

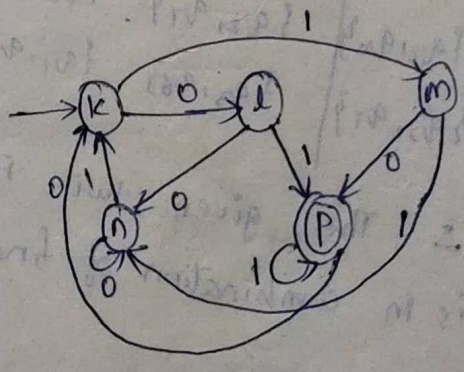
	c	d
→ {q1,q4}	{q1,q4}	{q2,q5}
{q2,q5}	{q3,q7}	{q1,q6}

The given two F.A are not equivalent because is in combination of final and non-final state {q1,q6}.

4]



M



M'

	0	1
$\{A,K\}$	$\{B,L\}$	$\{F,M\}$
$\{B,L\}$	$\{G,N\}$	$\{C,P\}$
$\{F,M\}$	$\{C,P\}$	$\{G,N\}$
$\{G,N\}$	$\{G,N\}$	$\{E,K\}$
$\{C,P\}$	$\{A,K\}$	$\{C,P\}$
$\{E,K\}$	$\{H,L\}$	$\{F,M\}$
$\{A,L\}$	$\{G,N\}$	$\{C,P\}$

equivalent

→	$\{q_0,q_4\}$	$\{q_1,q_5\}$
	$\{q_1,q_5\}$	$\{q_3,q_6\}$
	$\{q_3,q_6\}$	$\{q_2,q_7\}$
	$\{q_2,q_7\}$	$\{q_0,q_{11}\}$
	$\{q_0,q_{11}\}$	$\{q_1,q_{10}\}$
	$\{q_1,q_{10}\}$	$\{q_3,q_9\}$

check whether the following language is regular or not.

1] $L = \{a^i b^{2j} \mid i,j \geq 1\}$ is a RL because i & j are independent

2] $L = \{a^i b^{4j} \mid i,j \geq 1\}$ is a RL because a & b are independent

3] $L = \{a^n \mid n \text{ is even}\}$ is a RL because common difference is maintained throughout the ~~strings~~ series.

4] $L = \{a^n \mid n \text{ is odd}\}$ "

5] $L = \{a^n \mid n \text{ is prime}\}$ is not a RL because there is no comm dif. throughout the series.

6] $\{a^{n^2} / n \geq 1\}$ is not a R.L because "

7] $\{a^{n^3} / n \geq 1\}$ "

8] $\{a^{2^n} / n \geq 1\}$ "

9] $\{a^i b^{i^2} / i,j \geq 1\}$ x even if a & b are independent because CD is not maintained throughout the series for b.

10] $\{a^i b^{2n} \mid a, n \geq 1\}$ ✓

11] $\{a^i b^{2^n} \mid a, n \geq 1\}$ ✗

12] $\{a^n b^m c^k \mid n, m, k \geq 1\}$ ✓

13] $\{a^n b^{m+k} c^k \mid n, m, k \geq 1\}$ ✗

14] $\{a^m b^n \mid n \geq 1, m > n\}$ ✗

*15] $\{a^n b^n \mid n \geq 1\}$ ✗

*16] $\{a^n b^n \mid n \leq 1\}$ ✗ ✓ limits are limited to catain number

17] $\{a^n b^n / n \leq 10^{10^{10^{10}}}\}$ ✓

18] $\{a^n b^n \mid n \geq 10^{10^{10}}\}$ ✗

*19] $\{ww^r / w \in (a,b)^*\}$ ✗

*20] $\{ww^r / |w| = 2, w \in (a,b)^*\}$ ✓

*21] $\{a^i b^j c^k / i < j+k, j, k \geq 1\}$ ✗

*22] $\{a^n / n$ is a perfect square $n \geq 1\}$ ✗

*23] $\{a^n / n$ is perfect square, $n \leq 10\}$ ✓

Pumping lemma :-

Stmt :- let 'L' be RL then exist a constant n such that for every string w in 'L' such that $|w| \geq n$
we can break w into three parts i.e, $w = xyz$

such that

(i) $y \neq \varepsilon$ or $\emptyset$

(ii) $|xy| \leq n$

(iii) for $k \geq 0$ the string $xy^k z$ is in L

Proof :- Suppose L is regular then $L = L(A)$ for some DFA, A and suppose A have n states

Now, consider any string w of length n or more say

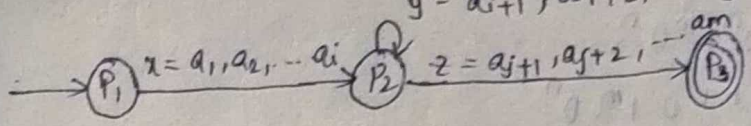
$w = a_1, a_2, a_3 \ldots a_m$ where $m \geq n$

each a_i is an input symbol we can break $w = xyz$ as follows

i. $y = a_1, a_2, \ldots a_i$

2. $y = a_{i+1}, a_{i+2}, \cdots a_j$

3. $z = a_{j+1}, a_{j+2}, \cdots a_m$



Now, the automata A received the input xy^kz for $k \geq 0$

If $k=0$, the state P_1 takes x and goes to P_2 and P_2 takes z and goes to P_3

i.e, The automata A accepts xz

if $k>0$, the automata A goes from P_1 to P_2 on input x, circles from P_2 to P_i for some k times on input y^k and then goes to accepting state and input z. then for any $k \geq 0$ xy^kz is accepted by A

1] PT the language $L = \{0^n 1^{2n} \mid n \geq 1\}$ is not regular.

Sol

Suppose the Language L is regular; then acc. to pumping lemma, $\exists$ an integer n such that for every string w

$|w| \geq n$

Choose the string w from the given language L and split the string w into three parts such as $w = xyz$ in which $|y| \neq 0$ and $|xy| \leq n$

Consider the string $w = 0^{n-1} 0 1^{2n}$

let $x = 0^{n-1}$, $y = 0$, $z = 1^{2n}$

i.e, $w = xyz$

pump y for 0 times, then string w will be $w = 0^{n-1} \cdot 1^{2n}$, $n \geq 1$

Here $n \neq n-1$, $n \geq 1$

So, by contradiction, given language is not regular.

2] Prove the language $L = \{0^n 1^m 0^n \mid m, n \geq 1\}$

Suppose, L is regular; then acc. to pumping lemma, $\exists$ ana integer n such that $\forall$ string w $|w| \geq n$

Choose the string w from given language L and split the w into 3 parts $w = xyz$ in which $|y| \neq 0$ and $|xy| \leq n$.

Consider the string $w = 0^n 1^m 0^n$

$= 0^{n-1} \cdot 0 \cdot 1^m \cdot 0^n$

let $x = 0^{n-1}$, $y = 0$, $z = 1^m 0^n$

pump y for 2 times

$w = 0^{n-1} 0^2 1^m 0^n$

$= 0^{n+1} 1^m 0^n$

$n+1 \neq n$

$\therefore$ L is not regular.

3] PT $L = \{1^n$ such that n is prime$\}$

Suppose L is regular language for any integer n / $w \geq n$

Choose the string w to be $w = 1^p$ where p is prime number and split w into 3 parts i.e, $w = xyz$ in which $|y| > 0$ and $|xy| \leq n$

i.e, Consider $w = xyz$ i.e, $|y| = m$ then $|xz| = p - m$

$w = xyz$

then $|w| = |xz| + |y|$

pump 'y' for 'p-m' times

$|w| = |xz| = |y|^{p-m}$

$= (p-m) + (p-m)|y|$

$= (p-m) + (p-m)m$

$= (p-m)[1+m]$

Given language is not regular because the string has two factors

4] P.T $L = \{0^n / n$ is perfect square$\}$ is not regular

Suppose L is regular for any integer n / $w \geq n$. choose the string w to be w from given language L and split the w into three parts $w = xyz$ in which $|y| \neq 0$ & $|xy| \leq n$

Choose $w = 0^s$ where s is perfect square

$|w| = s = k^2$, $k \geq 1$

split the string 'w' into three parts

let $|y| = m$, $|xz| = s - m = k^2 - m$

pump y for 2 times

$|xy^2z| = |xz| + |y^2| = k^2 - m + 2|y| = k^2 - m + 2m = k^2 + m$

$m \geq 1$

$k^2 + m$ is never a perfect square

So, the given L is not regular

4] Check whether $L = \{0^n | n$ is perfect cube $\}$ is regular or not

Suppose L is regular $\forall$ n $|w| \geq n$

Choose w to be $w = 0^c$ where c is perfect cube

$|w| = c \geq n^3$

$w = xyz$ $|y| \neq 0$ $|xz| \leq n$

let $|y| = m$, $|xz| = n^3 - m$

pump y for 2 times

$|xy^2z| = |xz| + |y^2|$

$= n^3 - m + 2m = n^3 + m$

$n^3 + m$ is never a perfect cube

So, L is not regular

3] Check whether $L = \{0^n | n$ is power of 2 $\}$ is regular or not

$|w| = c = 2^n$

let $|y| = m$, $|xz| = 2^n - m$

pump y for 2 times

$|xy^2z| = |xz| + 2|y|$

$= (2^n - m) + 2m$

$= 2^n + m$

$2^n + m$ is never a power of 2 So, not regular

Regular language
Closure properties of pumping lemma:

1. Union of two RL is regular
2. Intersection of two RL is regular
3. Compliment of a RL is regular
4. Difference of two RL is regular
5. Reverse of RL is regular
6. Closure operation on RL is regular
7. Homomorphism of a RL is regular
8. Inverse Homomorphism of a RL is regular

Theorem 1:- PT if L_1 & L_2 are two RL, then $L_1 \cup L_2$ is also regular.

Proof:- If L_1 & L_2 are regular then they have RL i.e, $L_1 = L(R_1)$ and $L_2 = L(R_2)$

$L_1 \cup L_2$ can be represented as $L(R_1) \cup L(R_2) = L(R_1 + R_2)$

$\therefore L_1 \cup L_2$ is regular.

$\therefore$ any language given by RE is regular

Theorem 2:- PT compliment of regular language is regular

Proof:-

Consider L_1 be RL accepted by some DFA, $M = (Q, \Sigma, \delta, q_0, F)$

The compliment of RL L_1 is denoted by $\bar{L_1}$ which is accepted by some DFA, $M' = (Q', \Sigma', \delta', q_0', F')$

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

i.e, M is DFA with final states $\in F$ and M is DFA in which all non-final states of M becomes final.

The strings that are accepted by M are rejected by M'

Theorem 3:- Prove if L_1 and L_2 are regular, then $L_1 \cap L_2$ is regular

Proof:- let L_1, L_2 be two regular sets

Compliment of L_1 and L_2 is denoted by $\bar{L_1}$ and $\bar{L_2}$ which are also

regular sets.

$\bar{L_1} \cup \bar{L_2}$ is also regular

$\overline{(\bar{L_1} \cup \bar{L_2})}$ is also regular

i.e, $\overline{(\bar{L_1} \cup \bar{L_2})} = \bar{\bar{L_1}} \cap \bar{\bar{L_2}} = L_1 \cap L_2$ is a regular

Theorem 4 :- Prove difference of two RL is regular.

Let L_1 & L_2 be two Rsets.

$\because L_2$ is regular, $\bar{L_2}$ is also regular.

then $L_1 \cup \bar{L_2}$ is also regular

and $L_1 \cap \bar{L_2}$ is also regular

i.e, $L_1 \cap \bar{L_2} = L_1 - L_2$ is also regular.

Arden's theorem :-

let P and Q be two regular expressions over input set Σ the RE, R is given as $R = Q + RP$ which has unique solution

$R = QP^*$

Proof :-

Let P and Q are two RE over input set Σ

if P does not contain ε, the Regular expression $R = Q + RP \rightarrow (1)$

Replace R by QP^* in right side of (1)

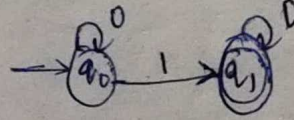
(1) $\Rightarrow$ $R = Q + QP^*P$

$= Q(\varepsilon + P^*P)$

$R = QP^*$

Hence it is proved $R = QP^*$ is unique solution for $R = Q + RP$

Convert the following automata to RE by Arden's theorem



$q_0 = \varepsilon + q_0 0 \rightarrow (1)$ [ε is added since q_0 is start state]

$q_1 = q_0 1 + q_1 0 \rightarrow (2)$

(1) $\Rightarrow$ $q_0 = \varepsilon + q_0$ which is in form of $R = Q + RP$

$R = QP^*$

$q_0 = \varepsilon \cdot 0^* = 0^*$

$q_0 = 0^*$

Sub q_0 in ②

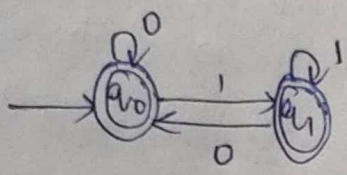
② $\Rightarrow$ $q_1 = q_0 1 + q_1 0$

$\Rightarrow q_1 = 0^* 1 + q_1 0$

$q_1 = 0^* 10^*$

R.E of Given FA is $0^* 10^*$

2]



$q_0 = \varepsilon + q_0 0 + q_1 0 \longrightarrow ①$

$q_1 = q_0 1 + q_1 1 \longrightarrow ②$

① $\Rightarrow$ $q_0 = \varepsilon + q_1 0 + q_0 0$

$= (\varepsilon + q_1 0) 0^* \longrightarrow ③$

$q_1 = (\varepsilon + q_1 0) 0^* 1 + q_1 1$

$= 0^* 1 + q_1 00^* 1 + q_1 1$

$= 0^* 1 + q_1 (00^* 1 + 1)$

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

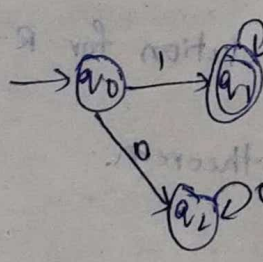
③ $\Rightarrow$ $q_0 = (\varepsilon + 0^* 1 (00^* 1 + 1)^*) 0^*$

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

~~$= 0^* + (00^* + 1)^*$~~ $= (\varepsilon + 0^* 1 (00^* 1 + 1)^* 0) 0^*$

R.E $= (\varepsilon + 0^* 1 (00^* 1 + 1)^* 0) 0^* + 0^* 1 (00^* 1 + 1)^*$

3]



$q_0 = \varepsilon +$

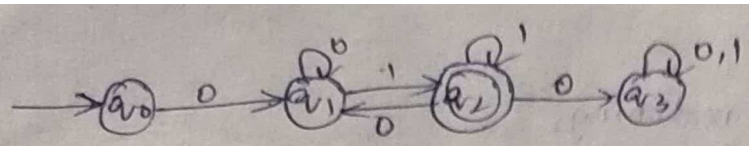
$q_1 = q_0 1 + q_1 0 + q_1 1$

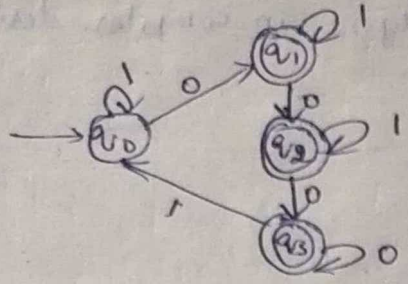
$= 1 + q_1 (0 + 1)$

$= 1 (0 + 1)^*$

R.E is $1(0+1)^*$

Karthika 8-2-16 Monday

4] 

5] 

4] $q_0 = \varepsilon$

$q_1 = q_0 0 + q_1 0 + q_2 0$

$q_2 = q_1 1 + q_2 1$

$q_3 = q_2 0 + q_3 (0+1)$

$q_1 = 0 + q_2 0 1 + q_1 0$

$= (0 + q_2 0) 0^*$

$q_2 = (0 + q_2 0) 0^* 1 + q_2 1$

$= 00^*1 + q_2 00^*1 + q_2 1$

$= 00^*1 + q_2 (00^*1 + 1)$

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

5] $q_0 = \varepsilon + q_0 1$

$q_1 = q_0 0 + q_1 1$

$q_2 = q_1 0 + q_2 1$

$q_3 = q_2 0 + q_3 0$

$q_0 = \varepsilon + q_0 1 = \varepsilon 1^* = 1^*$

$q_1 = 1^* 0 + q_1 1$

$= 1^* 0 1^*$

$q_2 = 1^* 0 1^* 0 + q_2 1$

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

$q_3 = 1^* 0 1^* 0 1^* 0 + q_3 0$

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

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

Applications of RE :-

1. text editors for pattern matching
2. used as lexical analyzer for lexical analysis in compiler design

Context Free Grammar

Def:- Context free Grammar can be formally defined a set $G = (V, T, P, S)$ where V, T are set of terminals and non-terminals respectively. → P is set of production rules whereas each production rule is in form of

non terminal → terminal

(or)

nonterminal → nonterminal

and finally S represents start symbol

Ex:-

$E \rightarrow E+T \mid T$

$T \rightarrow T * F \mid F$

$F \rightarrow (E) \mid id$

From the above grammar

$S = \{E\}$

$V = \{+, *, (,), id\}$

$T = \{E, T, F\}$

$P = \{E \rightarrow E+T \mid T$

$T \rightarrow T * F \mid F$

$F \rightarrow (E) \mid id\}$

Note:- In general, non terminals are denoted by capital letters and terminals are denoted by lower case letters.

$S \rightarrow aA \mid bB$

$A \rightarrow a \mid \varepsilon$

$B \rightarrow b \mid \varepsilon$

$T = \{S, A, B\}$

$V = \{a, b, \varepsilon\}$

$S = \{S\}$

Construct context free grammar for following language

1] $L = \{\varepsilon, a, aa, aaa, \ldots\}$

$RE = a^*$

$S \rightarrow aS \mid \varepsilon$

2] $L = \{a, b\}$

$S \rightarrow a \mid b$

3] $L = \{aa, ab, ba, bb\}$

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

$S \rightarrow AA$

$A \rightarrow a \mid b$

4] L = any no. of a's and any no of b's

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

$S \to aS \mid bS \mid \varepsilon$

(or) $S \to AS \mid \varepsilon$

$A \to a \mid b$

5] $(a+b+c)^*$

$S \to aS \mid bS \mid cS \mid \varepsilon$

6] starts with a and ends with b $\Sigma = \{a, b\}$

$a(a+b)^*b$

$V = \{a, b, \varepsilon\}$

$T = \{S, A\}$

$S = \{S\}$

$P = \{ = \}$

$S \to aAb$

$A \to aA \mid bA \mid \varepsilon$

7] atleast two a's

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

$S \to AaAaA$

$A \to aA \mid bA \mid \varepsilon$

8] atleast one occurence of 000 $\Sigma = \{0, 1\}$

$(0+1)^*000(0+1)^*$

$S \to A000A$

$A \to 0A \mid 1A \mid \varepsilon$

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

$S \to AB^*$

$A \to a \mid b$

$B \to aB \mid bB \mid \varepsilon$

10] $[(a+b)(a+b)]^*$

$S \to BS \mid \varepsilon$

$B \to AA$

$A \to a \mid b$

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

$S \to B \mid C$

$B \to aAb$

$A \to aA \mid bA \mid \varepsilon$

$C \to bAa$

12] $[(a+b)^*(a+b)]^*$

$S \to CS \mid \varepsilon$

$C \to AB$

$A \to aA \mid bA \mid \varepsilon$

$B \to a \mid b$

13] $L = \{a^n \mid n \geq 0\}$

$S \to aS \mid \varepsilon$

14] $L = \{a^n \mid n \geq 1\}$

$S \to aS \mid a$

15] $L = \{a^n \mid n \geq 2\}$

$S \to$ ~~$aaS \mid aa$~~ $aS \mid aa$

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

~~$S \rightarrow aS \mid bS \mid \varepsilon$~~

$S \rightarrow AB$
$A \rightarrow aA \mid \varepsilon$
$B \rightarrow bB \mid \varepsilon$

17] $L = \{a^n b^m \mid n \geq 1, m \geq 0\}$

$S \rightarrow AB$
$A \rightarrow aA \mid a$
$B \rightarrow bB \mid \varepsilon$

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

$S \rightarrow AB$
$A \rightarrow aA \mid a$
$B \rightarrow bB \mid b$

19] $L = \{a^n b^m \mid n \geq 0, m \geq 1\}$

$S \rightarrow AB$
$A \rightarrow aA \mid \varepsilon$
$B \rightarrow bB \mid b$

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

$S \rightarrow ABC$
$A \rightarrow aA \mid a$
$B \rightarrow bB \mid b$
$C \rightarrow cC \mid c$

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

$S \rightarrow ABCD$
$A \rightarrow aA \mid a$
$B \rightarrow bB \mid b$
$C \rightarrow cC \mid c$
$D \rightarrow dD \mid d$

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

$S \rightarrow ABCD$
$A \rightarrow aA \mid a$
$B \rightarrow bB \mid b$
$C \rightarrow cC \mid c$
$D \rightarrow dD \mid d$

23] accepts all pallindromes over $\Sigma = \{a, b\}$

$S \rightarrow asa \mid bsb \mid a \mid b \mid \varepsilon$

24] accepts all strings that are pallindro of length even.

$S \rightarrow asa \mid bsb \mid \varepsilon$

25] odd pallindrome

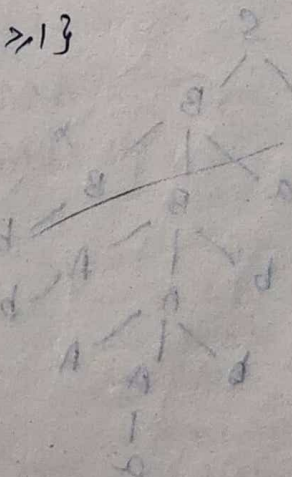
$S \rightarrow asa \mid bsb \mid a \mid b$

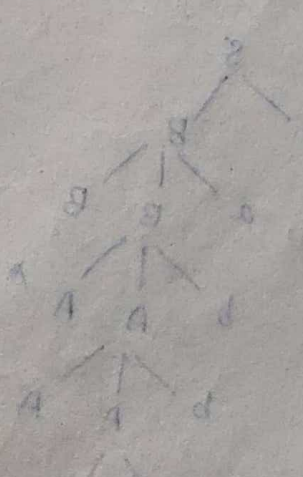
26] $L = \{w \in w^T \mid w \in (a+b)^*$

~~$S \rightarrow ASB \mid bSb \mid \varepsilon$~~
~~$A \rightarrow aA \mid bA$~~
~~$B \rightarrow$~~

$S \rightarrow asa \mid bsb \mid c$

~~27]~~





Derivation tree :-

It is a graphical representation for derivation of given production groups for given CFG.

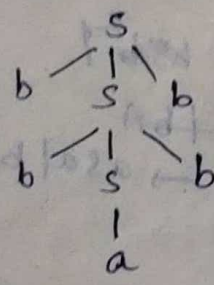
It is the simple way to show, how the derivation can be done to obtain some string from given set of production groups. It is also called as Parse tree.

The following are properties of derivation tree :-

1. Root node is always node indicating start symbol.
2. Derivation is read from left to right
3. leaf nodes are always terminal nodes.
4. Interior nodes are always non-terminal nodes

1] Derive a string bbabb for CFG given below

$S \rightarrow bSb \mid a \mid b$



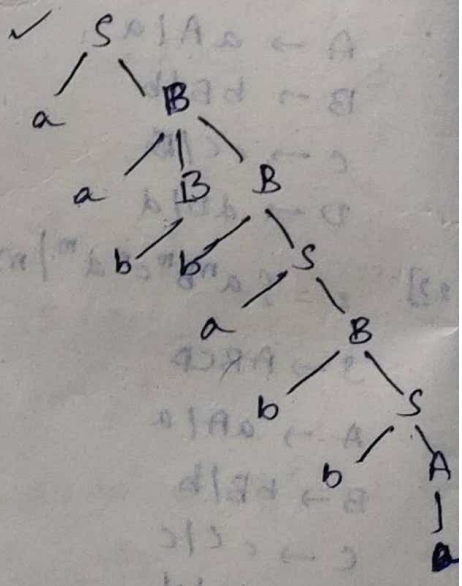
Parse tree

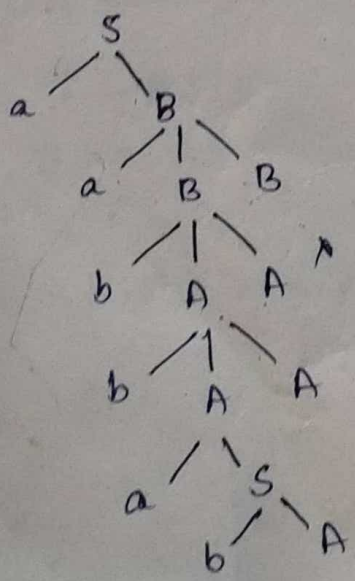
2] Construct DT for aabbabba for

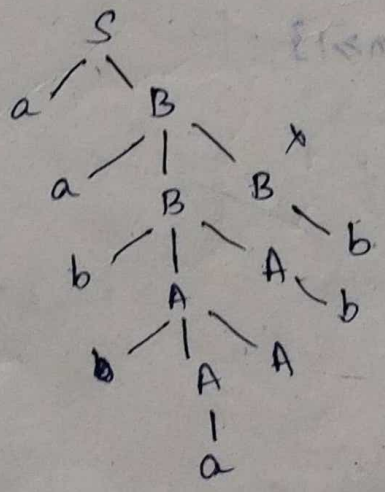
$S \rightarrow aB \mid bA$

$A \rightarrow a \mid aS \mid bAA$

$B \rightarrow b \mid bS \mid aBB$







left and right most derivation:-

The derivation in which left most non-terminal replaced from sentential form first is called LMD

The derivation in which right most non-terminal replaced from sentential form first is called RMD.

1] Derive LHD & RMD for following grammar.

"aba" $S \to XYX$
$X \to a$
$Y \to b$

LMD

$S \to XYX$
$\to aYX$
$\to abX$
$\to aba$

RMD

$S \to XYX$
$\to XYa$
$\to Xba$
$\to aba$

2] string 1000111 grammar

$S \to TOOT$
$T \to 0T \mid 1T \mid \varepsilon$

LMD

$S \to TOOT$
$\to 1OOT$
$\to 1TOOT$ [T → 1T]
$\to 1T0OT$ [T → 0T]
$\to 10\varepsilon OOT$ [T → ε]
$\to 1000IT$ [T → 1T]
$\to 1000 11T$ [T → 1T]
$\to 1000 111T$ [T → 1T]
$\to 1000111\varepsilon$ [T → ε]
$\to 1000111$

RMD

$S \to TOOT$
$S \to TOO1T$
$S \to TOO11T$
$S \to TOO111T$
$S \to TOO111\varepsilon$
$S \to 1TO0111$
$S \to 10TO0111$
$\to 10\varepsilon 00111$
$\to 1000111$

3] babbab ; $S \to AA$
$A \to AAA$
$A \to Ab \mid bA \mid a$

LMD

$S \to AA$
$\to bAA$
$\to baA$
$\to babA$
$\to babbA$
$\to babbAb$

RMD

$S \to AA$
$\to AAb$
$\to Aab$
$\to Abab$
$\to Abbab$
$\to bAbbab$
$\to babbab$

Ambiguous Grammar

A grammar G is said to be ambiguous, if ∃ one or more parse trees for given grammar i.e, there would be more than one LMD or RMD.

Ex:

Consider a grammar

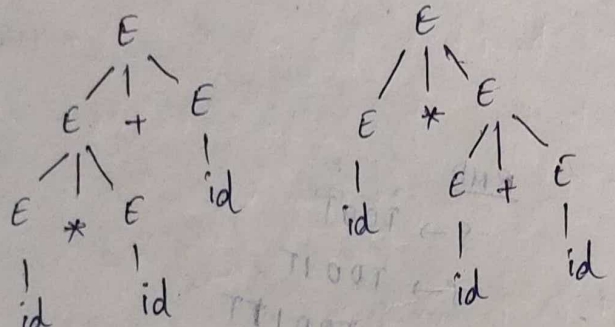
$E \rightarrow E+E$

$E \rightarrow E*E$

$E \rightarrow id$

$V = \{*, +, id\}$, $T = \{\epsilon\}$, $S = \{E\}$

id * id + id



There are two possibilities for given grammar

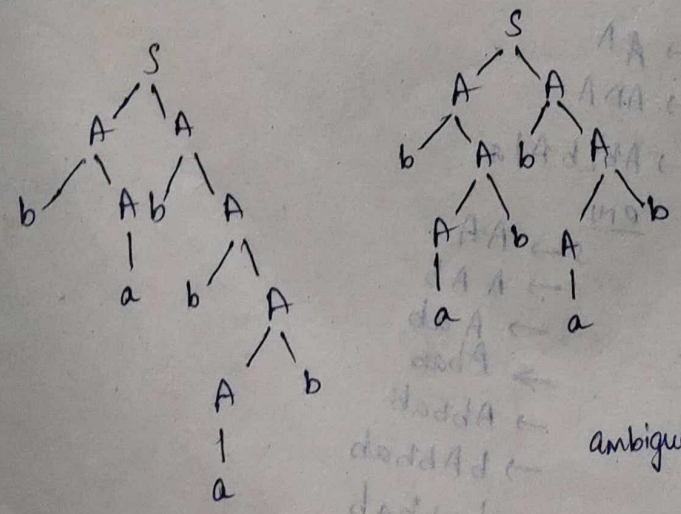
∴ The given grammar is said to be ambiguous.

1] Check whether the grammar is ambiguous or not.

1] $S \rightarrow AA$

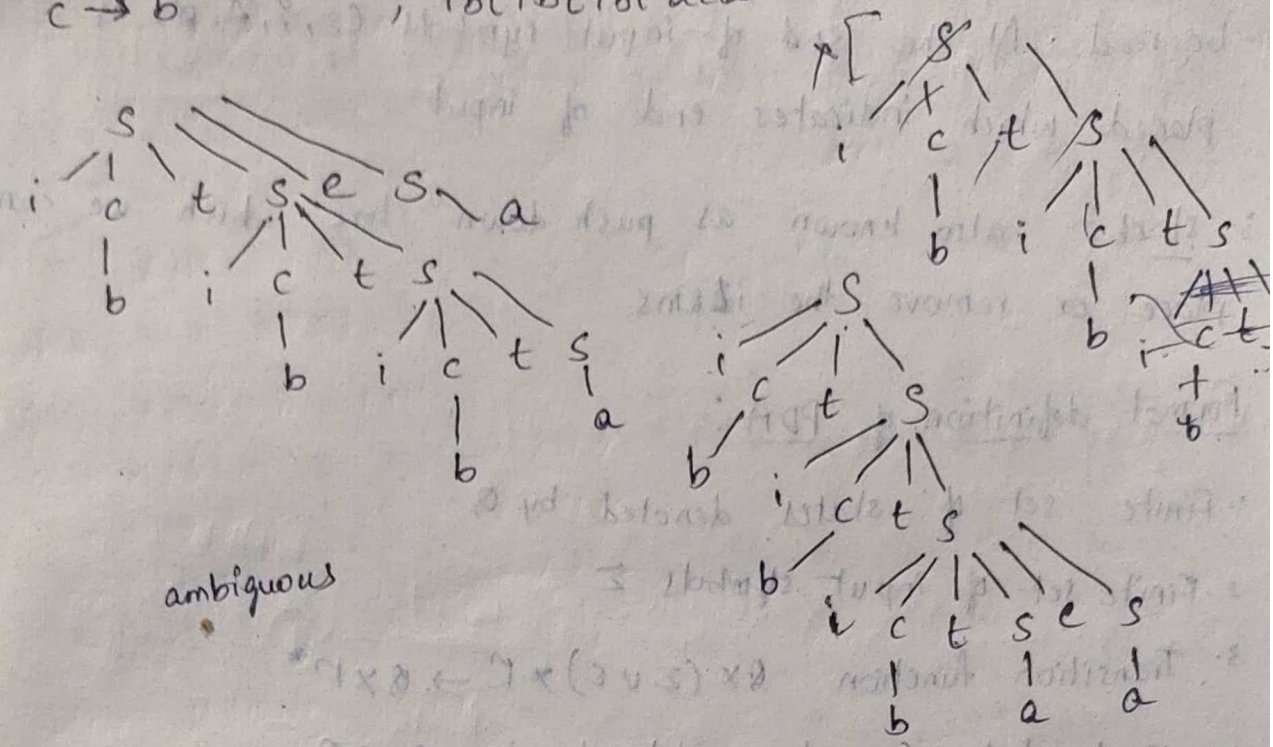
$A \rightarrow AAA$

$A \rightarrow Ab|bA|a$; babbab



ambiguous

s → ictS
s → ictses/a
c → b ; ibtibtibtaca



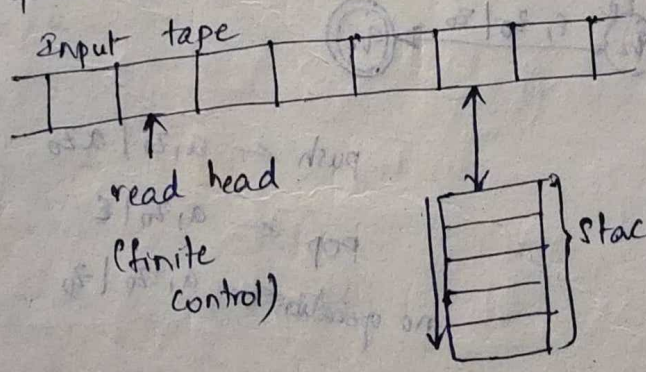
ambiguous

Push Down Automata :-

In PDA, automata is a tool to implement CFL
The biggest drawback of FA is absence of memory.
There is no mechanism to remember the count of inputs in FA
To overcome this issue, PDA is introduced.

Basic structure of PDA :-

PDA will have input tape, read head (finite control) and stack (push down store)



Input tape :- The one in which input string is placed
Input tape is divided into many cells, at each cell one input symbol is placed. Thus, certain input strings placed on table.

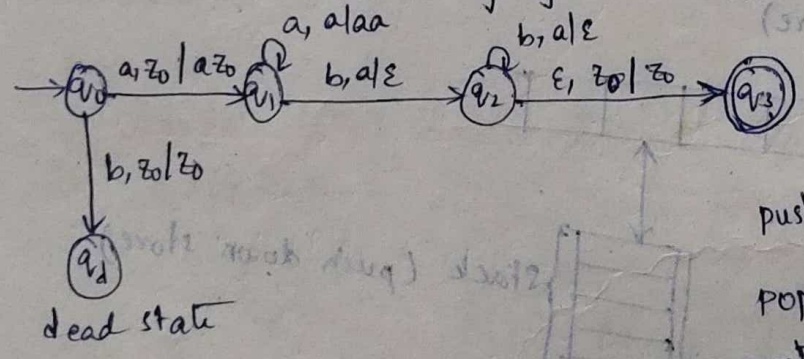
2. Read head :- Read head is also called as finite control has one pointer which points to current symbol which is to be read. At the end of input symbols ($\varepsilon, \$, \Delta, \not\varnothing$) is placed which indicates end of input

3- stack :- also known as push down store which we can place or remove the items

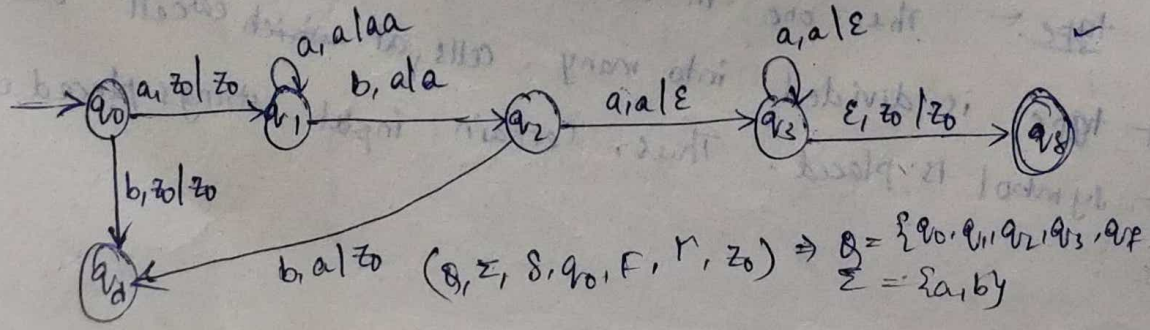
Formal definition of PDA :-

1. Finite set of states denoted by Q
2. Finite set of input symbols Σ
3. Transition function $Q \times (\Sigma \cup \varepsilon) \times \Gamma \to Q \times \Gamma^*$
4. start state 'q_0' of read head i.e., $q_0 \in Q$
5. F is known as set of final states.
6. Γ is stack alphabet ; set of symbols that can be pushed on to stack.
7. z_0 known as stack symbol which is in Γ. It indicates the stack is empty.
8. 7-tuple notation of PDA, $M = (Q, \Sigma, \delta, q_0, F, \Gamma, z_0)$

1] Construct PDA for language $L = \{a^n b^n / n \geq 1\}$



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



3] ~~$a^n b^n a^m$~~

$\delta(q_0, a, z_0) = (q_1, az_0)$

$\delta(q_0, b, z_0) = (q_d, z_0)$

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

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

$\delta(q_2, a, a) = (q_3, \varepsilon)$

$\delta(q_2, b, a) = (q_d, a)$

$\delta(q_3, a, a) = (q_3, \varepsilon)$

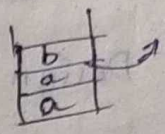
$\delta(q_3, b, z_0) = (q_f, z_0)$

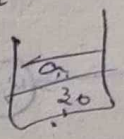
$q_0 = \{q_0\}$

$F = \{q_f\}$

$\Gamma = \{z_0, a\}$

$z_0 = z_0$





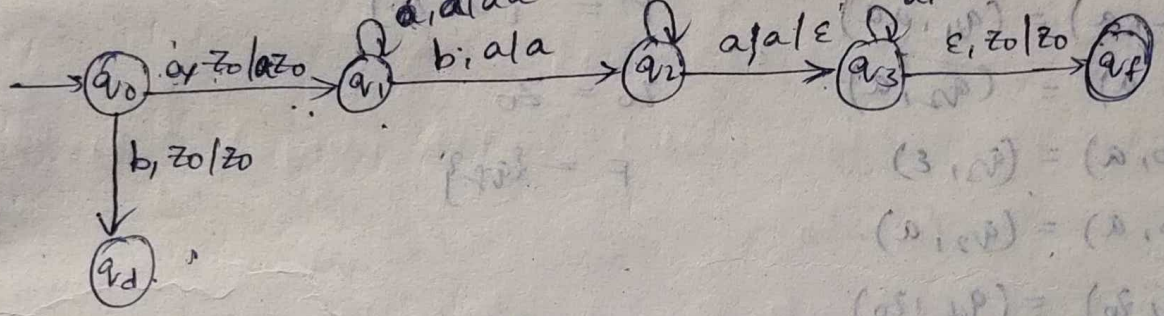
aab aa

aaab aa

a, a/aa

b, a/a

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



$\delta(q_0, a, z_0) = (q_1, az_0)$

$\delta(q_0, b, z_0) = (q_d, z_0)$

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

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

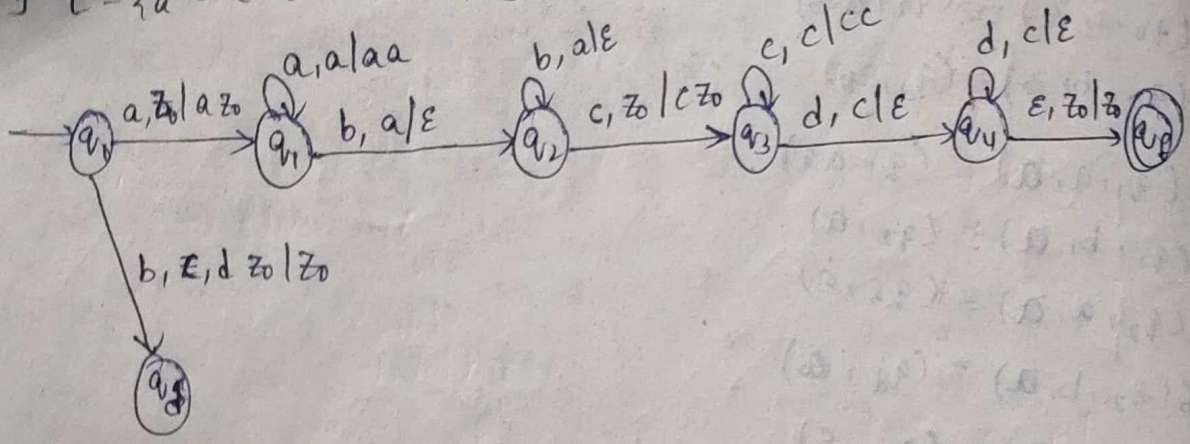
$\delta(q_2, a, a) = (q_3, \varepsilon)$

$\delta(q_3, b, a) = (q_2, a)$

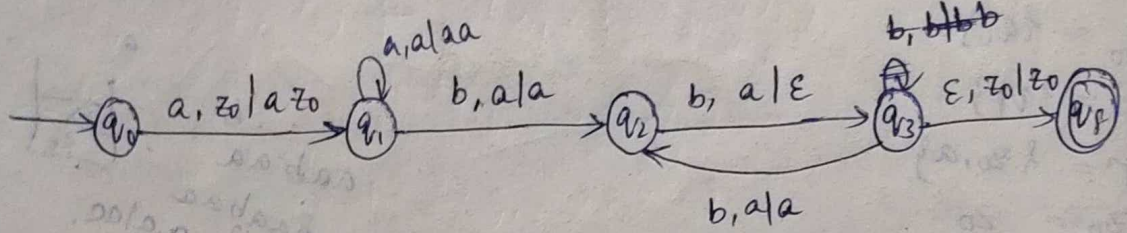
$\delta(q_3, a, a) = (q_3, \varepsilon)$

$\delta(q_3, \varepsilon, z_0) = (q_f, z_0)$

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



5] Design PDA $L = \{a^n b^{2n} \mid n \geq 1\}$



$\delta(q_0, a, z_0) = (q_1, a z_0)$

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

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

$\delta(q_2, b, a) = (q_3, \varepsilon)$

$\delta(q_3, b, a) = (q_2, a)$

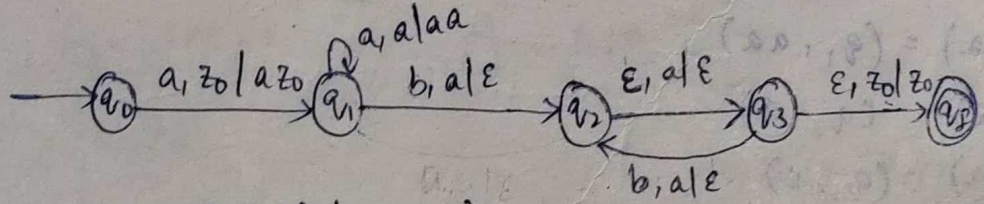
$\delta(q_3, \varepsilon, z_0) = (q_4, z_0)$

$\Gamma = \{a, z_0\}$

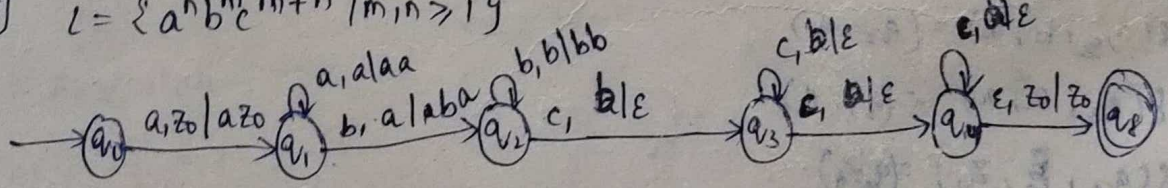
$z_0 = z_0$

$F = \{q_f\}$

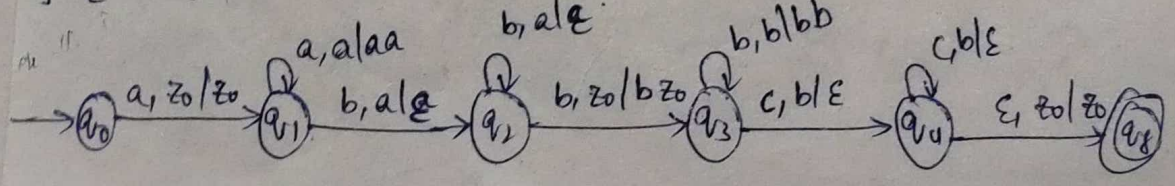
6] $L = \{a^{2n} b^n \mid n \geq 1\}$



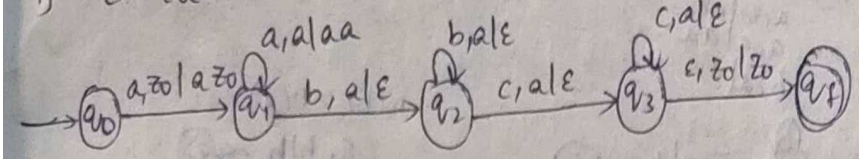
7] $L = \{a^n b^m c^{m+n} \mid m, n \geq 1\}$



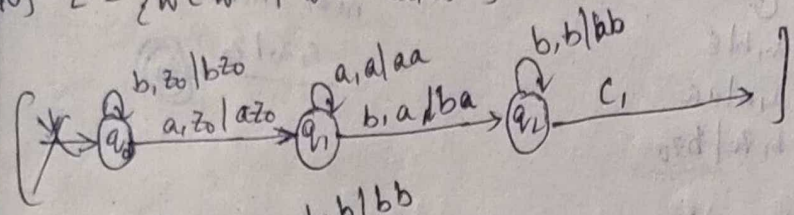
8] $L = \{a^m b^{m+n} c^n\}$

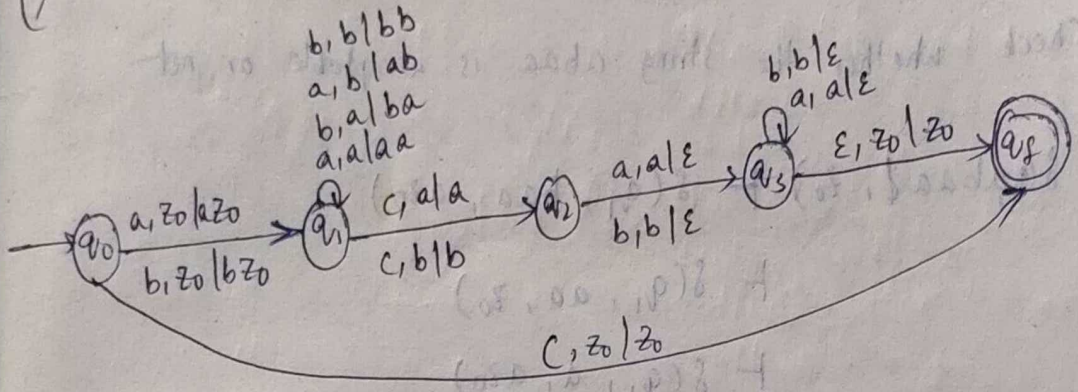


9] $L = \{a^{m+n} b^m c^n \mid m, n \geq 1\}$



10] $L = \{wcw^R \mid w \in (a,b)^*\}$





$\delta(q_0, a, z_0) = (q_1, az_0)$
$\delta(q_0, b, z_0) = (q_1, bz_0)$
$\delta(q_1, a, a) = (q_1, aa)$
$\delta(q_1, b, a) = (q_1, ba)$
$\delta(q_1, a, b) = (q_1, ab)$
$\delta(q_1, b, b) = (q_1, bb)$
$\delta(q_1, c, a) = (q_2, a)$
$\delta(q_1, c, b) = (q_2, b)$
$\delta(q_2, a, a) = (q_3, \varepsilon)$
$\delta(q_2, b, b) = (q_3, \varepsilon)$
$\delta(q_3, a, a) = (q_3, \varepsilon)$
$\delta(q_3, b, b) = (q_3, \varepsilon)$
$\delta(q_3, \varepsilon, z_0) = (q_f, z_0)$

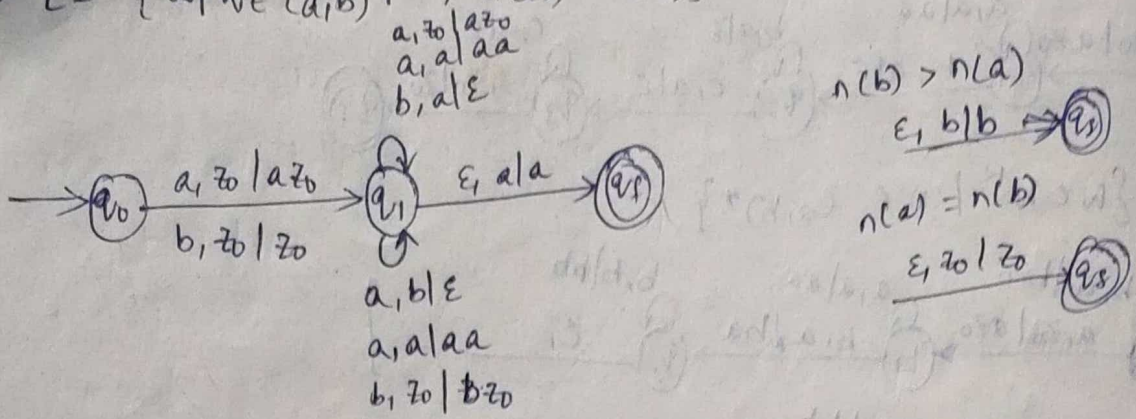
$\mu = \{a, b, z_0\}$
$z_0 = z_0$
$F = \{q_f\}$
$\Sigma = \{a, b\}$

check whether the string abbCbba is acceptable or not

$\delta(q_0, abbCbba, z_0) \vdash \delta(q_1, bbCbba, az_0)$
$\vdash \delta(q_1, bCbba, baz_0)$
$\vdash \delta(q_1, Cbba, bbaz_0)$
$\vdash \delta(q_2, bba, bbaz_0)$
$\vdash \delta(q_3, ba, baz_0) \vdash \delta(q_3, a, az_0)$
$\vdash \delta(q_3, \varepsilon, z_0) \vdash (q_f, z_0)$

acceptable

1] $L = \{w \mid w \in (a,b)^+ , n(a) > n(b)\}$



Check whether the string abaa is acceptable or not

$$\delta(q_0, abaa, z_0) \vdash \delta(q_1, baa, az_0)$$
$$\vdash \delta(q_1, aa, z_0)$$
$$\vdash \delta(q_1, a, az_0)$$
$$\vdash \delta(q_1, \varepsilon, aaz_0) \vdash \delta(q_f, aaz_0)$$

2] $L = \{w \mid w \in (a,b)^*, n(b) > n(a)\}$

The language can be accepted by PDA using two approaches

They are

Acceptance by final state

PDA accepts its input by consuming it and enter final state

Let $P = (Q, \Sigma, \delta, q_0, F, \Gamma, z_0)$ be a PDA, then language accepted by final state is $L(P) = \{w \mid (q_0, w, z_0) \overset{*}{\underset{P}{\vdash}} (q_2, \varepsilon, z_0)\}$

↳ final state

Ex:- Consider

$$\delta(q_0, ww^R, z_0) \vdash^* \delta(q_1, w^R, wz_0)$$
$$\vdash^* \delta(q_1, \varepsilon, z_0)$$
$$\vdash^* (q_f, z_0)$$

↳ final state

Ex:-

$L = \{w \mid n(a) > n(b)\}$

2. Acceptance by Empty stack :-

On reading input string from initial configuration for some PDA, a stack of PDA gets empty.

let $P = (Q, \Sigma, \delta, q_0, F, \Gamma, Z_0)$, then the language accepted by empty stack is $L(P) = \{w \mid (q_0, w, Z_0) \overset{*}{\vdash}_P (q_2, \varepsilon, \varepsilon)\}$

↳ indicates that stack is empty.

nonfinal | final

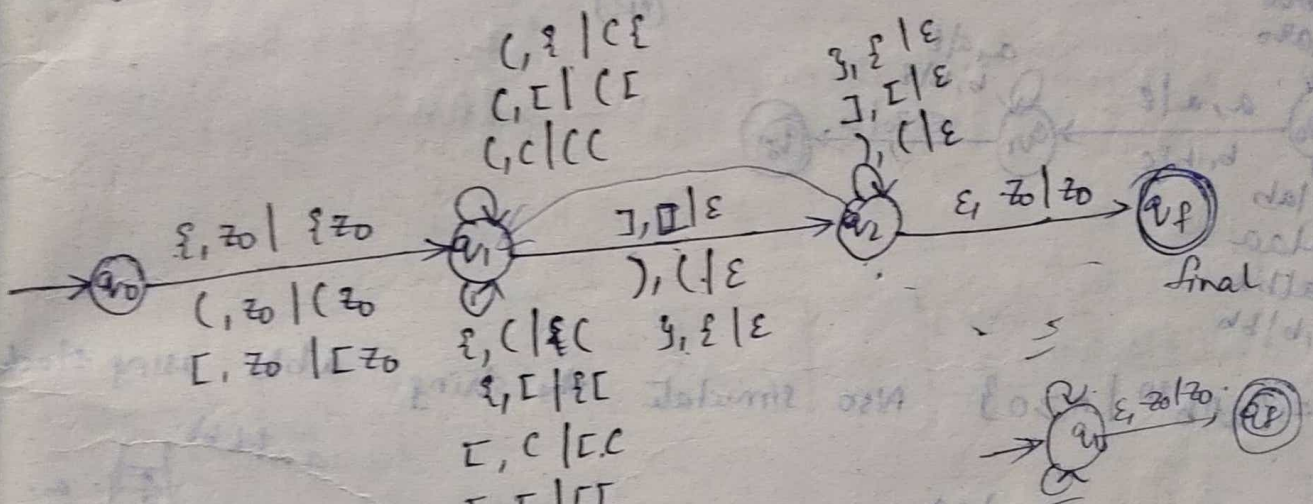
$(q_0, ww^R, Z_0) \overset{*}{\vdash} (q_1, w^R, wZ_0)$

$\overset{*}{\vdash} (q_2, \varepsilon, q_0)$

$\overset{*}{\vdash} (q_2, \varepsilon, \varepsilon)$ ↳ indicates stack is empty.

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

Design PDA which accepts well formed parenthesis.



$\Sigma = \{\{, [, (,),], \}\}$

$\Gamma = \{Z_0, \{, [, (\}$

$q_0 = q_0$

$F = \{q_f\}$

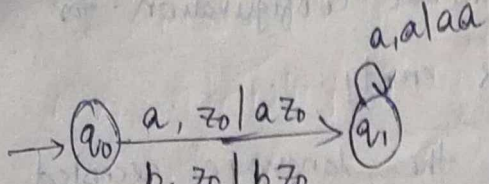
$Z_0 = Z_0$

$\delta(q_0, \{, Z_0) = (q_1, \{Z_0)$

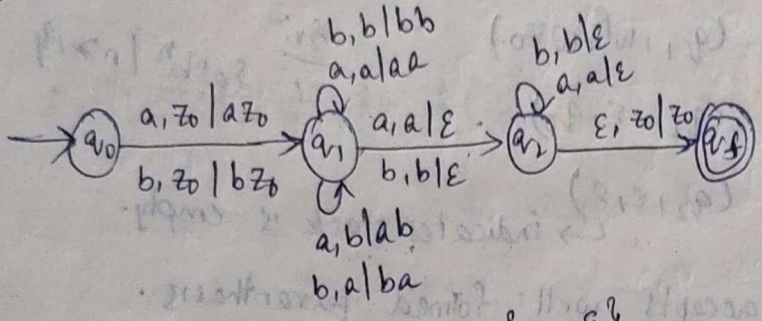
$\delta(q_0, (, Z_0) = (q_1, (Z_0)$

$\delta(q_0, [, Z_0) = (q_1, [Z_0)$

2] Construct PDA which accepts the language $L = \{a^n b^m c^{2n} d^m / m, n \geq 0\}$



3] $L = \{ww^R / w \in (a+b)^+\}$

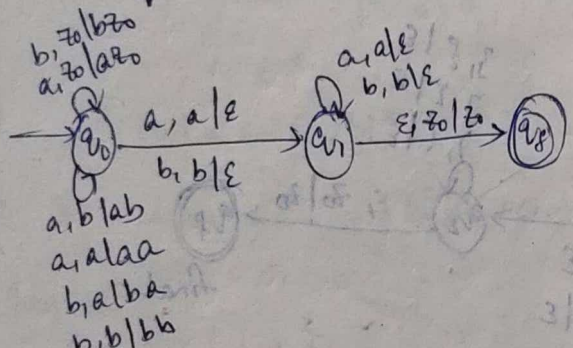


NDPDA

NPDA

$\delta(q_0, a, a) = \{q_1, aa\}, \{q_2, \varepsilon\}$

(OR)



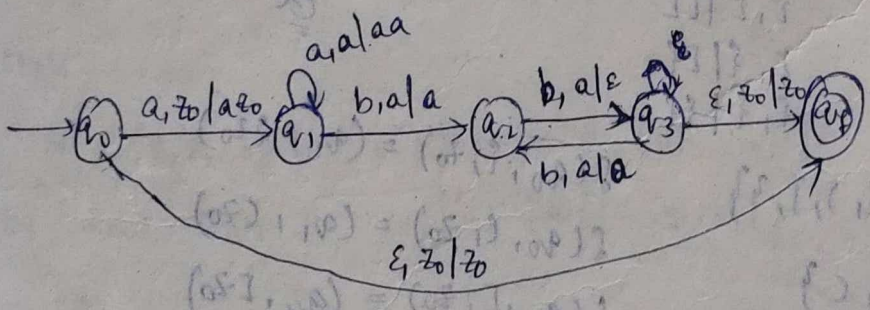
4] $L = \{a^n b^{2n} / n \geq 0\}$ Also simulate the string abb using stack

aabbbb

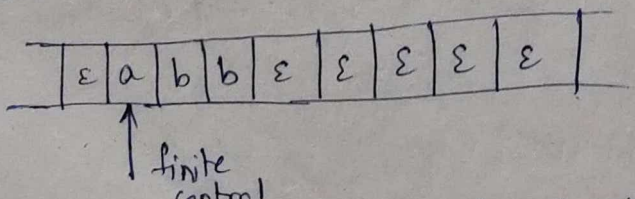
a.

abb

bbb



simulation:-



Current i/p symbol is 'a', top of stack is 'z0', current state is q_0.

Minimisation of CFG :-

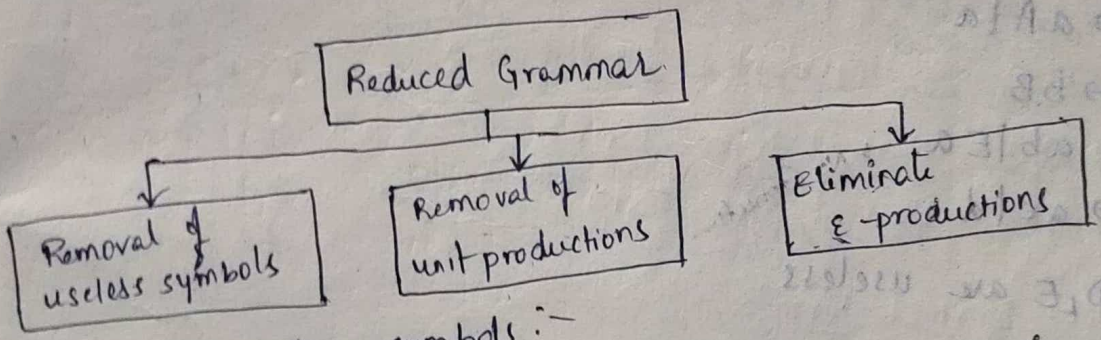
All the grammars are not always optimized i.e, the grammar may consist of some extra symbols (non terminals) having extra symbols, unnecessary increases the length of grammar.

simplification of grammar means reduction of grammar by removing useless symbols.

Properties of reduced grammar are given below :-

1. Each variable (non terminal) and each terminal of G appear in the derivation of some word in L.
2. There should not be any production as $X \rightarrow Y$ where X & Y are non terminals.
3. If ε is not in the language L, then there need not be the production $X \rightarrow \varepsilon$.

The following diagram represents reduced grammar.



Removal of useless symbols :-

Any symbol is useful when it appears on RHS in production group and generate some terminals

If no such derivation exists, then it is supposed to be an useless symbol.

Ex:- Consider CFG

$G = (V, T, P, S)$ where $V = \{S, A, B\}$, $T = \{0, 1\}$ and

$P = \{ S \rightarrow A11B \mid 11A$

$A \rightarrow 11B \mid 0$

$B \rightarrow BB$

$\}$

In the given grammar, there is one useless symbol namely B
So, remove all the productions that involves non-terminal B.

After eliminating useless symbol B, the grammar is

$S \to 11A$
$A \to 0$

Eliminate all useless symbols in following grammar.

1] $S \to AB \mid CA$
$A \to b$
$B \to BC \mid AB$
$C \to aB \mid b$

elimate B
$S \to CA$
$A \to b$
$C \to b$

2] $S \to aA \mid bB$
$A \to aA \mid a$
$B \to bB$
$D \to ab \mid Ea$ → No Connection
$E \to ac \mid d$

B, D, E are useless

$S \to aA$
$A \to aA \mid a$

3] $S \to aA \mid bB \mid cC \mid a$
$A \to aB$
$B \to a \mid Aa$
$C \to cCD$ → No Connection
$D \to ddd$

C, D are useless
$S \to aA \mid bB \mid a$
$A \to aB$
$B \to a \mid Aa$

Elimation of ε-productions in grammar:-

Remove ε production from following grammar

1] $S \to asa \mid bsb \mid \varepsilon$ ✓

after eliminating ε production

$S \to asa \mid bsb \mid aa \mid bb$

2] $S \to XYX$
$X \to 0X \mid \varepsilon$
$Y \to 1Y \mid \varepsilon$

$S \to xyx \mid xy \mid yx \mid y \mid xx \mid x$
$X \to 0x \mid 0$
$Y \to 1Y \mid 1$

3] S → 0S1 | 0B1
S → 0A1
B → ε
A → 01

S → 0S1 | 0B1 | 01
S → 0A1
A → 01

Elimination of unit productions :
Grammar in which one non-terminal gives another non-terminal productions is known as unit production.

X → Y } unit productions
Y → Z
X → aY → not unit production

1] S → 0A | 1B | C
A → 0S | 00
B → 1 | A
C → 01

S → 0A | 1B | 01
A → 0S | 00
B → 1 | 0S | 00

S → C
B → A
are unit productions

2] S → AB
A → a
B → C | b
C → D
D → E | bc
E → d | Ab

S → AB
B → C
C → D
D → E

S → AB
A → a
B → C | b
C → D
D → d | Ab | bc
E → d | Ab

S → AB
A → a
B → d | Ab | bc | b
C → d | Ab | bc
D → d | Ab | bc
E → d | Ab

Minimise the following.

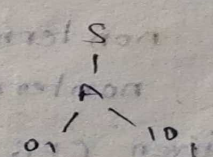
1] S → AB | C
A → 0A | B
B → 1B | C
C → 01

2] S → A | 0C | 1
A → B | 01 | 10
C → ε | CD

S → A | 01
A → 01 | 10

S → 01 | 10 | 01

S → 01 | 10



3] S → 0A | 1B | 01
B → 1B | 01

3] $S \to AB$
$A \to a$
$B \to C|b$
$C \to D$
$D \to E|bc$
$E \to d|Ab$

$S \to AB$
$A \to a$
$B \to d|Ab|bc|b$
use less $C \to d|Ab|bc$
use less $D \to d|Ab|bc$
use less $E \to d|Ab$

$S \to AB$
$A \to a$
$B \to d|Ab|bc|b$

Normal forms :-

A Grammar can be simplified by reducing unit productions, ε-productions and useless symbols.

There is also need to have a grammar in specific form.

~~To normalise such grammar~~

If any grammar is represented in specific form, we need to normalise such grammer i.e, there should be fixed no. of terminals and non-terminals in CFG.

There are two important normal forms

1. Chomsky's Normal Form (CNF)
2. Griebach Normal Form (GNF)

Chomsky's Normal Form :-

It can be defined as

non terminal $\longrightarrow$ non terminal .. non terminal

nonterminal $\longrightarrow$ terminal

Given CFG should be converted into above formula.

Ex:- $X \to YZ$, $Y \to b$, $Z \to c$ $\}$ $\to$ CNF

$X \to XYZ$, $Y \to ab$ $\}$ CNF

. Convert the following CFG to CNF.

$S \to aSa|bSb|a|b$

$S \to aSa$

$S \to bSb$

$S \to a$

$S \to b$

let $A \to a$

$B \to b$

then $S \to ASA$

$S \to BSB$

$S \to a$

$S \to b$

let $P \to SA$

$Q \to SB$

then grammar will be

$S \to AP$

$S \to BQ$

$S \to a$

$S \to b$

$P \to SA$

$Q \to SB$

$A \to a$

$B \to b$

2] $S \to aaaaS$

$S \to aaaa$

let $A \to aa$

$S \to AAS$

$S \to AA$

let $B \to a$

$P \to AS$

then

$S \to AP$

$S \to AA$

$A \to BB$

$P \to AS$

$B \to a$

3] $S \to aAS \mid a$

$A \to SbA \mid SS \mid ba$

$S \to aAS$

$S \to a$

$A \to SbA$

$A \to SS$

$A \to ba$

let ~~$P \to aA$~~

$Q \to Sb$

$P \to a$

$Q \to b$

then

$S \to PAS$

$S \to a$

$A \to SQA$

$A \to SS$

$A \to QP$

let $R \to AS$

$T \to QA$

[$S \to aP$

$S \to a$

$A \to QA$

$A \to SS$

$A \to ba$]

then $S \to PR$

$S \to a$

$A \to ST$

$A \to SS$

$A \to QP$

$P \to a$

$Q \to b$

$R \to AS$

$T \to QA$

4] $S \to ABA$

$A \to aA \mid \varepsilon$

$B \to bB \mid \varepsilon$

In CNF, ε-productions are not allowed.

After eliminating ε-productions, the grammar will be

$S \to ABA \mid B \mid AA \mid BA \mid AB \mid A$

$A \to aA \mid a$

$B \to bB \mid b$

$\Rightarrow$

$S \to ABA \mid bB \mid b \mid AA \mid BA \mid AB \mid aA \mid a$

$A \to aA \mid a$

$B \to bB \mid b$

let $P \to a$

$Q \to AB$

$R \to bB$

$\Rightarrow S \to QA \mid RB \mid b \mid AA \mid BA \mid AB \mid PA \mid a$

$A \to PA \mid a$

$B \to RB \mid b$

$P \to a$

$Q \to AB$

$R \to bB$

5] $S \to A \mid 0C1$

$A \to B \mid 01 \mid 10$

$C \to \varepsilon \mid CD$

$\Rightarrow$

$S \to A \mid 0C1 \mid 01$

$A \to B \mid 01 \mid 10$

$C \to CD \mid D$

$\to S \to A \mid 0Q1 \mid 01$

$A \to 01 \mid 10$

$\Rightarrow S \to 1B \mid 01$

let $P \to 1$

$Q \to 0$

$\Rightarrow S \to PB \mid QP$

$P \to 1$

$Q \to 0$

6] $S \rightarrow aB \mid bA$

$A \rightarrow a \mid aS \mid bAA$

$B \rightarrow b \mid aS \mid aBB$

let $P \rightarrow a$

$Q \rightarrow b$

$R \rightarrow AA$

$T \rightarrow BB$

$\Rightarrow S \rightarrow PB \mid QA$

$A \rightarrow a \mid PS \mid QR$

$B \rightarrow b \mid PS \mid PT$

Griebach Normal Form:-

It can be defined as

nonterminal → terminal. Any no. of nonterminals.

Ex:- $S \rightarrow a$, $S \rightarrow aS$, $S \rightarrow aSSS$ } GNF

$S \rightarrow aa$, $S \rightarrow Sa$, $S \rightarrow SS$ } not GNF

1] $S \rightarrow CA$

$A \rightarrow a$

$C \rightarrow aB \mid b$

$\Rightarrow$ useless symbols

$S \rightarrow CA$

$A \rightarrow a$

$C \rightarrow b$

$\Rightarrow S \rightarrow bA$

$A \rightarrow a$

2] $S \rightarrow ABA$

$A \rightarrow aA \mid \varepsilon$

$B \rightarrow bB \mid \varepsilon$

$\Rightarrow$ ε-productions

$S \rightarrow ABA \mid AB \mid BA \mid B \mid AA \mid A$

$A \rightarrow aA \mid a$

$B \rightarrow bB \mid b$

$\Rightarrow$ unit productions

$S \rightarrow ABA \mid AB \mid BA \mid bB \mid b \mid AA \mid aA \mid a$

$A \rightarrow aA \mid a$

$B \rightarrow bB \mid b$

$\Rightarrow S \rightarrow aABA \mid aBA \mid aAB \mid aB \mid bBA \mid bA \mid aAA \mid aA \mid a \mid bB \mid b$

$A \rightarrow aA \mid a$

$B \rightarrow bB \mid b$

5] $S \to AA \mid 0$

$A \to SS \mid 1$

$\Rightarrow$ useless

$S \to 0$

$S \to SS \mid 1$

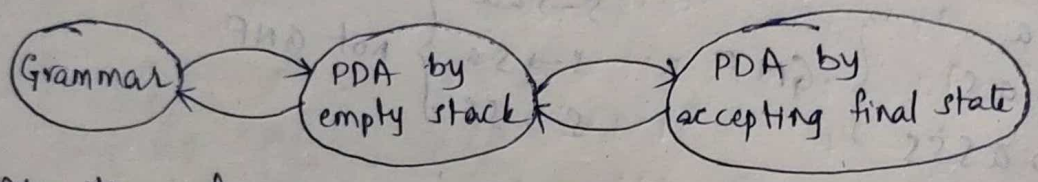
$\Rightarrow S \to 0$

$S \to 0S \mid 1$

$\Rightarrow S \to 0S \mid 1S \mid 0 \mid 1$

Equivalence of PDA's & CFG's :-

The language defined by PDA are exactly context free language. The following diagram represents PDA's and CFG's



Algorithm for conversion of PDA to CFG :-

Let $P = (Q, \Sigma, \delta, q_0, F, \Gamma, z_0)$ be a PDA, then there is a CFG, G such that $L(G) = L(P)$

Proof :-

Construct $G = \{V, T, P, S\}$ where the set of variables represented by the V, set of input symbols or terminals represented by T, production rules by P and starting symbol by S.

All symbols of the form $[pxq]$ where p, q are states in Q. x is a stack symbol in Γ.

The production of G are as follows.

(a) for all states p. G has a production $S \to [q_0 z_0 p]$

(b) let $\delta(q, a, x)$ contain the pair $(r, y_1, y_2, \cdots y_i)$ where a is either a symbol in Σ or ε and i can be any no. including '0' in which case, the pair is (r, ε), then for all list of states $(r_1, r_2, \cdots r_i)$, G has a production $[qx r_i] \to$

$[q_i x r_i] \rightarrow a[r y_1 r_1][r y_2 r_2] \cdots [r_n y_{n+1} r_{n+1}] \cdots [r_{i-1} y_i r_i]$

Convert the PDA = $(\{p,q\}, \{0,1\}, \{x, z_0\}, \delta, q_0, z_0)$ to CFG if δ is given by

1. $\delta(q, 1, z_0) = (q, x z_0)$
2. $\delta(q, 0, x) = (p, x)$
3. $\delta(q, 1, x) = (q, xx)$
4. $\delta(q, \varepsilon, x) = (q, \varepsilon)$
5. $\delta(p, 1, x) = (p, \varepsilon)$
6. $\delta(p, 0, z_0) = (q, z_0)$

Sol:-

step1 :- finding start symbol i.e,

$S \rightarrow q z_0 q \mid q z_0 p$

step2 :- find productions from given rules.

consider rule 1 : $\delta(q, 1, z_0) = (q, x z_0)$

$[q z_0 q] \rightarrow 1[q x q][q z_0 q]$

$[q z_0 q] \rightarrow 1[q x p][p z_0 q]$

$[q z_0 p] \rightarrow 1[q x q][q z_0 p]$

$[q z_0 p] \rightarrow 1[q x p][p z_0 p]$

step3:-

rule 3:

$\delta(q, 1, x) = (q, xx)$

$[q x q] \rightarrow 1[q x q][q x q]$

$[q x q] \rightarrow 1[q x p][p x q]$

$[q x p] \rightarrow 1[q x q][q x p]$

$[q x p] \rightarrow 1[q x p][p x p]$

rule 2 :-

$\delta(q, 0, x) = (p, x)$

$[q x q] \rightarrow 0[p x q]$

$[q x p] \rightarrow 0[p x p]$

rule 6:

$\delta(p,0,z_0) = (q, z_0)$

$[pz_0q] \rightarrow 0[qz_0q]$

$[pz_0p] \rightarrow 0[qz_0p]$

rule 5:

$\delta(p,1,x) = (p,\varepsilon)$

$[pxp] \rightarrow 1$

rule 4: $\delta(q,\varepsilon,x) = (q,\varepsilon)$

$[qxq] \rightarrow \varepsilon$

let $A \rightarrow [qz_0p]$

$B \rightarrow [qz_0q]$

$C \rightarrow [qxq]$

$D \rightarrow [qxp]$

$E \rightarrow [pz_0q]$

$F \rightarrow [pz_0p]$

$G \rightarrow [pxq]$

$H \rightarrow [pxp]$

$S \rightarrow A|B$

$B \rightarrow 1CB$

$B \rightarrow 1DE$

$A \rightarrow 1CA$

$A \rightarrow 1DF$

$C \rightarrow 1CC$

$C \rightarrow 1DG$

$D \rightarrow 1CD$

$D \rightarrow 1DH$

$C \rightarrow 0G$

$D \rightarrow 0H$

$E \rightarrow 0B$

$F \rightarrow 0A$

$H \rightarrow 1$

$C \rightarrow \varepsilon$

$S \rightarrow A|B$

$A \rightarrow 1CA|1DF$

$B \rightarrow 1CB|1DE$

$C \rightarrow 1CC|1DG|0G|\varepsilon$

$D \rightarrow 1CD|1DH|0H$

$E \rightarrow 0B$

$F \rightarrow 0A$

$H \rightarrow 1$

$A \rightarrow [qz_0p]$

$B \rightarrow [qz_0q]$

$C \rightarrow [qxq]$

$D \rightarrow [qxp]$

$E \rightarrow [pz_0q]$

$F \rightarrow [pz_0p]$

$G \rightarrow [pxq]$

$H \rightarrow [pxp]$

2) Convert the following PDA into grammar.

1. $\delta(q_0, a, z_0) = (q_0, az_0)$
2. $\delta(q_0, a, a) = (q_0, aa)$
3. $\delta(q_0, b, a) = (q_1, a)$
4. $\delta(q_1, b, a) = (q_1, a)$
5. $\delta(q_1, a, a) = (q_1, \varepsilon)$
6. $\delta(q_1, \varepsilon, z_0) = (q_1, \varepsilon)$

Step 1 :- $S \rightarrow [q_0 z_0 q_0] | [q_0 z_0 q_1]$

Step 2 :-

rule 1: $\delta(q_0, a, z_0) = (q_0, az_0)$

$[q_0 z_0 q_0] \rightarrow a[q_0 a q_0][q_0 z_0 q_0]$

$[q_0 z_0 q_0] \rightarrow a[q_0 a q_1][q_1 z_0 q_0]$

$[q_0 z_0 q_1] \rightarrow a[q_0 a q_0][q_0 z_0 q_1]$

$[q_0 z_0 q_1] \rightarrow a[q_0 a q_1][q_1 z_0 q_1]$

rule 2: $\delta(q_0, a, a) = (q_0, aa)$

$[q_0 a q_0] \rightarrow a[q_0 a q_0][q_0 a q_0]$

$[q_0 a q_0] \rightarrow a[q_0 a q_1][q_1 a q_0]$

$[q_0 a q_1] \rightarrow a[q_0 a q_0][q_0 a q_1]$

$[q_0 a q_1] \rightarrow a[q_0 a q_1][q_1 a q_1]$

rule 3: $\delta(q_0, b, a) = (q_1, a)$

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

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

rule 4: $\delta(q_1, b, a) = (q_1, a)$

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

$[q_1 a q_1] \rightarrow b[q_1 a q_1]$

rule 5: $\delta(q_1, a, a) = (q_1, \varepsilon)$

$[q_1 a q_1] \rightarrow a$

rule 6: $\delta(q_1, \varepsilon, z_0) = (q_1, \varepsilon)$

$[q_1 z_0 q_1] \rightarrow \varepsilon$

et

$A \to [q_0 z_0 q_0]$

$B \to [q_0 z_0 q_1]$

$C \to [q_0 a q_0]$

$D \to [q_0 a q_1]$

$E \to [q_1 z_0 q_0]$

$F \to [q_1 z_0 q_1]$

$G \to [q_1 a q_0]$

$H \to [q_1 a q_1]$

$S \to A/B$

$A \to aCA$

$A \to aDE$

$B \to aCB$

$B \to aDF$

$C \to aCC$

$C \to aDG$

$D \to aCD$

$D \to aDH$

$C \to bG$

$D \to bH$

$G \to bG$

$H \to bH$

$H \to a$

$F \to \varepsilon$

$S \to A/B$

$A \to aCA/aDE$

$B \to aCB/aDF$

$C \to aCC/aDG/bG$

$D \to aCD/aDH/bH$

$G \to bG$

$H \to bH/a$

$F \to \varepsilon$

$(q_0, a, X)/XX$

$(q_0, a, z_0)/Xz_0$

$(q_0, \varepsilon, z_0)/\varepsilon$

3]

$(q_0, b, z_0)/Yz_0$

$(q_0, b, Y)/YY$

$(q_0, a, Y)/\varepsilon$

$(q_0, b, X)/\varepsilon$

1. $\delta(q_0, \varepsilon, z_0) = (q_0, \varepsilon)$
2. $\delta(q_0, a, z_0) = (q_0, Xz_0)$
3. $\delta(q_0, a, X) = (q_0, XX)$
4. $\delta(q_0, b, z_0) = (q_0, Yz_0)$
5. $\delta(q_0, b, Y) = (q_0, YY)$
6. $\delta(q_0, a, Y) = (q_0, \varepsilon)$
7. $\delta(q_0, b, X) = (q_0, \varepsilon)$ $\longrightarrow$ $S \to [q_0 z_0 q_0]$

rule 1: $[q_0 z_0 q_0] \to \varepsilon$

rule 2: $\delta(q_0, a, z_0) = (q_0, Xz_0)$

$[q_0 z_0 q_0] \longrightarrow a[q_0 X q_0][q_0 z_0 q_0]$

rule 3: $\delta(q_0, a, X) = (q_0, XX)$

$[q_0 X q_0] \to a[q_0 X q_0][q_0 X q_0]$

rule 4: $\delta(q_0, b, z_0) = (q_0, Yz_0)$

$[q_0 z_0 q_0] \rightarrow b[q_0 Y q_0][q_0 z_0 q_0]$

rule 5: $\delta(q_0, b, Y) = (q_0, YY)$

$[q_0 Y q_0] \rightarrow b[q_0 Y q_0][q_0 Y q_0]$

rule 6: $\delta(q_0, a, Y) = (q_0, \varepsilon)$

$[q_0 Y q_0] \rightarrow a$

rule 7: $\delta(q_0, b, Y) = (q_0, \varepsilon)$

$[q_0 Y q_0] \rightarrow b$

let $A \rightarrow [q_0 z_0 q_0]$

$B \rightarrow [q_0 X q_0]$

$C \rightarrow [q_0 Y q_0]$

$S \rightarrow A$

$A \rightarrow aBA \mid \varepsilon$

$B \rightarrow aBB$

$A \rightarrow bCA$

$C \rightarrow bCC$

$C \rightarrow a$

$B \rightarrow b$

(A)

$S \rightarrow A$

$A \rightarrow aBA \mid bCA \mid \varepsilon$

$B \rightarrow aBB \mid b$

$C \rightarrow bCC \mid a$

4] Consider PDA, $P = (\{q\}, \{0,1\}, \{X, Z, U\}, \delta, q, z)$

where 1. $\delta(q, \varepsilon, z) = (q, \varepsilon)$

2. $\delta(q, 0, z) = (q, Xz)$

3. $\delta(q, 0, X) = (q, XX)$

4. $\delta(q, 1, z) = (q, Uz)$

5. $\delta(q, 1, U) = (q, UU)$

6. $\delta(q, 0, U) = (q, \varepsilon)$

7. $\delta(q, 1, X) = (q, \varepsilon)$

$S \rightarrow [qzq]$

rule 1:

$[qzq] \rightarrow \varepsilon$

rule 2:

$[qzq] \rightarrow 0[qxq][qzq]$

rule 3:

$[qxq] \rightarrow 0[qxq][qxq]$

rule 4:

$[qzq] \rightarrow 1[qUq][qzq]$

rule 5:

$[qUq] \rightarrow 1[qUq][qUq]$

rule 6:

$[qUq] \rightarrow 0$

rule 7:

$[qXq] \rightarrow 1$

$A \rightarrow [qZq]$

$B \rightarrow [qXq]$

$C \rightarrow [qUq]$

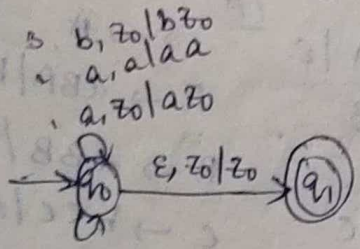
$S \rightarrow A$

$A \rightarrow \varepsilon \mid 0BA \mid 1CA$

$B \rightarrow 0BB \mid 1$

$C \rightarrow 1CC \mid 0 \mid 1$

5]



4 b, b | bb

5 a, a | ε

6 b, b | ε

$S \rightarrow [q_0 z_0 q_0] \mid [q_0 z_0 q_1]$

rule 1:

$\delta(q_0, a, z_0) = (q_0, az_0)$

$[q_0 z_0 q_0] \rightarrow a[q_0 a q_0][q_0 z_0 q_0]$

$[q_0 z_0 q_0] \rightarrow a[q_0 a q_1][q_1 z_0 q_0]$

$[q_0 z_0 q_1] \rightarrow a[q_0 a q_1][q_1 z_0 q_1]$

$[q_0 z_0 q_1] \rightarrow a[q_0 a q_0][q_0 z_0 q_1]$

rule 2: $\delta(q_0, a, a) = (q_0, aa)$

$[q_0 a q_0] \rightarrow a[q_0 a q_0][q_0 a q_0]$

$[q_0 a q_0] \rightarrow a[q_0 a q_1][q_1 a q_0]$

$[q_0 a q_1] \rightarrow a[q_0 a q_1][q_1 a q_1]$

$[q_0 a q_1] \rightarrow a[q_0 a q_0][q_0 a q_1]$

rule 3: $\delta(q_0, b, z_0) = (q_0, bz_0)$

$[q_0 z_0 q_0] \rightarrow b[q_0 b q_0][q_0 z_0 q_0]$

$[q_0 z_0 q_0] \rightarrow b[q_0 b q_1][q_1 z_0 q_0]$

$[q_0 z q_1] \rightarrow b[q_0 b q_1][q_1 z q_1]$

$[q_0 z q_1] \rightarrow b[q_0 b q_0][q_0 z q_1]$

rule 4: $\delta(q_0, b) = (q_0, bb)$

$[q_0 b q_0] \rightarrow b[q_0 b q_0][q_0 b q_0]$

$[q_0 b q_0] \rightarrow b[q_0 b q_1][q_1 b q_0]$

$[q_0 b q_1] \rightarrow b[q_0 b q_1][q_1 b q_1]$

$[q_1 b q_1] \rightarrow b[q_0 b q_0][q_0 b q_1]$

rule 5: $\delta(q_0, a, a) = (q_0, \varepsilon)$

$[q_0 a q_0] \rightarrow a$

~~$[q_0 a q_1] \rightarrow \varepsilon$~~

rule 6: $\delta(q_0, b, b) = (q_0, \varepsilon)$

$[q_0 b q_0] \rightarrow b$

rule 7: $\delta(q_0, \varepsilon, z_0) = (q_1, z_0)$

$[q_0 z_0 q_0] \rightarrow \varepsilon [q_1 z_0 q_0]$

$A \rightarrow [q_0 z_0 q_0]$

$B \rightarrow [q_0 z_0 q_1]$

$C \rightarrow [q_0 a q_0]$

$D \rightarrow [q_0 a q_1]$

$E \rightarrow [q_1 z_0 q_0]$

$F \rightarrow [q_1 z_0 q_1]$

$S \rightarrow A/B$

$A \rightarrow aCA \mid aDE$

$B \rightarrow aDF \mid aCB$

$C \rightarrow aCC \mid aD$

Algorithm for Conversion of Grammar to PDA :-

Let $G = (V, T, P, S)$ be a CFG then construct PDA, P by empty stack as follows

(i) for each variable, $A \to B$

$\delta(q, \varepsilon, A) = \{(q, B) \mid A \to B \text{ is a production of } P\}$

(ii) for each terminal a, $\delta(q, a, a) = \{(q, \varepsilon)\}$

1] Convert the following grammar into PDA by empty stack.

$S \to iEtSeS \mid eS$

$E \to id$.

Soln

In the given grammar, there are two variables S and E

There are i, t, e, id terminals

$\delta(q, \varepsilon, S) = (q, iEtSeS), (q, eS)$

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

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

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

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

$\delta(q, id, id) = (q, t)$

2] $E \to I \mid E*E \mid E+E \mid (E)$

$I \to a \mid b \mid Ia \mid Ib \mid I0 \mid I1$

Soln

variables are E and I

terminals are $*, +, (,), a, b, 0, 1$

$\delta(q, \varepsilon, E) = (q, I), \{q, (E*E)\}, \{q, (E+E)\}, \{q, (E)\}$

$\delta(q, \varepsilon, I) = \{(q, a)\}, \{(q, b)\}, \{(q, Ia)\}, \{(q, Ib)\}, \{(q, I0)\}, \{(q, I1)\}$

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

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

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

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

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

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

$\delta(q, 0, 0) = (q, \varepsilon)$

$\delta(q, 1, 1) = (q, \varepsilon)$

Ex: ~~For terminal~~

Construct PDA for CFG

$S \to AB$

$A \to 0S \mid 0$

$B \to 1S \mid 1$

$S \to AB \quad \delta(q, \epsilon, S) = (q, AB)$

$A \to 0S \quad \delta(q, \epsilon, A) = (q, 0S)$

$A \to 0 \quad \delta(q, \epsilon, A) = (q, 0)$

$B \to 1S \quad \delta(q, \epsilon, B) = (q, 1S)$

$B \to 1 \quad \delta(q, \epsilon, B) = (q, 1)$

$\delta(q, 0, 0) = (q, \epsilon)$

$\delta(q, 1, 1) = (q, \epsilon)$

→ Transition functions

3] $E \to E+T$

$T \to T*F$

$F \to (E) | id$

variables are E, T, F

terminals are $+, *, (,), id$

$\delta(q_1, \varepsilon, E) = (q_1, E+T), (q_1, T)$

$\delta(q_1, \varepsilon, T) = (q_1, T*F)$

$\delta(q_1, \varepsilon, F) = (q_1, (E)), (q_1, id)$

$\delta(q_1, +, +) = (q_1, \varepsilon)$

$\delta(q_1, *, *) = (q_1, \varepsilon)$

$\delta(q_1, (, () = (q_1, \varepsilon)$

$\delta(q_1,),)) = (q_1, \varepsilon)$

$\delta(q_1, id, id) = (q_1, \varepsilon)$

Pumping lemma for CFL :-

A tool that showing that whether the language is a CFL or not is known as pumping lemma for CFL.

Pumping lemma for CFL is similar to pumping lemma for regular languages. But, here we break the input string z into 5 parts i.e, uvwxy and here we pump 2nd and 4th letter for any no. of times.

Def :- Let L be a CFL and z be any input string. Split the string z into 5 parts. i.e, uvwxy in which

(i) $|vwx| \leq n$, where n is no. of symbols.

(ii) $|vx| \neq \varepsilon$

(iii) $\forall i \geq 0$, $uv^i wx^i y$ is in L i.e, v, x are pumped for any no. of times including 0, the resulting string should be in L

i] Prove that the Language $L = \{a^i b^j c^i d^j \mid i, j \geq 1\}$ is not a CFL

Suppose L is a CFL,

Consider any input string $z = a^k b^k c^k d^k$.

split the string z into 5 parts.

$z = uvwxy = a^k b^{k+1} b \cdot c^{k-1} c \cdot d^k$

here $u = a^k b^{k-1}$, $v = b$, $w = c^{k-1}$, $x = c$, $y = d^k$

pump v & x for any no of times

i.e., $z = uv^i wx^i z$

Consider $i=2$, then $z = a^k b^{k+1} b^2 c^{k-1} c^2 d^k$

$= a^k b^{k+1} c^{k+1} d^k \notin L$

∴ By contradiction, the given language is not a CFL

2] PT the language $L = \{a^n b^n c^n / n \geq 1\}$ is not a CFL

3] $L = \{a^i b^j c^k / i < j \text{ and } j < k\}$ is not a CFL

~~$z = uvwxy = a^n b^{n+1} c^{n+2}$~~ consider

$z = uvwxy = a^{n-1} \cdot a \cdot b^{n+1} \cdot c^{n+2}$

$u \; v \; w \; x \; y$

$u = a^{n-1}$, $v = a$, $w = b^{n+1}$, $x = c^{n+2}$, $y = \varepsilon$

pump v and x for any times.

$z = uv^i wx^i y$.

Consider $i=2$, $z = a^{n-1} \cdot a^2 b^{n+1} \cdot c^{2n+4} \notin L$

∴ Given language is not a CFL

4] P.T language $L = \{a^n b^m c^n / m > n\}$ is not CFL

$z = a^k b^{k+1} c^k$

$z = uvwxy = a^{k-1} \cdot a \cdot b^{k+1} \cdot c^k$

$u \; v \; w \; x \; y$.

pump v and x

$z = a^{k-1} a^i b^{k+1} c^i$

$i=2$

$z = a^{k-1} \cdot a^2 b^{k+1} c^{2k} = a^{k+1} b^{k+1} c^{2k}$

not a CFL

5] $L = \{ww / w \in (0,1)^*\}$ is not CFL.

$z = 0^n 1^n 0^n 1^n$ $u = 0^n$, $v = 1^n$, $w = 0^n$, $x = 1^n$

$u \; v \; w \; x \; y$

Pump v and x. $\Rightarrow 0^n 1^{in} 0^n 1^{in}$

$i=2$

$z = 0^n 1^{2n} 0^n 1^{2n}$ not a CFL

closure properties of CFL :-

* CFLs are closed under the following operations.

1. union
2. concatenation
3. closure (*) and positive closure (+)
4. Homomorphism

1. Union :-
Let L_1, L_2 be CFL s then $L_1 \cup L_2$ is a language $S(L)$ where S is substitution function.
L is $\{1,2\}$
then substitution, s defined by $s(1) = L_1$, $s(2) = L_2$

2. Concatenation :- Let L_1 & L_2 be two CFL 's then $L_1 L_2$ is a language $S(L)$ where L is the language $\{1,2\}$ and S is same substitution as defined in above case.

3. Closure (*) :- If L_1 is CFL, L is language $\{1\}^*$ and S is substitution $s(1) = L_1$ then $L_1^* = S(L)$.
My if the language $L = \{1\}^+$ then $L_1^+ = S(L)$

4. Homomorphism :- let L is a CFL over Σ and h is a homomorphism on Σ. Let 's' be the substitution that replaces each symbol a in Σ by the language consisting of one string i.e, $h(a)$ i.e, $s(a) = \{h(a)\}$ $\forall$ a in Σ
then $h(L) = S(L)$

Decision properties of Context Free languages :-

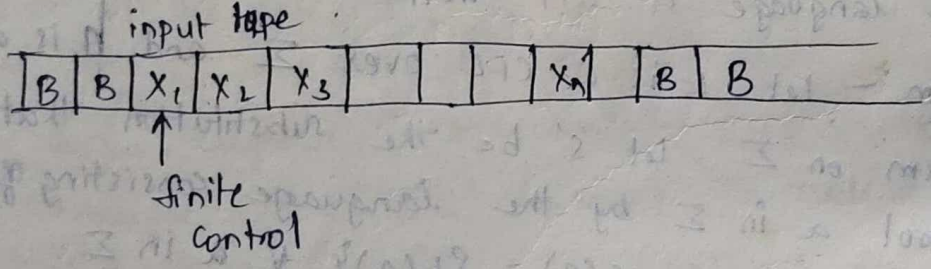
UNIT-IV

→ Decision Algorithm depend on converting context Free Grammar into CNF. The following is the list of observations

1. The reachable and generating symbols for grammar can be done in $O(n)$ times.
2. Constructing unit pairs and eliminating unit productions takes $O(n^4)$ time and resulting grammar has length $O(n^2)$
3. Replacement of terminals by variables in production bodies take $O(n)$ time.
4. Breaking of production bodies of length three or more into bodies of length two also take $O(n)$ time and results in a grammar length $O(n)$.

Turing machine :-

The abract model of computing machine is turing machine. The basic model of turing machine is shown below.



Turing machine consists of (i) input tape.
(ii) finite control
(iii) Blank

Input tape :- The input which is finite length string of symbols choosen from input alphabet Σ is placed on the tape. The tape is divided into cells or space. Each cell can store any no. of finite set of symbols.

Tape cells extending to the left and right infinitely. The input tape initially holds a special symbol called Blank.

finite control :- Finite control scans cell of an input tape at a time.

Blank :- It is a tape symbol but not an input symbol.

A move of turing machine is state of finite control and tape symbol scan.

In one move, the turing machine will

(i) change the state

(ii) write a tape symbol in the cell scan

(iii) move the tape head left or right.

Formal Notation for Turing machine :-

The Formal notation of turing machine is similar to FA or PDA.

The 7-tupe Turing machine is described as

$M = (Q, \Sigma, \delta, q_0, F, B, \Gamma)$ whose components have the following names.

Q- finite set of states of finite control

Σ - finite set of input symbols

Γ - Complete set of ~~tape~~ symbols, $\Sigma \subseteq \Gamma$

δ - transition function, $\delta(q, x)$ is defined as triple (P, Y, D) where P denotes next state in Q. Y denotes symbol in Γ. D denotes the direction either ~~from~~ left(L) or right (R)

q_0 - starting state

F - set of final states.

B - Blank symbol, symbol in Γ not in Σ

1] Construct TM for $L = \{a^n b^n / n \geq 1\}$

Sol

BBaabbBB Convert 'a' into x and move right insearch of 'b'

BBXabbBB Keep it same i.e, move right till 'b' is reached.

BBXabbBB Convert 'b' into 'Y' move left in search of 'x'.

BBXaYbBB Keep it same, move left until X is reached.

BBXaYbBB Convert 'a' into x & move right in search of b.

Decision properties of Context Free languages :-

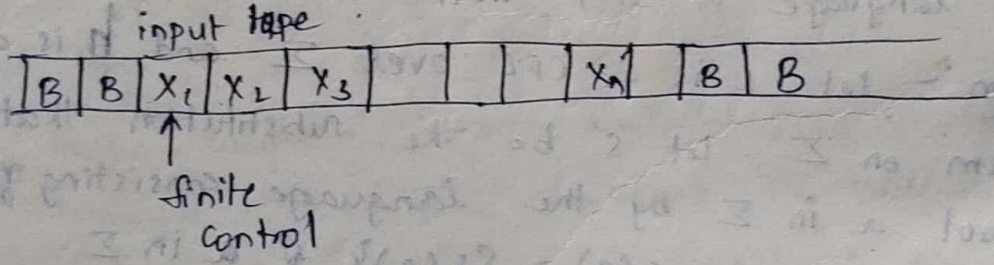
UNIT-4

→ Decision Algorithm depend on converting context Free Gramar into CNF. The following is the list of observations

1. The reachable and generating symbols for grammar can be done in $O(n)$ times.
2. Constructing unit pairs and eliminating unit productions takes $O(n^4)$ time and resulting grammar has length $O(n^2)$
3. Replacement of terminals by variables in production bodies take $O(n)$ time.
4. Breaking of production bodies of length three or more into bodies of length two also take $O(n)$ time and results in a gramar length $O(n)$.

Turing machine :-

The abract model of computing machine is turing machine the basic model of turing machine is shown below.



Turing machine consists of (i) input tape.
(ii) finite control
(iii) Blank

Input tape :- The input which is finite length string of symbols choosen from input alphabet Σ is placed on the tape. The tape is divided into cells or space. Each cell can store any no. of finite set of symbols.

Tape cells extending to the left and right infinitely. The input tape initially holds a special symbol called Blank.

finite control :- finite controls scans cell of an input tape at a time.

Blank : It is a tape symbol but not an input symbol.

A move of turing machine is state of finite control and tape symbol scan.

In one move, the turing machine will

(i) change the state
(ii) write a tape symbol in the cell scan
(iii) move the tape head left or right.

Formal Notation for Turing machine

The Formal notation of turing machine is similar to FA or PDA's

The 7-tupe Turing machine is described as

$M = (Q, \Sigma, \delta, q_0, F, B, \Gamma)$ whose components have the following names.

Q- finite set of states of finite control

Σ - finite set of input symbols

Γ - Complete set of ~~tape~~ symbols, $\Sigma \subseteq \Gamma$

δ - transition function, $\delta(q, x)$ is defined as triple (P, Y, D) where P denotes next state in Q. Y denotes symbol in Γ. D denotes the direction either ~~from~~ left(L) or right (R)

q_0 - starting state

F - set of final states.

B - Blank symbol, symbol in Γ not in Σ

1] Construct TM for $L = \{a^n b^n \mid n \geq 1\}$

Sol

BBaabbBB — Convert 'a' into x and move right insearch of 'b'

BBXabbBB — Keep it same i.e, move right till 'b' is reached.

BBXabbBB — convert 'b' into 'Y' move left in search of 'x'.

BBXaYbBB — Keep it same, move left until x is reached.

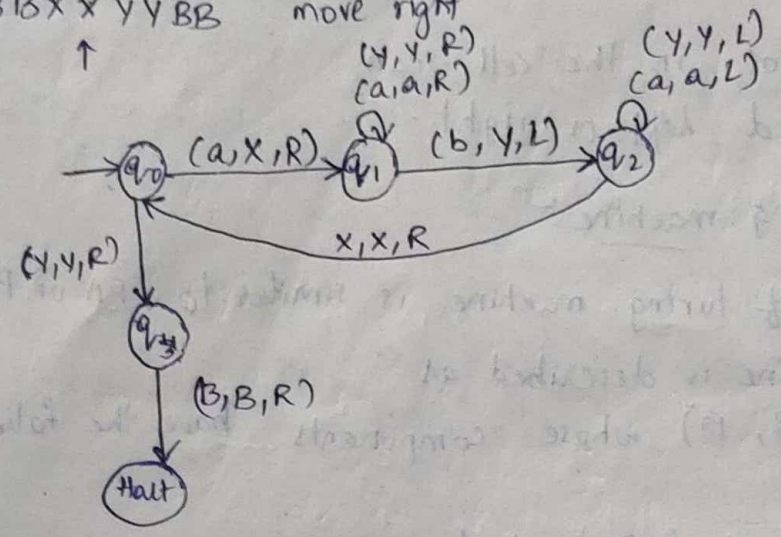
BBXaYbBB — Convert 'a' into x & move right in search of b.

BBXXYbBB Keep it same, move right until b is reached.
↑

BBXXYbBB Convert b into Y ie, move left insearch of 'X'.
↑

BBXXYYBB Keep it same, move left
↑

BBXXYYBB move right
↑



	a	b	B	X	Y
q_0	(q_1,X,R)				(q_3,Y,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			(halt, B, R)		
Halt					

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

B → blank

~~let n=2~~

BB a abb c c BB Convert a into X, move right in search of b.
↑

BB X a bb c c BB Keep it same, move right.
↑

BB X a b b c c BB Convert b into Y, move right in search of 'c'.
↑

BB X a Y b c c BB Keep it same, move right
↑

BB X a Y b c c BB Convert c into Z, move left in search of X
↑

BB X a Y b Z c BB Keep it same, move left for X
↑

BBXaYbZcBB Keep it same, move left for X.

BBXaYbZcBB Keep it same, move right

BBXaYbZcBB Convert 'a' into x and move right for b

BBXXYbZcBB Keep it same, move right for b

BBXXYbZcBB Convert b into Y and move right for c

BBXXYbZcBB Keep same and move right for c

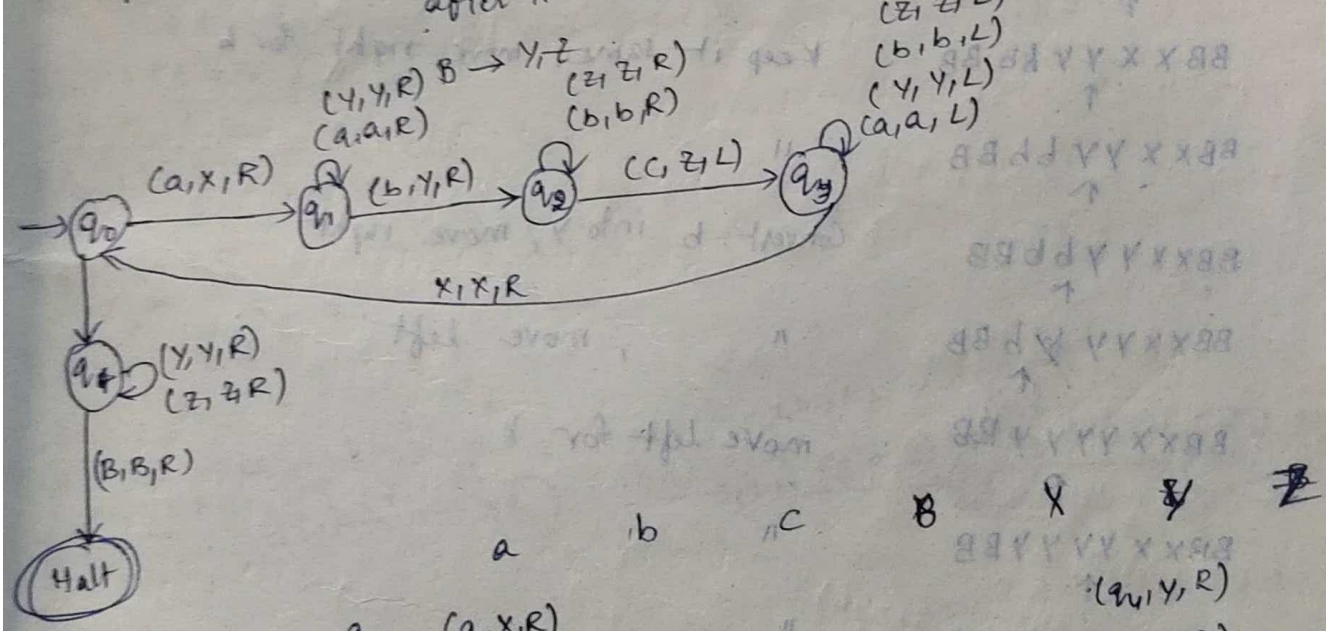
BBXXYYZcBB Convert 'c' into Z and move left for X.

BBXXYYZcBB Keep it same, move left in search of X.

BBXXYYZZBB "

BBXXYYZZBB "

BBXXYYZZBB Keep it same, move right. ~~for~~ Y is appeared after X immediately, stop the process.



	a	b	c	B	X	Y	Z
q_0	(q_1, X, R)					(q_4, Y, R)	
q_1	(q_1, a, R)	(q_2, Y, R)				(q_4, Y, R)	
q_2		(q_2, b, R)	(q_3, Z, L)				(q_2, Z, R)
q_3	(q_3, a, L)	(q_3, b, L)	~~q_3~~	(q_0, X, R)		(q_3, Y, R)	(q_3, Z, R)
q_4				$(Halt, B, R)$		(q_4, Y, R)	(q_4, Z, R)
~~Halt~~							

3] $L = \{a^n b^{2n} / n \geq 1\}$

$n=2$

BBa abbbb BB — Convert a to X, move right

BBX abbbb BB — Keep same, move right for b

BBXa bbbb BB — Convert b to Y, move right

BBXaYbbb BB — Convert b to Y, move left

BBXaYbbBB — Keep it same, move left for X

BBXaYYbbBB — "

BBXaYYbbBB — move right for a

BBXaYYbbBB — Convert a to X, move right of b

BBXXYYbbBB — Keep it same, move right for b

BBXXYYbbBB — "

BBXXYYbbBB — Convert b into Y, move right

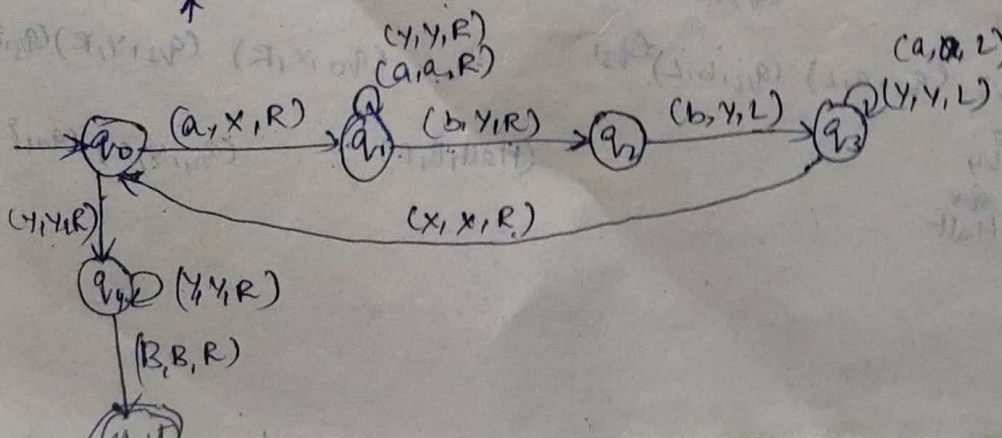
BBXXYYbbBB — " , move left

BBXXYYYYBB — move left for X

BBXXYYYYBB — "

BBXXYYYYBB — "

BBXXYYYYBB — move right, Y is appeared



	a	b	X	Y	Z	B
q_0	(q_1, X, R)			(q_4, Y, R)		
q_1	(q_1, a, R)	(q_2, Y, R)		(q_1, Y, R)		
q_2		(q_3, Y, L)				
q_3	(q_3, a, L)		(q_0, X, R)	(q_3, a, L)		
q_4				(a_0, Y, R)		$(Halt, B, R)$
Halt						

y] Compute following function $f(n) = n+1$ where n is unary number using Turing machine.

n=4

BB1111+1BB move right for '+'

BB1111+1BB same, right '+'

BB1111+1BB "

BB1111+1BB "

BB1111+1BB Convert '+' into 1 move right for B

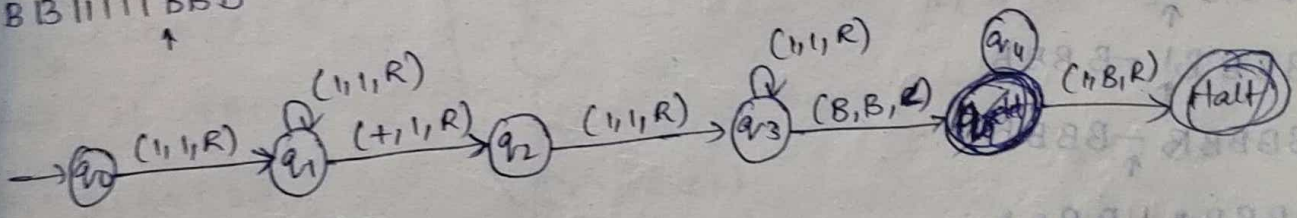
BB1111 1BB move right for B

BB11111BB move left.

BB11111BB Convert 1 to B

BB11111BB

BB11111BBB



	1	+	B
q_0	$(q_1, 1, R)$		
q_1	$(q_1, 1, R)$	$(q_2, 1, R)$	
q_2	$(q_3, 1, R)$		
q_3	$(q_3, 1, R)$		(q_4, B, L)
q_4	$(Halt, B, R)$		
Halt			

5] Subtraction of two numbers.

$f = m - n$ where m & n are unary numbers.

BB|||−||BB
↑

BBB||−||BB
↑

BBBB|−||BB
↑

BBB||−||BB
↑

BBB||−||BB
↑

BBB||−||BB
↑

BBB||−||BB
↑

BBB||−|BBB
↑

BBB||−BBBB
↑

BBB||−BBBB
↑

BBB||−BBBB
↑

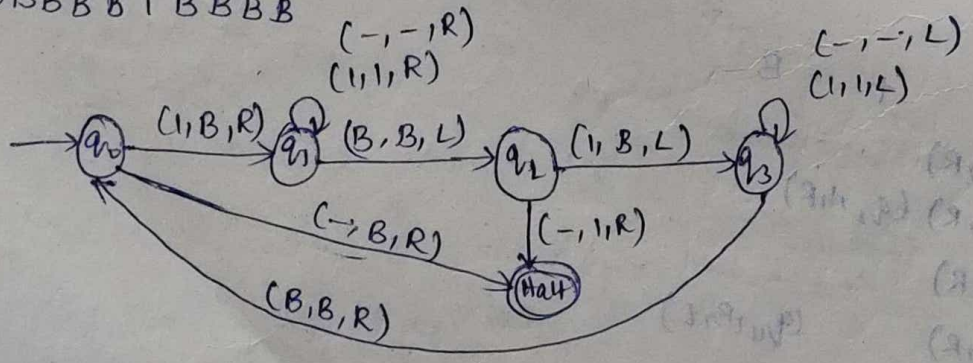
BBB||−BBBB
↑

BBB||−BBBB
↑

BBBB|−BBBB
↑

BBBBB−BBBB
↑

BBBBB|BBBB



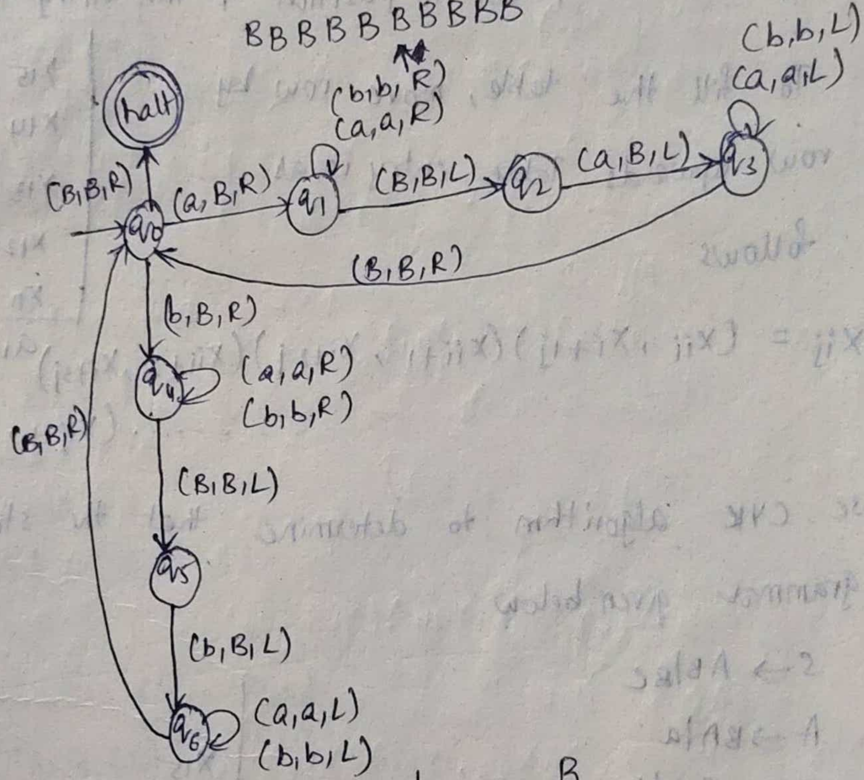
6] $f = m * n$

7] $L = \{ww^T \mid w \in (a,b)^+\}$

let $w = aab$, $w^T = baa$

BB aabbaa BB
↑
BBBabbaaBB
↑
BBBabbaaBB
↑
BBBabbaaBB
↑
BBBabbaaBB
↑↑
BBBabbaaBB
↑
BBBabbaaBB
↑
BBB abba a BB
↑
BBB abbaBBB
↑
BBB abbaBBB
↑
BBB abba BBB
↑
BBB abba BBB
↑
BBB abba BBB
↑
BBB abba BBB
↑
BBB a bba BBB
↑
BBB B bba BBB
↑
BBBB bba BBB
↑
BBBB bba BBB
↑
BBBB bba BBB
↑
BBB B bba BBB
↑↑
BBB B b bBBBB
↑
BBBB b bBBBB
↑
BBBB b b BBBB
↑

BB BB bb BB B B
↑
BBBBb bBBBB
↑
BBBB B bBBBB
↑
BBBBB b BBBB
↑
BBBBB b BBBB
↑
BBBBBb BBBB
↑
BBBBBBBBBB
↑



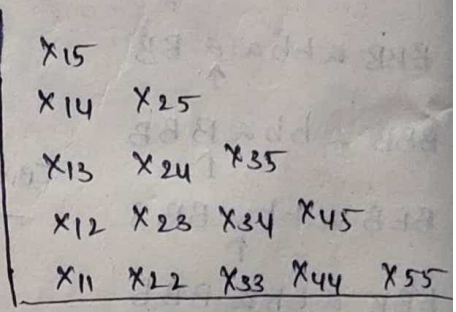
	a	b	B
→ q0	(q1, B, R)	(q4, B, R)	(halt, B, R)
q1	(q1, a, R)	(q1, a, R)	(q2, B, L)
q2	(q3, B, L)		
q3	(q3, a, L)	(q3, b, L)	(q0, B, R)
q4	(q4, a, R)	(q4, b, R)	(q5, B, L)
q5	(q6, B, L)		
q6	(q6, a, L)	(q6, b, L)	(q0, B, R)
halt			

Testing membership of a string in CFL :-

There is an efficient technique based on the idea of dynamic programming which may also be known as cocke-younger kasami algorithm.

In CYK algorithm, construct a triangular table in which horizontal axis as corresponds to position of the string $w = a_1, a_2, \ldots a_n$.

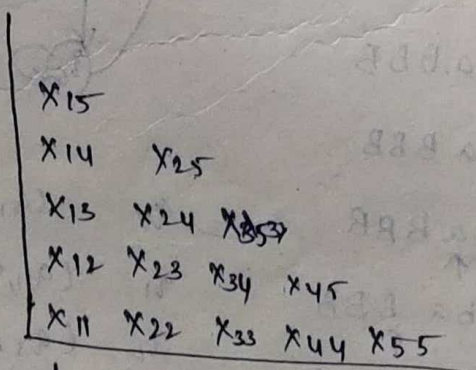
To fill the table, move row by row upwards. Table entry is as follows



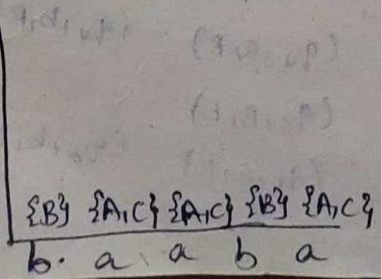
$$x_{ij} = (x_{ii}, x_{i+1j})(x_{ii+1}, x_{i+2j})(x_{ii+2}, x_{i+3j}) \ldots (x_{ij-1}, x_{jj})$$

Use CYK algorithm to determine that the string baaba is in grammar given below.

$S \rightarrow AB/BC$
$A \rightarrow BA/a$
$B \rightarrow CC/b$
$C \rightarrow AB/a$



To construct first row, use basic rule i-e, variables have the production body



$X_{12} = (X_{11}, X_{22})$

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

$X_{12} = \{BA, BC\} = \{S, A\}$

$X_{23} = (X_{22}, X_{33})$

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

$= (AA, AC, CA, CC)$

$= \{B\}$

$X_{34} = (X_{33}, X_{44})$

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

$= \{AB, CB\} = \{S, C\}$

$X_{45} = (X_{44}, X_{55}) = (\{B\}, \{A,C\})$

$= \{BA, BC\} = \{A, S\}$

$X_{13} = (X_{11}, X_{23})(X_{12}, X_{33})$

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

$= (BB)(SA, SC, AA, AC)$

$= \emptyset$

$X_{24} = (X_{22}, X_{34})(X_{23}, X_{44})$

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

$= (\{AS, CS\})(BB) = \emptyset \{B\}$

$X_{35} = (X_{33}, X_{45})(X_{34}, X_{55})$

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

$= \emptyset \{B\}$

$X_{14} = (X_{11}, X_{24})(X_{12}, X_{34})(X_{13}, X_{44})$

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

$= \emptyset$

$X_{25} = (X_{22}, X_{35})(X_{23}, X_{45})(X_{24}, X_{55})$

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

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

$= \{S, C, A\}$

$X_{15} = (X_{11}, X_{25})(X_{12}, X_{35})(X_{13}, X_{45})(X_{14}, X_{55})$

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

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

$\emptyset$				
$\emptyset$	$\emptyset$	~~$\{B,C,A\}$~~ $\{B\}$		
$\{S,A\}$	$\{B\}$	$\{S\}$	$\{A\}$	
$\{B\}$	$\{A,C\}$	$\{A,C\}$	$\{B\}$	$\{A,C\}$
b	a	a	b	a

i.e, X_{15} contains start symbol, S, so, the given string baaba is in given CFG

2] (i) ababa
(ii) baab
(iii) abba

$S \to AB|BC$
$A \to BA|a$
$B \to CC|b$
$C \to AB|a$

$X_{ij} = (X_{ii}, X_{i+1\,j})(X_{i+1}\,{}_iX_{i+2\,j}) \ldots (X_{ij-1}, X_{jj})$

$X_{12} = (X_{11}, X_{22})$
$= (\{A, C\})(\{B\})$
$= (AB, \cancel{\Box}B)$
$= \{S, C\}$

$X_{23} = (X_{22}, X_{33})$
$= (\{B\}, \{A, C\}) = \{A, S\}$

$X_{34} = (X_{33}, X_{44})$
$= (\{A, C\}, \{B\}) = \{S, C\}$

$X_{45} = \{S, A\}$

$X_{13} = (X_{22}, X_{34})(X_{23}, X_{44})$
$= \{B\}$

$X_{24} = \{S, C\}$

$X_{35} = \{B\}$

$X_{14} = (X_{11}, X_{24})(X_{12}, X_{44})(X_{13}, X_{44})$
$= \{\cancel{A, S, B}\}\ \{B\}$

$X_{25} = (X_{22}, X_{35})(X_{23}, X_{45})(X_{24}, X_{55})$
$= (\{B\}, \{B\})(\{A, S\}, \{S, A\})(\{S, C\}, \{A, C\})$
$= \{B, \cancel{A, S}\}$

$X_{15} = \{S, A, C\}$

i.e, X_{15} contains start symbol.
- accepted.

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

chomsky's heirarchy:-

It represents a class of languages accepted by different machines. The categories of languages in chomsky's heirarchy is given below.

language class	language	Grammar	machine	one example
Type 3	R.L	R.G	F.A	a^*b^*
Type 2	CFL	CFL	PDA	a^nb^n
Type 1	Context sensitive language (decidable)	CSG	LBA (linear bounded automata)	$a^nb^nc^n$
Type 0	recursively enumerable language (Computable language)	unrestricted grammar	TM	$n!$

Undecidability and post correspondence problem:-

Def 1:- there are some problems which are not computable, such problems are said to be unsolvable or undecidable.

Def 2:- language L is subset of Σ^* is recursive, if $\exists$ turing machine that satisfies the following conditions.

(i) If $w \in L$, then TM accepts w (i.e, reaches an accepting state on processing w) and halts.

(ii) if $w \notin L$ then H eventually halts without reaching an accepting state.

Def:- A language $L \subseteq \Sigma^*$ is recursively innumerable if $\exists$ a TM, M such that $L = L(M)$

A problem with two answers (yes/no) is decidable if the corr. language is recursible. In this case, the language L is called decidable.

A problem or language is undecidable if it is not decidable.

Post Correspondence problem (PCP):-

The undecidability of string is determined by PCP

PCP consists of two lists of strings that are of equal length over the input Σ, the two lists are

$A = w_1 w_2 w_3 \ldots w_n$ and $B = x_1 x_2 x_3 \ldots x_n$ then $\exists$ a non empty set of integers $I_1 I_2 I_3 \ldots I_n$ such that $w_1 w_2 w_3 \ldots w_n = x_1 x_2 x_3 \ldots x_n$

PCP with two lists $X = (b, bab^3, ba)$ and $Y = (b^3, ba, a)$ has a solution with sequence 2113

2	1	1	3
bab^3	b	b	ba

2	1	1	3
bab^3	b^3	b^3	a

Check whether the PCP with two lists has solution or not.

(i) $X = (0, 01000, 01)$
$\quad\quad\quad 1 \quad 2 \quad 3$

$Y = (000, 01, 1)$
$\quad\quad\quad 1 \quad 2 \quad 3$

2113

(ii) $X = (01, 1, 1)$
$\quad\quad\quad 1 \quad 2 \quad 3$

$Y = (01^2, 10, 1)$
$\quad\quad\quad 1 \quad 2 \quad 3$

$\forall\ x_i \in X$ and $y_i \in Y$, we have $|x_i| < |y_i|\ \forall\ i$.

Hence the string generated by sequence of substrings of X is shorter than the strings generated by sequence of corr. sub. strings of Y.

So, PCP has no solution.

(iii) $(10, 01, 0, 100, 1)$
$\quad\quad 1 \quad 2 \quad 3 \quad 4 \quad 5$

$(101, 100, 10, 0, 010)$

1 5 2 3 4 4 3 4

1	1	5	2	3	4	4	3	4
10	1	01	0	100	100	0	100	
101	101	010	100	10	0			

(iv) $A = (100, 0, 1)$, $B = (1, 100, 00)$
$\quad\quad\quad 1 \quad 2 \quad 3$

13 11 3 2 2

Programming techniques in Turing Machine :-

There are three programming techniques in TM

1. Storage in states
2. Multiple tracks or multi tape.
3. Subroutines.

2] Design DFA which accepts strings contains a's & b's having single a fol. by any no. of b's

3] Design DFA which accepts all strings contain 0's and 1's having either two consecutive 0's or two consecutive 1's.

5] Design DFA which accepts all strings that contain exactly one and two 0's over the alphabets $\Sigma = \{0,1\}$

4] Construct DFA for the following

(i) $(ab)^n$, $n \geq 2$, $\Sigma = \{0,1\}$

(ii) $(ab)^n$, $n \geq 0$, $\Sigma = \{0,1\}$

5] Design DFA which accepts all strings that starts with either 01 or 10.

Unit

15 - Math 24
16 - ATFL 24
17 - CO 22
18 - DAA 22
19 - GUI 29
20 - WT 22

1] what is FA with ex.
2] Give FA for $\emptyset$ & ϵ
3] Applications of FA
4] Compare NFA & DFA
5] Justify various symbols used in FA.
6] Design DFA for $a^m b^n$, $m,n \geq 1$
7] Give the significance of ϵ-transition.
8] Diff. transition & ϵ-transition.
9] Define ϵ-closure(q) with example
10] limitations of DFA.

Sahithi Karthika

Math - 28 24
ATFL - 23 22
DAA - 22 22
CO - 20 29
GUI - 26 22
WT - 17