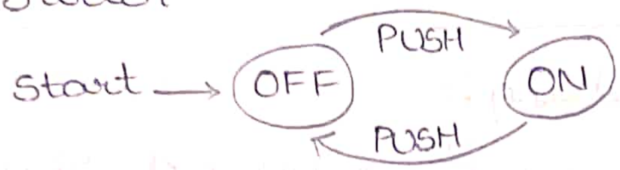
MODULE-1

* Finite automation modeling:-

On/off switch



•

---

* ALPHABET:- Finite, non-empty set of symbols.

Denoted by Σ

Eg:- Binary: $\Sigma = \{0,1\}$

lower case: $\Sigma = \{a, b, c, \dots, z\}$

Alphanumeric: $\Sigma = \{a\text{-}z, 0\text{-}9, A\text{-}Z\}$

• no use of special characters.

---

* String:- set of finite subsets.

Eg:- Alphabet $= \Sigma = \{0,1\}$

string $= \{0011, 10, 11, 111, \dots\}$

* String does not contain upper case letter.

---

* Length of String:-

u = string.

Then length $= |u|$.

Eg:- $u = 000111 \Rightarrow |u| = 6$

$u = abab \Rightarrow |u| = 4$

*Empty (or) Null String:- (ε / λ)

- String has no length.

$$|\varepsilon| = |\lambda| = 0$$

---

* Reverse of String:-

- u = string $\Rightarrow$ u^R = reverse of string.

Eg:- If $u = abc \Rightarrow u^R = cba$.

---

* Power 'm' of a string:-

Σ^m = Set of all string of length m.

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

$\Sigma^0 = \{\varepsilon\}$ $\quad\quad$ $\Sigma^2 = \{aa, bb, ab, ba\}$.

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

---

* Kleen Star (Closure):- Σ^*

The set include ε with all possible strings

$$\Sigma = \{0\} \Rightarrow \Sigma^* = \{\varepsilon, 0, 00, 000, \ldots\}$$

$$\Sigma = \{a, b\} \Rightarrow \Sigma^* = \{\varepsilon, a, b, ab, bc, \ldots\}$$

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

* $\Sigma^+ \Rightarrow$ does not has ε.

* $\varepsilon \cdot a = a$

* $\phi \cdot a = \phi$

* ϕ (Null) $\Rightarrow$ has no string.

* Concatenation:-

$u = a_1 a_2 \cdots a_i$ $v = b_1 b_2 \cdots b_j$

$uv = a_1 a_2 \cdots a_i b_1 b_2 \cdots b_j$

Length of $u \Rightarrow |u| = i$ Length of $uv \Rightarrow |uv| = i+j$.

Length of $v \Rightarrow |v| = j$

- $u \cdot \varepsilon = \varepsilon \cdot u = u$

- $w = uv \Rightarrow$ u = prefix of w
  v = suffix of w

Eg:- $w = 0111 \Rightarrow$ prefix $= \{\varepsilon, 0, 01, 011, 0111\}$
suffix $= \{\varepsilon, 1, 11, 111, 0111\}$.

---

* Palindrome:- $u = u^R$.

Eg:- madam, malayalam, amma.

* Properties:-

(i) $u(vw) = (uv)w$

(ii) $|uv| = |u| + |v|$

(iii) $wu = wv \Rightarrow u = v$ (~~Righ~~ Left cancellation)

(iv) $uw = vw \Rightarrow u = v$ (Right cancellation)

Eg:- $\Sigma = \{a, b\}$ $\Sigma^* = \{\varepsilon, a, b, aa, ab, ba, bb, \ldots\}$

$\{ab, abab, aabb\}$ ⇒ Finite langoge

$L = \{a^n b^n, n \geq 0\}$ ⇒ Infinite language

---

* L = Language over
  Subset of Σ^*

(i) complement of $L = \tilde{L} = \Sigma^* - L$.

(ii) inverse of $L = L^{-1} = \{W^R ; W \in L\}$.

(iii) Concatenation of L_1 & L_2 = $L_1 L_2 = \{uv, u \in L \,\&\, v \in L\}$

(iv) union of L_1 & L_2 = $L_1 \cup L_2$

(v) Intersection of L_1 & L_2 = $L_1 \cap L_2$

---

* **Graphs:-**

- graph denoted by $G = (V, E)$ vertices Edge
- path in a graph is sequence of vertices $V_1, V_2, V_3, \ldots, V_k$, $k \geq 1$ & edge (V_i, V_{i+1}) $1 \leq i < k$.

  length of path $= k - 1$

  path is cyclic ⇒ $V_1 = V_k$.

Formal Proof.

* Deductive proof
* Reduction to definitions
* Other theorem forms.

1] Formal proof / Derivation:-

* usage of ~~pro~~ predefined sentences and well defined formulaes.

2] Deductive proofs:-

* Based on statement we need to check & verify whether statement is valid / Invalid.
* Seq of statements ⇒ 2 parts
  (i) Initial statement / Hypothesis.
  (ii) Conclusion

- Based on Hyp conclusion will come.

Eg:- if $a>b$. ⇒ Hyp.
a big than b. ⇒ Conclusion.

* Steps:-
  (i) we need to accept logic
  (ii) given facts
  (iii) previous statements in deductive proof
  (iv) all the above

* Hyp may be true/False based on parameter

* Theorem proved from Hyp to conc ⇒ Statement

* $\boxed{\text{If H then C}}$ / C is deducted from H.

Deductive Proof :- Theorem I :

☆ If $X \geq 4$ then $2^x \geq X^2$

Hyp $H \Rightarrow X \geq 4$

H is based on value of X.

Here $X \geq 4 \Rightarrow X = 4, 5, 6, \ldots$

$x = 4 \Rightarrow 2^4 \geq 4^2 \Rightarrow 16 \geq 16.$

$x = 5 \Rightarrow 2^5 \geq 5^2 \Rightarrow 32 \geq 25$

∴ Hyp is True when $X = 4, 5, 6, \ldots$

Deductive Proof :- Theorem II.

☆ If $X = a^2 + b^2 + c^2 + d^2$ then $2^x \geq X^2$.

(i) $X = a^2 + b^2 + c^2 + d^2$ $(a, b, c, d \geq 1)$.

(ii) $a \geq 1, b \geq 1, c \geq 1, d \geq 1.$

(iii) $a^2 \geq 1, b^2 \geq 1, c^2 \geq 1, d^2 \geq 1$

(iv) $X = 1 + 1 + 1 + 1 = 4 \Rightarrow X \geq 4$

(v) $2^4 \geq 4^2 \Rightarrow 16 \geq 16.$

Eg:- $a = 2, b = 2, c = 2, d = 2$.

$X = 4 + 4 + 4 + 4 = 16.$

$2^{16} \geq 16^2 \Rightarrow 65536 \geq 256.$

3] Reduction to definition:-

Eg:- If S&T are both subsets of U, then

$\boxed{\text{T is complement of S}}$

$U = \{1, 2, 3, \ldots, 9\}$.

$S = \{1, 2, 3, 4, 5\} \Rightarrow T = \{6, 7, 8, 9\}$

T is complement of S.

* If $S \cup T = U \Rightarrow S \cup T$ = hyp. U = Concl.
* If $S \cap T = \phi \Rightarrow S \cap T$ = hyp. ϕ = Concl.
* S = finite subset of U → infinite set

T = complement of S.

T = infinite subset of U.

4] Other Theorem Forms:-

* if - then:-

H implies C

H only if C

C if H

whenever H holds, C follows.

---

* Theorem 1: If $\underset{H}{x \geq 4}$ then $\underset{C}{2^x \geq x^2}$.

$x \geq 4$ implies $2^x \geq x^2$.

$x \geq 4$ only if $2^x \geq x^2$

$2^x \geq x^2$ if $x \geq 4$.

* If and only if:-

A if and only if B
A iff B.
A is equivalent to B
A exactly when B.

"if A then B" & "if B then A".

---

* If H then C ⇒ conditional statement
* A if & only if B ⇒ Biconditional statem

* Prove "A if and only if B."

if ⇒ If B then A.

only if ⇒ If A then B.

Eg:- $\lfloor x \rfloor = \lceil x \rceil$
floor ceiling.
If and only if x is an integer.

* Theorems not appear to be if - then:-

Eg:- $\sin^2\theta + \cos^2\theta = 1$

---

* Equivalence about sets:-

$R \cup (S \cap T) = R \cup (S \cap T)$

S.No	Statement	Justification
1	x is in $R \cup (S \cap T)$	Given
2	x is in R or x is in $S \cap T$	Def in union
3	x is in R or x is in both S&T	Def in Intersecti
4	x is in $R \cup S$	
5	x is in $R \cup T$	
6	x is in $(R \cup S) \cap (R \cup T)$	

Steps in if - part of Theorem.

Eg:- $\Sigma = (1,2)$.

$L = \{1^m 2^n \mid m,n >= 0\}$

String $= \{1^1 2^2\} = \{1.2.2\}$

$\{1^4 2^3\} = \{1111222\}$

Eg:- $\Sigma = \{1,2\}$

$\Sigma^2 = \{11, 22, 12, 21\}$.

---

* Theorem 5:- $R \cup (S \cap T) = (R \cup S) \cap (R \cup T)$

• Steps in If-Part:-

1) X is in $R \cup (S \cap T)$	given.
2) X is in R or in $(S \cap T)$	def of union
3) X is in R or X is in both S and T	def of Intersection
4) X is in $R \cup S$.	def of union
5) X is in $R \cap T$	def of union.
6) X is in $(R \cup S) \cap (R \cup T)$	def of Intersection

• Steps in ~~else~~ only-if-Part:-

1) X is in $(R \cup S) \cap (R \cup T)$	Given
2) X is in $R \cup S$.	def of Intersection
3) X is in $R \cup T$	def of Intersection
4) X is in R or X is in both S and T	reasoning in union
5) X is in R or X is in $S \cap T$	def of Intersection.
6) X is in $R \cup (S \cap T)$	def of union.

* Contrapositive:-

"If H then C" $\Rightarrow$ "If not C then not H".

Eg:- If $x \geq 4$, then $2^x \geq x^2$.

H C.

Contrapositive $\Rightarrow$ If not $2^x \geq x^2$ then not $x \geq 4$.

[$\therefore$ not $a \geq b \Rightarrow a < b$]

$\Rightarrow$ If $2^x < x^2$ then $x < 4$.

---

* Counter Examples:-

- A Δ^{le} is equi iff its angles all measure 60°.
- I love you and I didn't love you.
- If no. divisible by two is even.

---

* Inductive Proof:-

- Essential when dealing with recursively defined objects.
- Induction of Integers – Mathematical Induction:

  Statement S(n) about an integer n.

  There are 2 types:- 1) Basis step

  2) Inductive step

1) **Basis**:- Substitute $S(i) \Rightarrow i=0$ (or) 1

2) **Inductive**:- we should assume the algo will work for $(K+1)^{th}$ terms.

Eg:- $1+2+3+\cdots+n = \frac{n(n+1)}{2}$.

Sol:- $P(n) = 1+2+3+\cdots+n = n(n+1)/2$.

Basis step:-

$n=1 \Rightarrow$ LHS $=1$

RHS $=1$

Inductive step:-

Assume $n=k$

$1+2+3+\cdots+K = \frac{K(K+1)}{2}$. (Here we assume algo also works for $(K+1)^{th}$ term also)

we have to prove result for $n=K+1$.

$$1+2+3+\cdots+K+K+1 = \frac{(K+1)(K+2)}{2}.$$

LHS $\Rightarrow$ $[1+2+3+\cdots+K]+K+1$.

$\Rightarrow \frac{K(K+1)}{2} + K+1.$

$\Rightarrow (K^2 + K + 2K + 1)/2$

$$\Rightarrow \frac{K^2 + 3K + 1}{2}$$

$$= \frac{(K+1)(K+2)}{2}$$

$= RHS.$

$\therefore$ Hence proved.

Eg:- $\sum_{i=0}^{1} i^2 = \frac{n(n+1)(2n+1)}{6}$

$p(n): 1^2 + 2^2 + 3^2 + \text{---} + n^2 = \frac{n(n+1)(2n+1)}{6}$

Basis step:- $n = 1$

$LHS \Rightarrow n^2 = 1$

$RHS \Rightarrow 1(2)(3)/6 = 1$

$LHS = RHS.$

Inductive Step:-

$p(n): 1^2 + 2^2 + 3^2 + \text{---} + K^2 + (K+1)^2 = \frac{(K+1)(K+2)(2K+3)}{6}$

$LHS \Rightarrow [1^2 + 2^2 + 3^2 + \text{---} + K^2] + (K+1)^2.$

$$\Rightarrow \frac{K(K+1)(2K+1)}{6} + (K+1)^2.$$

$$\Rightarrow \frac{(2K^3 + 3K^2 + K) + 6(K^2 + 2K + 1)}{6}$$

$$\Rightarrow \frac{2K^3 + 3K^2 + K + 6K^2 + 12K + 6}{6}$$

$$\Rightarrow \frac{2K^3 + 9K^2 + 13K + 6}{6}.$$

$$\Rightarrow \frac{(K+1)(K+2)(2K+3)}{6}$$

$\Rightarrow RHS.$

$\therefore$ Hence Proved.

Finite Automata

- It checks whether the string is valid/not.
- Finite Automata → Deterministic F.A
  ↘ Non-Deterministic F.A.

~~DNFA~~

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

* DFA:-

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

  Q : Set of finite states.

  Σ = Alpha (~~Set~~ finite set of input symbols)

  q_0 = ini state.

  δ = Transition Function

  F = Accepting state $[F \subseteq Q]$ (Value in ◎)

Eg:- $\delta : Q \to \Sigma \to Q$

I am in a state moving with i/p ~~Sigma~~ Sym Σ to other state in Q.

→ Rules:-

1) NFA/DFA machine should have only one ini state
2) DFA is considered it has one accepting state
3) δ Transition ⇒ DFA should not have null value
4) In DFA, Trans Table does not consist of null space
5) DFA $\subseteq$ NFA.

* Notations for NFA/DFA:-

○ ⇒ States

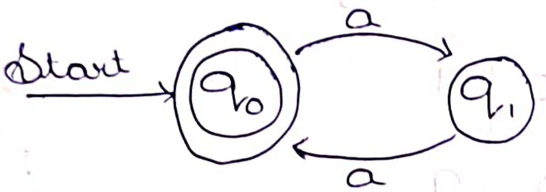
◎ q_1 ⇒ Final state

start → q_0 ⇒ Ini state

Prob-1 Def a DFA that accepts a string having even no. of a's.

Sol:- $S = \{\varepsilon, aa, aaaa, \cdots\}$

Draw the machine:- $\boxed{\text{DFA}}$



} Transition Diagram.

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

Q = Set of finite state = $\{q_0, q_1\}$

Σ = Alphabet = $\{a\}$

$q_0 = \{q_0\}$

F = Accepting state = $\{q_0\}$

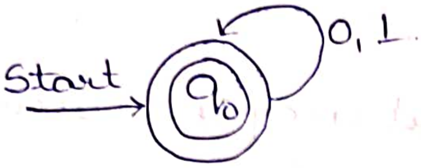
δ	a
$*\rightarrow q_0$	q_1
q_1	q_0

} Transition Table

Prob:-] Define DFA accepts string of 0's & 1's.

Sol:-] String = $\{0, 1, 00, 01, 111, 000, \varepsilon, \ldots\}$.

Diagram:-



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

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

$\Sigma^+ = \{0, 1\}$ $\quad$ $F = \{q_0\}$.

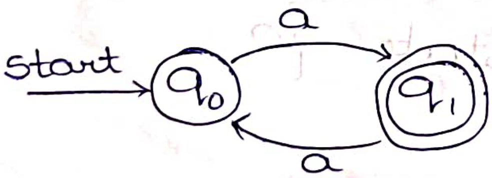
δ- Table:-

δ	0	1
$\rightarrow * q_0$	q_0	q_0

Prob:-] Define PDFA accepts string of odd a's.

Sol:-] String = $\{\varepsilon, a, aaa, aaaaa, \ldots\}$

Diagram:-



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

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

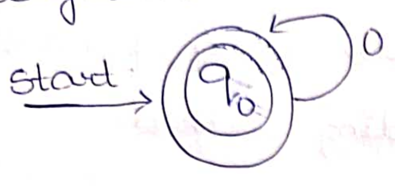
$\Sigma = \{a\}$ $\quad$ $F = \{q_1\}$

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

Prob:- Def a DFA that accepts the string of "zero"

Sol:- String = $\{\varepsilon, 0, 00, 000, ----\}$

Diagram:-



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

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

$\Sigma = \{0\}$ $\quad$ $F = \{q_0\}$

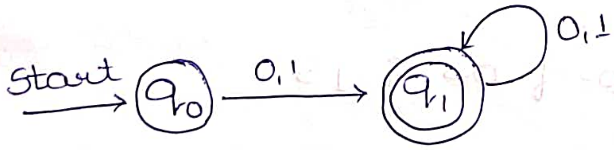
δ-Table:-

δ	0
$* \rightarrow q_0$	q_0.

Prob:- Define DFA that accepts any String of any length of 0's & 1's which has to begin with 0 or 1

Sol:- String = $\{0, 1, 011, 100, ---\}$.

Diagram:-



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

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

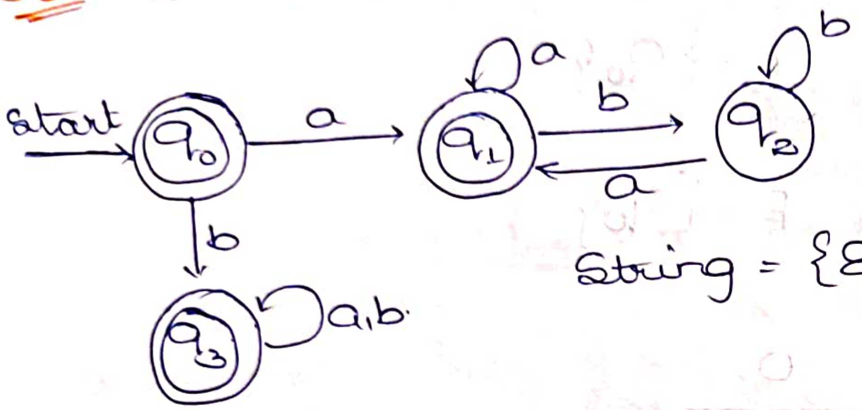
$\Sigma = \{0, 1\}$ $\quad$ $F = \{q_1\}$

δ-Table:-

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

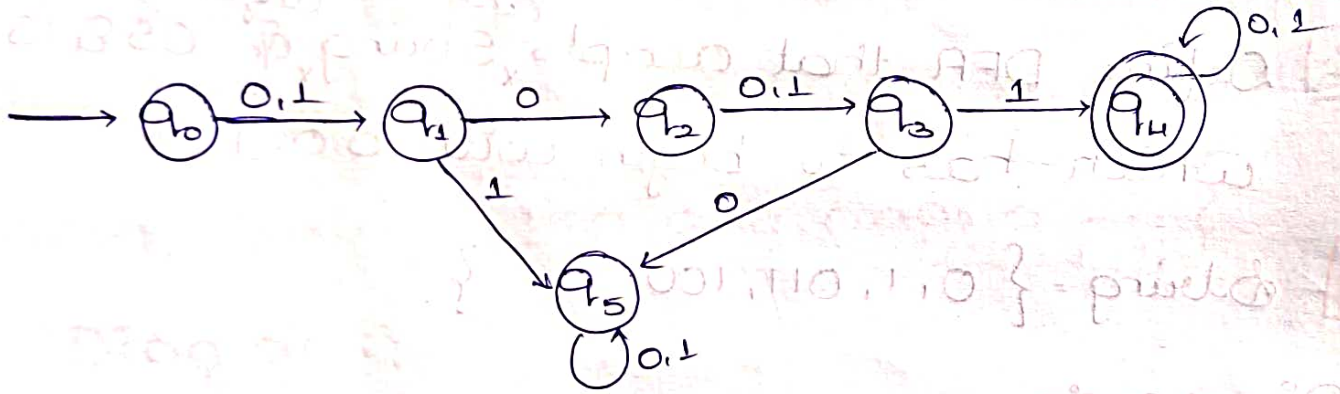
Prob:-] Const DFA that accepts lang of all strings not starting with a (or) not ending with 'b'.

Sol:- $(A \cup B)^c = A^c \cap B^c$

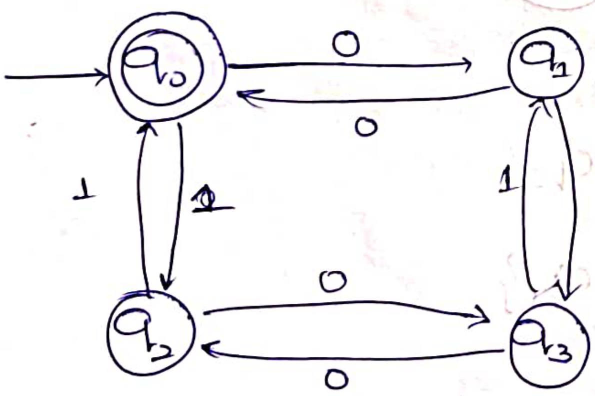


String = {ε, a, b, aba, ba ---}

Prob:-] DFA for second = 0 & fourth = 1.

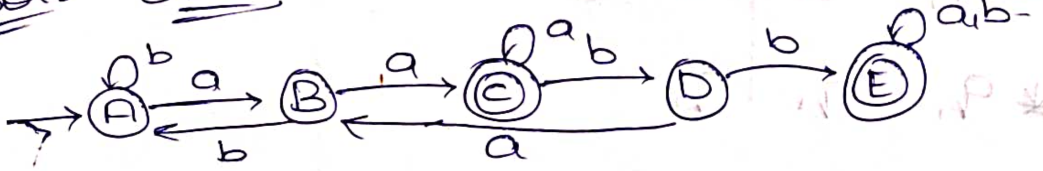


Prob:-] Even no. of 0's & 1's.



Prob:-] Does not contain aabb.

Sol:- aabb

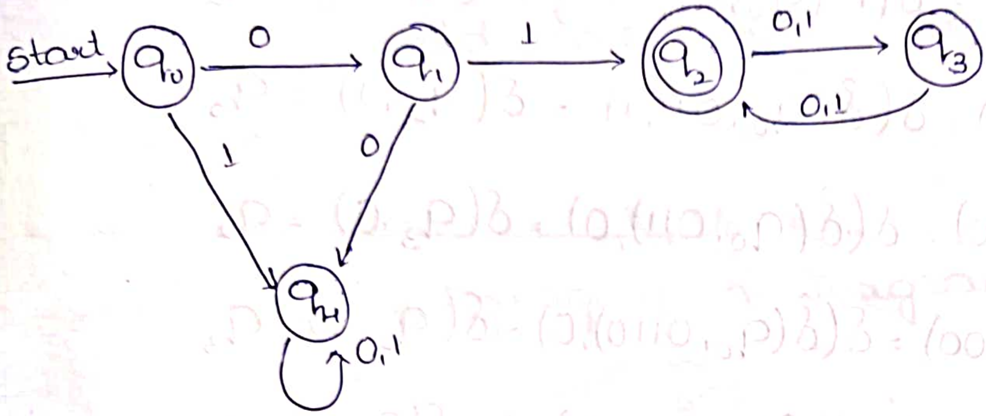


[Make Final as non Final states]

Prob:-] Def a DFA accepts $L = \{w \mid w$ is even length and start with $01\}$

Sol:-] String $= \{01, 0111, 0110, 0101, \ldots\}$

Diagram:-



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

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

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

δ- Table:-

δ	0	1
$\rightarrow q_0$	q_1	q_4
q_1	q_4	q_2
$* q_2$	q_3	q_3
q_3	q_2	q_2
q_4	q_4	q_4

Machine Validation:-Extended Transition Func:- $\hat{\delta}$.

we have varify a string whetur its work for that machine.

$\omega = 011001 \Rightarrow \hat{\delta}(q_0, 011001)$

$\hat{\delta}(q_0, \varepsilon) = q_0.$

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

$\hat{\delta}(q_0, 01) = \delta(\hat{\delta}(q_0, 0), 1) = \delta(q_1, 1) = q_2$

$\hat{\delta}(q_0, 011) = \delta(\hat{\delta}(q_0, 01), 1) = \delta(q_2, 1) = q_3$

$\hat{\delta}(q_0, 0110) = \delta(\hat{\delta}(q_0, 011), 0) = \delta(q_3, 0) = q_2$

$\hat{\delta}(q_0, 01100) = \delta(\hat{\delta}(q_0, 0110), 0) = \delta(q_2, 0) = q_3$

$\hat{\delta}(q_0, 011001) = \delta(\hat{\delta}(q_0, 01100), 1) = \delta(q_3, 1) = q_2$

$\therefore$ Here we get q_2 which is final state

$\therefore$ $\omega = 011001 \Rightarrow$ valid string.

Eg: $\omega = 011.$

$\hat{\delta}(q_0, 011).$

$\hat{\delta}(q_0, \varepsilon) = q_0.$

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

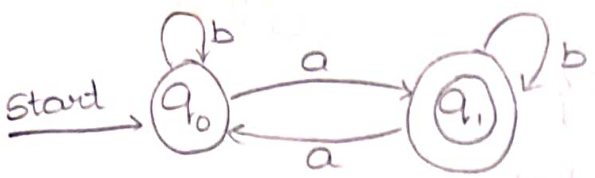
$\hat{\delta}(q_0, 01) = \delta(\hat{\delta}(q_0, 0), 1) = \delta(q_1, 1) = q_2$

$\hat{\delta}(q_0, 011) = \delta(\hat{\delta}(q_0, 01), 1) = \delta(q_{\cancel{0}2}, 1) = q_3.$

$\therefore$ Here we get $q_3 \neq F$

$\therefore$ $\omega = 011$ is not valid string

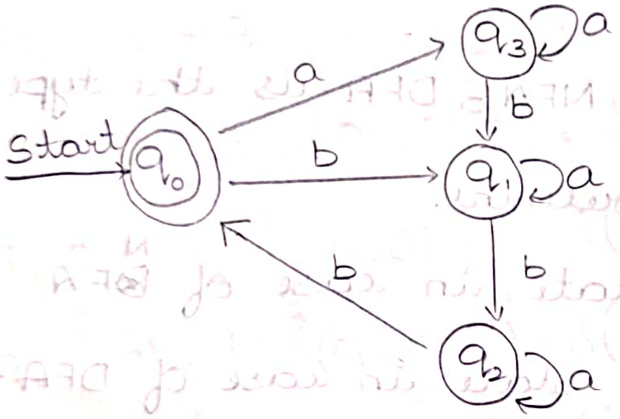
Prob-9] Find Language of δ-Diagram.



Sol:-] $L = \{w \mid w$ is set of string having odd no. of a's.$\}$

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

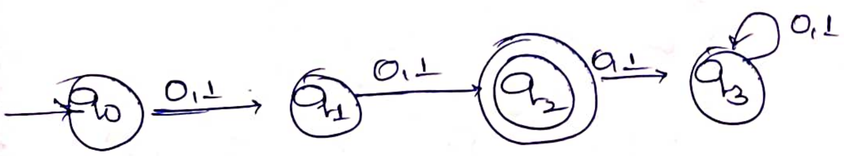
Prob:-10] Find language of δ-Diagram.



Sol:-] $L = \{w \mid w$ is a set of string of any length of a,b having no. of b's is multiple of 3$\}$

Prob:-] Ending with 'ab'.

Prob:-] {0,1} of length 2.



Non-Deterministic Finite Automata (NFA)

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

  Q: finite set of states

  Σ: finite set of input symbol

  q_0 = Initial state

  F = accepting state $F \subseteq Q$.

  $\delta : Q \times \Sigma \longrightarrow Q$

* only diff b/w NFA & DFA is the type of value that returns.

  δ: A set of state in case of ~~D~~NFA and single state in case of DFA.

Prob:-1]

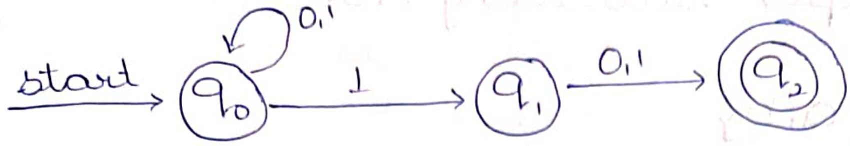
δ	0	1
$\rightarrow q_0$	q_0	$q_0 q_1$
q_1	q_2	q_2
$* q_2$	ϕ	ϕ

Draw δ-Diagram and machine Def?

Sol:-]

$$M = \{\underset{Q}{\{q_0, q_1, q_2\}}, \underset{\Sigma}{\{a, b\}}, \underset{\delta}{\delta}, \underset{q_0}{\{q_0\}}, \underset{F}{\{q_2\}}\}$$

Transition Diagram:-



Language:- $L = \{w \mid w$ has string of any length ending with 10 or 11$\}$

verification:- $w = 01010$

$\hat{\delta}(q_0, 01010) = \{ \quad \} \cap F \neq \phi.$

$\hat{\delta}(q_0, \varepsilon) = q_0.$

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

$\hat{\delta}(q_0, 01) = \delta(\hat{\delta}(q_0, 0), 1) = \delta(q_0, 1) = \{q_0, q_1\}$

$\hat{\delta}(q_0, 010) = \delta(\hat{\delta}(q_0, 01), 0) = \delta(\{q_0, q_1\}, 0)$

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

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

$\hat{\delta}(q_0, 0101) = \delta(\hat{\delta}(q_0, 010), 1) = \delta(\{q_0, q_2\}, 1)$

$= \delta(q_0, 1) \cup \delta(q_2, 1)$

$= \{q_0, q_1, \cancel{\phi}\}$

$\hat{\delta}(q_0, 01010) = \delta(\hat{\delta}(q_0, 0101), 0) = \delta(\{q_0, q_1, \cancel{\phi}\}, 0)$

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

$= \cancel{\phi}$

$\Rightarrow \{q_0, q_2\}$

$\therefore \hat{\delta}(q_0, 01010) = \{q_0, q_2\} \cap F$

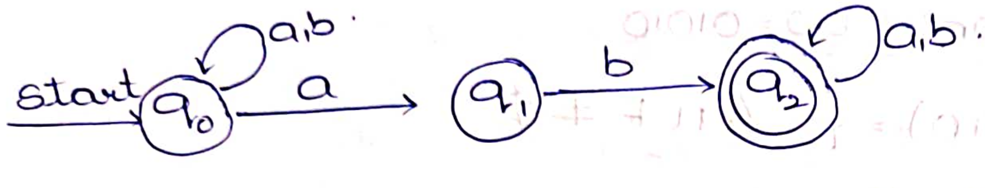
$= q_2 \neq \phi.$

Prob-2] Construct NFA over an alphabet (a,b) that accepts substring "ab".

Sol:-] $\Sigma = \{a, b\}$.

string :- {aab, aabbb, ab, bbabb, ...}

Diagram:-



$M = \{\{q_0, q_1, q_2\}, \{a,b\}, \{\delta, \{q_0\}, \{q_2\}\}$

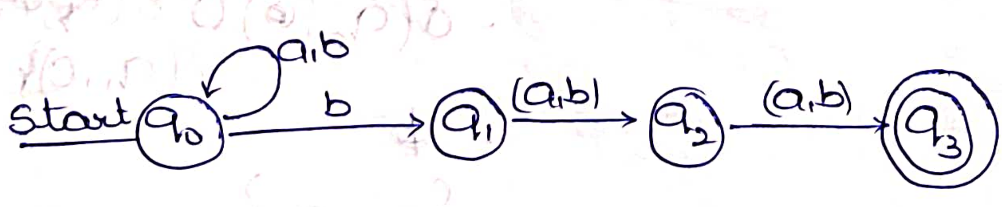
δ- Table:-

δ	a	b
q_0	q_0, q_1	q_0.
q_1	ϕ	q_2
q_2	q_2	q_2

Prob:-3] Cons NFA accepts strings which has 3rd symbol 'B' from right.

Sol:-] $\Sigma = \{a, b\}$

string = {baa; aabab;



$M = \{\{q_0, q_1, q_2, q_3\}, \{a,b\}, \{q_0\}, \{q_3\}\}$.

δ-Table:-

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

validation:- $\omega = abab$.

$\hat{\delta}(q_0, abab) = \{ \quad \} \cap F \neq \phi \Rightarrow$ valid.

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

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

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

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

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

$= \{q_0, q_2\}$

$\hat{\delta}(q_0, abab) = \delta(\hat{\delta}(q_0, aba), a) = \delta(\{q_0, q_2\}, a)$

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

$= \{q_0, q_3\}$

$\therefore \hat{\delta}(q_0, abab) = \{q_0, q_3\} \cap F$

$\neq \phi \Rightarrow$ valid.

* Construction of DFA equivalence to NFA:-

- The equivalence of DFA has multiple/null state
- ① They will give NFA ⇒ M input.

  ② output ⇒ DFA ⇒ M' output. (Equi of NFA)

- Similarly the prop may also change

$$\left.\begin{array}{l} M = \{Q, \Sigma, \delta, q_0, F\} \\ M' = \{Q', \Sigma, \delta', q_0, F'\} \end{array}\right\} q_0, \Sigma \text{ will not find.}$$

Eg:- Cons an DFA; equivalent to NFA

NFA ⟶ DFA.

$M = \{\{q_0, q_1\}, \{0, 1\}, \delta, \{q_0\}, \{q_1\}\}$

where

δ:	0	1
$\rightarrow q_0$	$q_0 q_1$	q_1
$* q_1$	–	$q_0 q_1$

Sol:- Given NFA we have to find DFA

M = NFA $\quad M'$ = DFA

$M = \{\{q_0, q_1\}, \{0, 1\}, \delta, \{q_0\}, \{q_1\}\}$

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

Q' = Set of all subsets of Q except null state

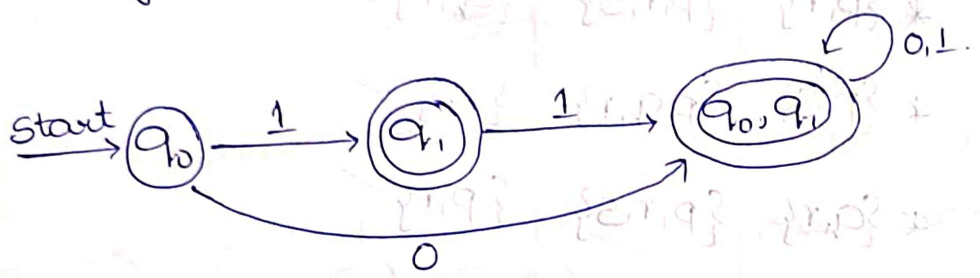
$Q' = \{\{q_0\}, \{q_1\}, \{q_0, q_1\}\}$

δ':	0	1
$\rightarrow \{q_0\}$	$\{q_0, q_1\}$	$\{q_1\}$
$*\{q_1\}$	–	$\{q_0, q_1\}$
$*\{q_0, q_1\}$	$\{q_0, q_1\}$	$\{q_1, q_0\}$

$\Downarrow$

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

δ- Diagram:-



---

* NFA to DFA conversion:-

NFA $\Downarrow$ M   DFA $\Downarrow$ M'.

Prob:-2] Convert to a DFA the following NFA.

NFA $\longrightarrow$ DFA

δ:	0	1
$\rightarrow p$	$\{p, r\}$	$\{q\}$
q	$\{r, s\}$	$\{p\}$
$* r$	$\{p, s\}$	$\{r\}$
$* s$	$\{q, r\}$	$\emptyset$

NFA

Step:-1]

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

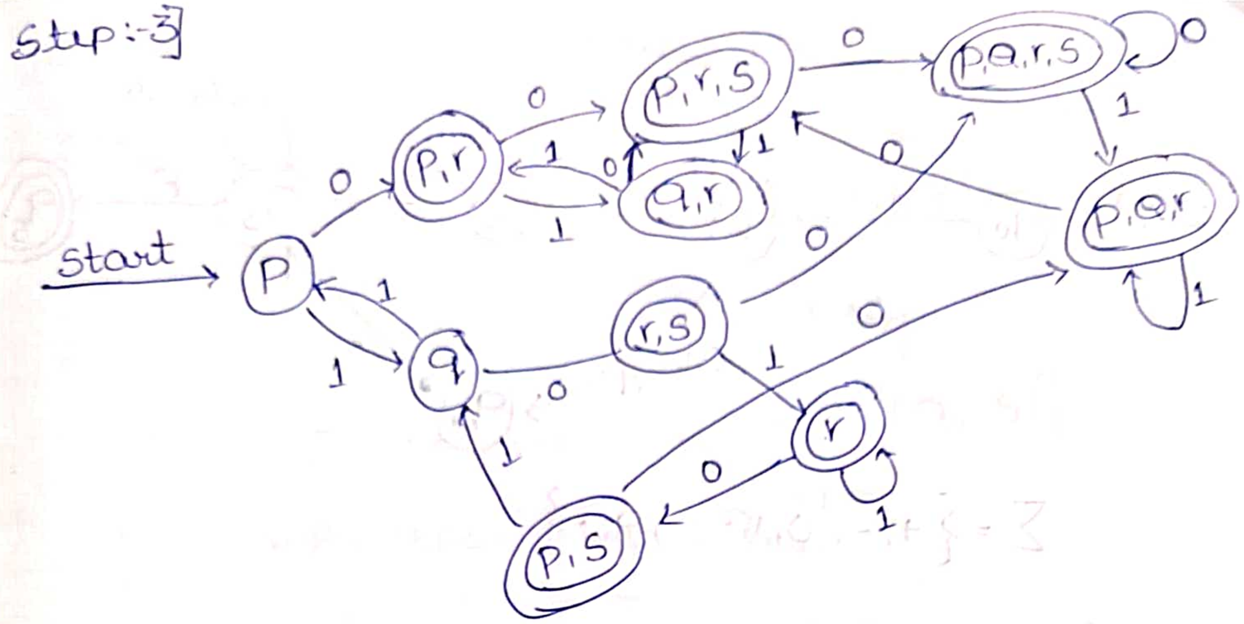
$Q' = \{ p, q, r, s, \{p, q\}, \{p, r\}, \{p, s\}, \{q, r\}, \{q, s\},$
$\{r, s\}, \{p, q, r\}, \{p, q, s\}, \{p, r, s\},$
$\{q, r, s\}, \{p, q, r, s\} \}$

Step:-2]

δ':-

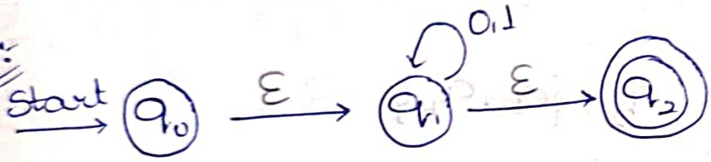
δ'	0	1
$\rightarrow$ p	{p, r}	{q}
Q	{r, s}	{p}
* r	{p, s}	{r}
* s	{q, r}	{ϕ}
{p, q}	{p, r, s}	{p, q}
* {p, r}	{p, r, s}	{q, r}
* {p, s}	{p, q, r}	{q}
* {q, r}	{p, r, s}	{p, r}
* {q, s}	{q, r, s}	{p}
* {r, s}	{p, q, r, s}	{r}
* {p, Q, r}	{p, r, s}	{p, q, r}
* {p, q, s}	{p, q, r, s}	{p, Q}
* {p, r, s}	{p, q, r, s}	{q, r}
* {Q, r, s}	{p, q, r, s}	{p, r}
* {p, q, r, s}	{p, q, r, s}	{p, q, r}

Step:-3]



E-Transition:-

Eg:



Travelling from one state to another without any input

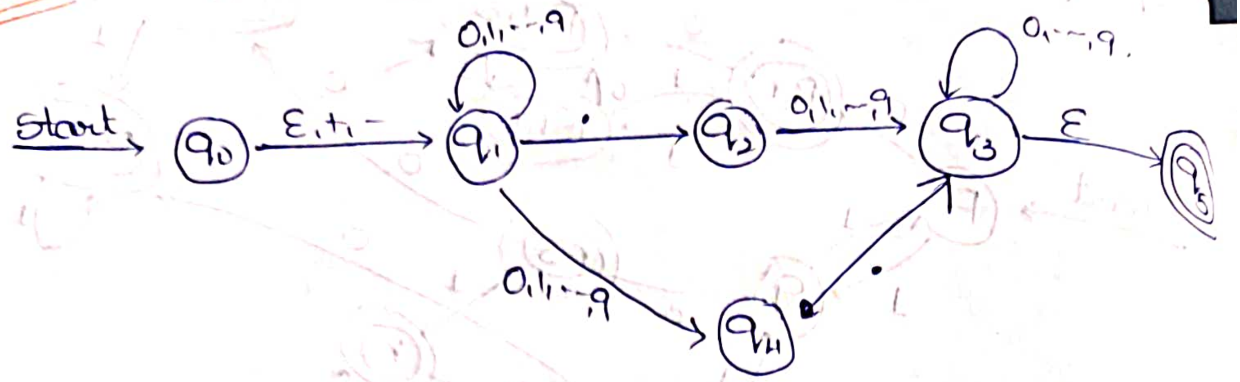
Eg: Design machine that accepts deci no. with following conditions.

Sol:-

1. An optional + or - sign
2. A string of digits
3. A decimal pt.
4. Another string of digits.

Cond:- Either this string of digits, or the string (ii) can be empty, but atleast one of 2 strings of digits must be non-empty.

Sol:-



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

δ:	ε	+	-	.	0, 1, ..., 9
$\rightarrow q_0$	q_1	q_1	q_1	-	-
q_1	-	-	-	q_2	$\{q_1, q_4\}$
q_2	-	-	-	-	q_3
q_3	q_5	-	-	q_4	q_3
q_4	-	-	-	-	-
$* q_5$	-	-	-	-	-

ε-closure:- (ε-closure):- If transition diagram has ε then we have to find ε-closure for each & every state. Used to valid a string And convertion of $\boxed{\text{ε-NFA} \longrightarrow \text{NFA}}$

ε-closure$(q_0) = \{q_0, q_1\}$

ε-closure$(q_2) = \{q_2\}$

ε-closure$(q_3) = \{q_3, q_5\}$

ε-closure$(q_1) = \{q_1\}$

ε-closure$(q_4) = \{q_4\}$

ε-closure$(q_5) = \{q_5\}$

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

$$\hat{\delta}(q_0, 5) = [\hat{\delta}(q_0, 5)]$$

$$= \varepsilon\text{- closure}(\hat{\delta}(q_0, 5))$$

$$= \varepsilon\text{- closure}[\delta(\hat{\delta}(q_0, \varepsilon), 5)]$$

$$= \varepsilon\text{- closure}[\delta(q_0, 5) \cup \delta(q_1, 5)]$$

$$= \varepsilon\text{- closure}[\{q_1, q_4\}]$$

$$= \{q_1, q_4\}.$$

$$\hat{\delta}(q_0\ 5\cdot) = [\delta(q_0 5\cdot)]$$

$$= \varepsilon\text{- closure}(\hat{\delta}(q_0\ 5\cdot))$$

$$= \varepsilon\text{- closure}(\delta[\hat{\delta}(q_0, 5), \cdot)]$$

$$= \varepsilon\text{- closure}(\delta(q_1, q_4), \cdot]$$

$$= \varepsilon\text{- closure}[\delta(q_1, \cdot) \cup \delta(q_4, \cdot)]$$

$$= \varepsilon\text{- closure}[q_2, q_3]$$

$$= \{q_2, q_3, q_5\}$$

$$\hat{\delta}(q_0, 5\cdot 6) = \varepsilon\text{- closure}(\hat{\delta}(q_0, 5\cdot 6))$$

$$= \varepsilon\text{- closure}(\delta(\hat{\delta}(q_0, 5\cdot), 6)$$

$$= \varepsilon\text{- closure}(\delta(q_2, q_3, q_5), 6)$$

$$= \varepsilon\text{- closure}(\delta(q_2, 6) \cup \delta(q_3, 6) \cup \delta(q_5, 6)]$$

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

$$= \{q_3, q_5\}$$

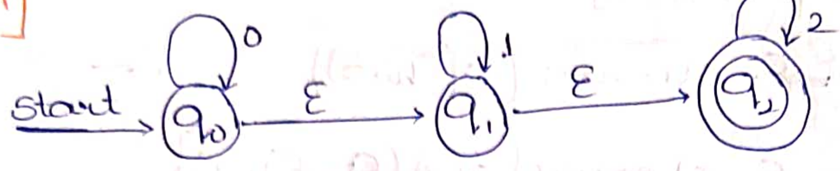
$$\therefore \hat{\delta}(q_0, 5\cdot 6) = \{q_3, q_5\} \cap q_5$$

$$\neq \phi$$

$\therefore$ String valid.

Converstion of ε-NFA $\longrightarrow$ NFA:-

Prob:-1]



Convert ε-NFA $\rightarrow$ NFA

Sol:-] $M = \varepsilon\text{-NFA}$.

Step-1] $M' = \text{NFA} = \{Q, \Sigma, \delta', q_0, F'\}$

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

Step-2] $\varepsilon\text{-closure}(q_1) = \{q_1, q_2\}$

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

Step-3] $F' = \{q_0, q_2\}$ $\Rightarrow$ when ε-closure of ini state has final state. Then q_0 (ini state) also becomes final state.

Step-4] δ':

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

$\cancel{\varepsilon\text{-closure}(\hat{\delta}(q_0,1))}$

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

$\cancel{\varepsilon\text{-closure}(\hat{\delta}(q_0,2))}$

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

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

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

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

$\Rightarrow \varepsilon\text{-closure}(q_1) \Rightarrow \boxed{\{q_1, q_2\}}$

$\varepsilon\text{-closure}(\hat{\delta}(q_0,2))$

$\Rightarrow \varepsilon\text{-closure}(\delta(q_0,2) \cup \delta(q_1,2) \cup (q_2,2))$

$\Rightarrow \varepsilon\text{-closure}(q_2) = \boxed{\{q_2\}}$

ε-closure$(\hat{\delta}(q_1,0))$

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

$\Rightarrow \varepsilon$-closure$(\delta(q_1,q_2),0)$

$\Rightarrow \varepsilon$-closure$(\delta(q_1,0) \cup \delta(q_2,0)) = \boxed{\phi}$

ε-closure$(\hat{\delta}(q_1,1))$

$\Rightarrow \varepsilon$-closure$(\delta(q_1,1) \cup \delta(q_2,1))$

$\Rightarrow \varepsilon$-closure$(q_1) = \boxed{\{q_1,q_2\}}$

ε-closure$(\hat{\delta}(q_1,2))$

$\Rightarrow \varepsilon$-closure$(\delta(q_1,2) \cup \delta(q_2,2))$

$\Rightarrow \varepsilon$-closure$(q_2) = \boxed{\{q_2\}}$.

ε-closure$(\hat{\delta}(q_2,0))$

$\Rightarrow \varepsilon$-closure$(\delta(q_2,0)) = \boxed{\phi}$

ε-closure$(\hat{\delta}(q_2,1))$

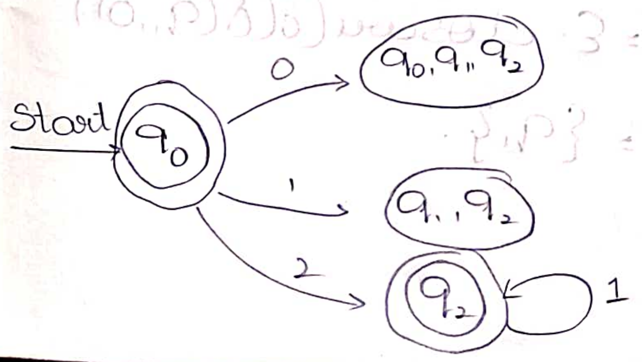
$\Rightarrow \varepsilon$-closure$(\delta(q_2,1)) = \boxed{\phi}$

ε-closure$(\hat{\delta}(q_2,2))$

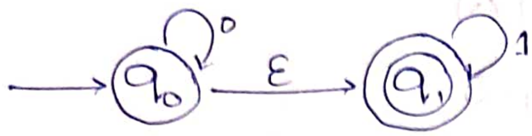
$\Rightarrow \varepsilon$-closure$(q_2) = \boxed{\{q_2\}}$

δ' - Table.

δ	0	1	2
$\rightarrow * q_0$	$\{q_0,q_1,q_2\}$	$\{q_1,q_2\}$	q_2
q_1	—	$\{q_1,q_2\}$	q_2
$* q_2$	—	—	q_2



Prob:-2] Convert ε-NFA $\rightarrow$ NFA



Sol:-] $M = \varepsilon$-NFA

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

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

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

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

δ' :-

$$\begin{aligned}
\varepsilon\text{-closure}(\hat{\delta}(q_0, 0)) &= \varepsilon\text{-closure}(\delta(\hat{\delta}(q_0), 0)) \\
&= \varepsilon\text{-closure}(\delta(q_0, 0)) \\
&= \varepsilon\text{-closure}(q_0) \\
&= \{q_0, q_1\}
\end{aligned}$$

$$\begin{aligned}
\varepsilon\text{-closure}(\hat{\delta}(q_0, 1)) &= \varepsilon\text{-closure}(\delta(\hat{\delta}(q_0, q_1), 1)) \\
&= \varepsilon\text{-closure}(\delta(q_0, 1) \cup \delta(q_1, 1)) \\
&= \varepsilon\text{-closure}(q_1) \\
&= \{q_1\}
\end{aligned}$$

prq

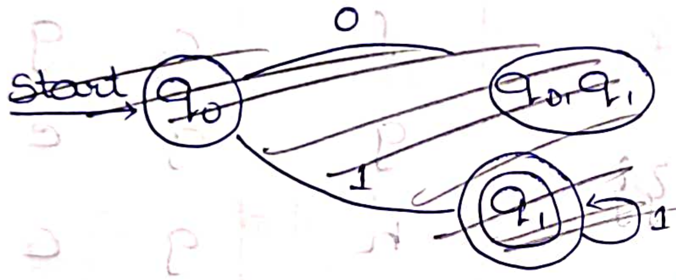
$$\begin{aligned}
\varepsilon\text{-closure}(\hat{\delta}(q_1, 0)) &= \varepsilon\text{-closure}(\delta(\hat{\delta}(q_1, \varepsilon), 0)) \\
&= \varepsilon\text{-closure}(\delta(q_1, 0)) \\
&= \phi
\end{aligned}$$

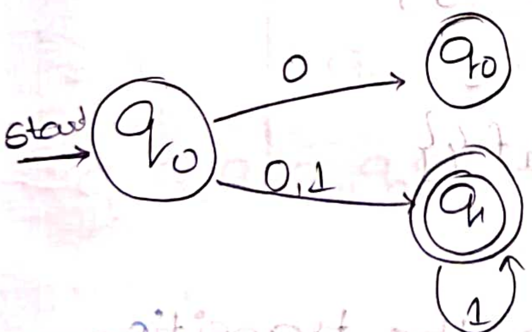
$$\begin{aligned}
\varepsilon\text{-closure}(\hat{\delta}(q_1, 1)) &= \varepsilon\text{-closure}(\delta(\hat{\delta}(q_1, 0))) \\
&= \{q_1\}.
\end{aligned}$$

δ- Table:

δ	0	1
$\rightarrow q_0$	$\{q_0, q_1\}$	$\{q_1\}$
$* q_1$	$\{\phi\}$	$\{q_1\}$

Transition diagram:-





Minimization of DFA:-

Prob:-1] Min the 8 states to 5 state by removing of repeated string.

δ/States	0 (i/p)	1 (i/p)
→a	b	f
b	g	c
*c	a	c
d	c	g
e	h	f
f	c	g
g	g	e
h	g	c

Sol:-] $S_0 = \{\underbrace{\{c\}}_{Q_1}, \underbrace{\{a,b,d,e,f,g,h\}}_{Q_2}\}$ (state)

$S_1 = \{\{c\},\{a,e,g\},\{b,d,f,h\}\}$

$= \{\underbrace{\{c\}}_{Q_1}, \underbrace{\{a,e,g\}}_{Q_2}, \underbrace{\{b,h\}}_{Q_3}, \underbrace{\{d,f\}}_{Q_4}\}$

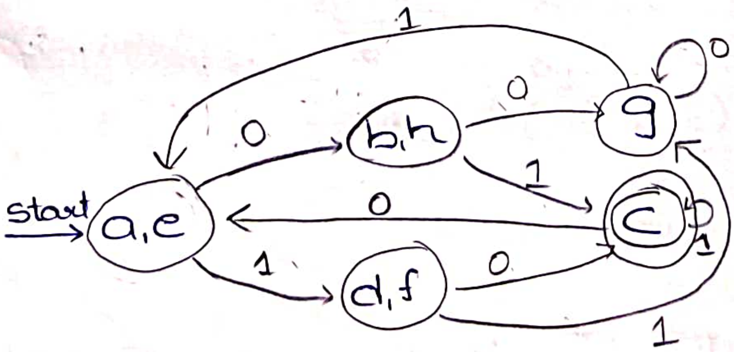
$S_2 = \{\underbrace{\{c\}}_{Q_1}, \underbrace{\{a,e\}}_{Q_2}, \underbrace{\{g\}}_{Q_3}, \underbrace{\{b,h\}}_{Q_4}, \underbrace{\{d,f\}}_{Q_5}\}$

$S_3 = \{\underbrace{\{c\}}_{Q_1}, \underbrace{\{a,e\}}_{Q_2}, \underbrace{\{g\}}_{Q_3}, \underbrace{\{b,h\}}_{Q_4}, \underbrace{\{d,f\}}_{Q_5}\}$

$S_3 = S_2 \Rightarrow$ we can stop the transition.

∴ 8 states converted into 5 states

δ	0	1
→[a,e]	[b,h]	[d,f]
*[c]	[a,e]	[c]
[g]	[g]	[a,e]
[b,h]	[g]	[c]
[d,f]	[c]	[g]



Prob :- 2] Min following DFA

δ/State	a	b
$\rightarrow q_0$	q_1	q_3
q_1	q_2	q_4
q_2	q_1	q_4
q_3	q_2	q_4
$* q_4$	q_4	q_4

Sol:-] $S_0 = \{ \underbrace{\{q_4\}}_{Q_1}, \underbrace{\{q_0, q_1, q_2, q_3\}}_{Q_2} \}$

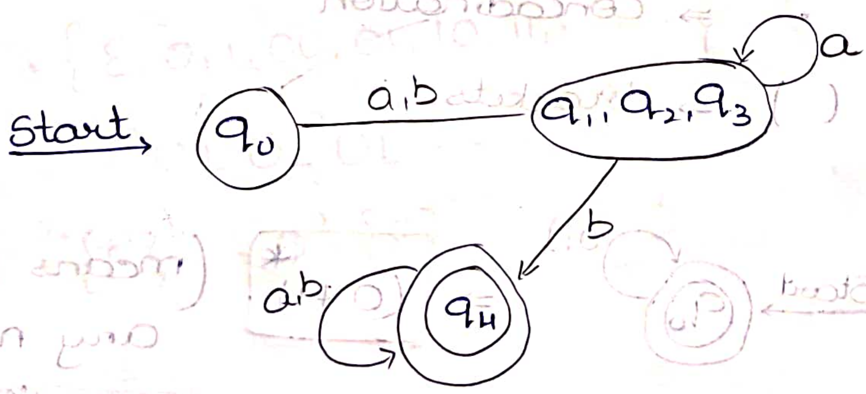
$S_1 = \{\{q_4\}, \{q_0\}, \{q_1, q_2, q_3\}\}$

$= \{ \underbrace{\{q_4\}}_{Q_1}, \underbrace{\{q_0\}}_{Q_2}, \underbrace{\{q_1, q_2, q_3\}}_{Q_3} \}$

$S_2 = \{ \underbrace{\{q_4\}}_{Q_1}, \underbrace{\{q_0\}}_{Q_2}, \underbrace{\{q_1, q_2, q_3\}}_{Q_3} \}$

$S_2 = S_1 \Rightarrow$ Total 3 states.

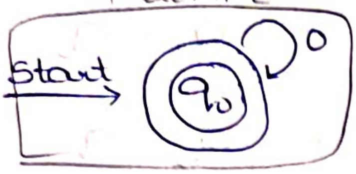
δ	a ~~b~~	b
$\rightarrow q_0$	q_1, q_2, q_3	q_1, q_2, q_3
$* q_4$	q_4	q_4
q_1, q_2, q_3	q_1, q_2, q_3	q_4



MODULE - 2.

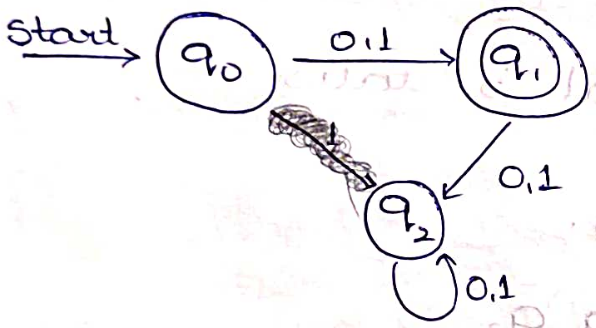
REGULAR EXPRESSIONS. (RE)

Machine Notation

Eg:- 

$\Rightarrow$ $\boxed{RE = 0^*}$ means any no. of zero.

Eg:- A machine can produce either 0|1.



$\boxed{RE = 0+1}$ means. 0|1.

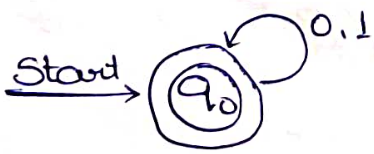
Operators :-

$+$ $\Rightarrow$ addition (union)

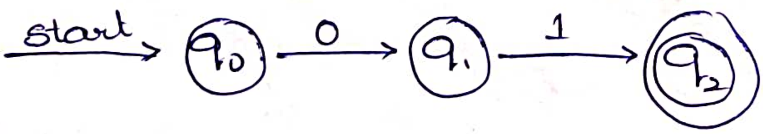
$*$ $\Rightarrow$ mul (repeatation)

$\cdot$ $\Rightarrow$ concadination

( ) $\Rightarrow$ brackets

Eg:- 

$\Rightarrow$ $\boxed{(0+1)^*}$ (means string of any no. of 0's or 1's)

Eg:- 

$\Rightarrow$ $\boxed{RE = 01}$

Topics:- 1) How to construct a regular exp

2) DFA $\longrightarrow$ R.E. $\boxed{R_{ij}^{k}}$

3) RE $\longrightarrow$ ε-NFA

4) Pumping lemma (verify language)

Explanation of operators:-

(i) Union :- $L \cup M$.

$L = \{001, 01, 11\}$ $\quad M = \{\varepsilon, 10\}$

$L \cup M = \{\varepsilon, 10, 01, 11, 001\}$

(ii) Concatenation :- $L \cdot M$. $\quad L = \{001, 10, 11\}$ $M = \{\varepsilon, 01\}$

$L \cdot M = \{10, 11, 001, 1001, 1101, 00101\}$

$M \cdot L = \{0110, 0111, 01001, 10, 11, 001$

(iii) Star/Kleene closure:-

If $L = \{0, 1\}$

$L^* = \{\varepsilon, 0, 1, 00, 01, 10, 11, \ldots\}$

$= L^0 \cup L^1 \cup L^2 \cup L^3 \ldots$

$\phi^* = \{\varepsilon\}$ $\quad \phi^0 = \{\varepsilon\}$ $\quad \phi^+ = \phi$.

Regular Expression:-

(i) a, language $\{a\}$

(ii) ε , language $\{\varepsilon\}$

(iii) ϕ , empty language

(iv) $R_1 + R_2 \Rightarrow$ R_1 & R_2 are RE & + = union

(v) $R_1 \cdot R_2 \Rightarrow$ $\cdot$ concatenation

(vi) R^* $\Rightarrow$ signifies closure

(vii) (R) $\Rightarrow$ R is RE

* Building of RE:-

• Induction:-

$(R_1 \; R_2) \Rightarrow L_1 \cdot L_2$

$(R_1 \cup R_2) \Rightarrow L_1 \cup L_2$

$(R_1)^* \Rightarrow L_1^*$

Precedence:-

Highest ( )

$*$

$\cdot$

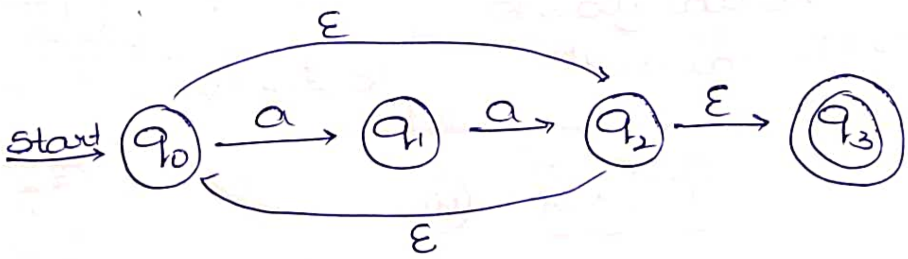
lowest $\cup$ or + or |

Eg: $R_1{}^* R_2 \cup R_3 \Rightarrow ((R_1{}^*) \cdot R_2) \cup R_3$.

Egs for given language:-

1] {w | w has exactly a single 1}	$R = 0^* 1 0^*$.
2] Set of all strings of 0's and is ending 00.	$R = (0+1)^* 0 0$
3) Set of all strings begin with 0 & ending with 1	$R = 0(0+1)^* 1$.
4] $L = \{\varepsilon, a, aa, aaa, \dots\}$	$R = a^*$
5] $L = \{\varepsilon, aa, aaaa, \dots\}$	$R = (aa)^*$
6] any no. of 0's followed by any no. of 1's followed by any no. of 2's.	$R = 0^* 1^* 2^*$
7] {w | w has length ≥ 3 & its 3rd symbol is 0.	$R = (0+1)(0+1)0(0+1)^*$

Sol of 5:-



Write a R.E that has

Prob:- which has strings of alternative 0s & 1s

Sol:- String = {01, 101, 0101, 1010, ---}

$$RE = (\varepsilon+1)(0\cdot1)^*(\varepsilon+0)$$

Prob:- const a RE for $L=\{0^n \mid n \bmod 3 = 0\}$

Sol:- String = {000, 000000, ---}.

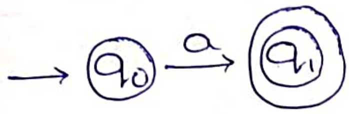
$$RE = (000)^*$$

Prob:- write RE to produce no. of conse 1's where $\Sigma=\{0,1\}$.

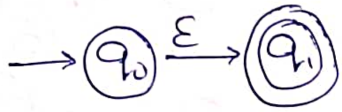
Sol:- $RE = 0\cdot1^*\cdot0$

RE → NFA

* **Base case.**



RE = a.



RE = ε

RE = ∅

Induction on length of R

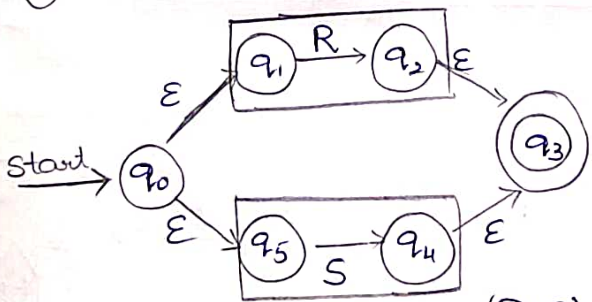
* **Thompson's construction:-**

Inductive step:-

R = length K>1

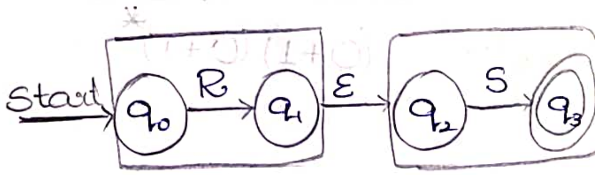
Four possibilities of R:-

(i) $\boxed{R = R+S}$



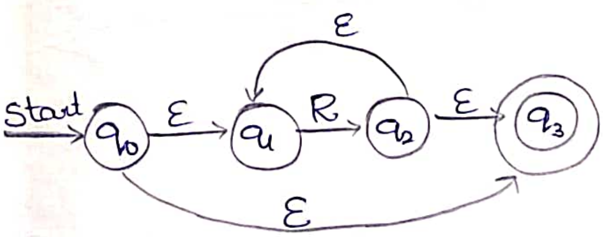
union Theorem (R+S)

(ii) $\boxed{R = R \cdot S}$



Concatenation Theorem - (R·S)

(iii) $\boxed{R = R^*}$



closure Theorem.

$R = R^*$

(iv) $\boxed{R = (R)}$

Paranthesized.

R = (R)

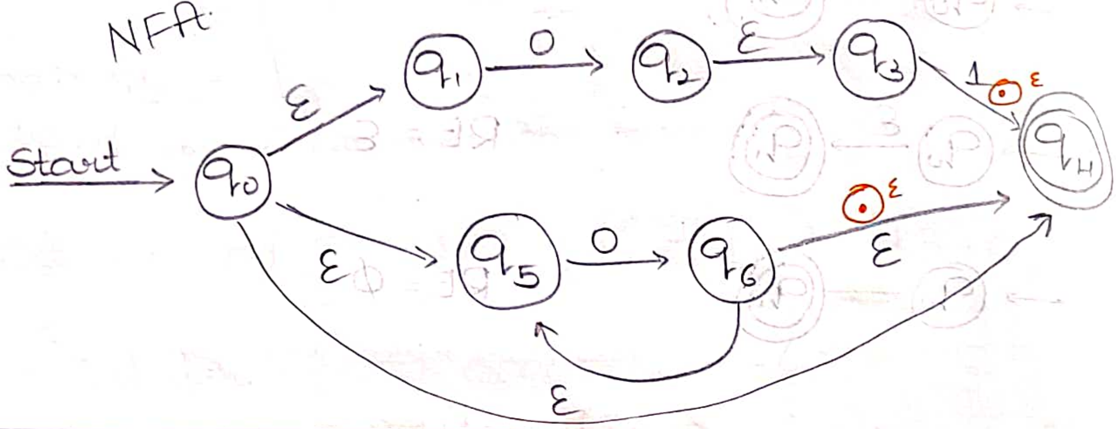
Prob:-] Const the RE → NFA

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

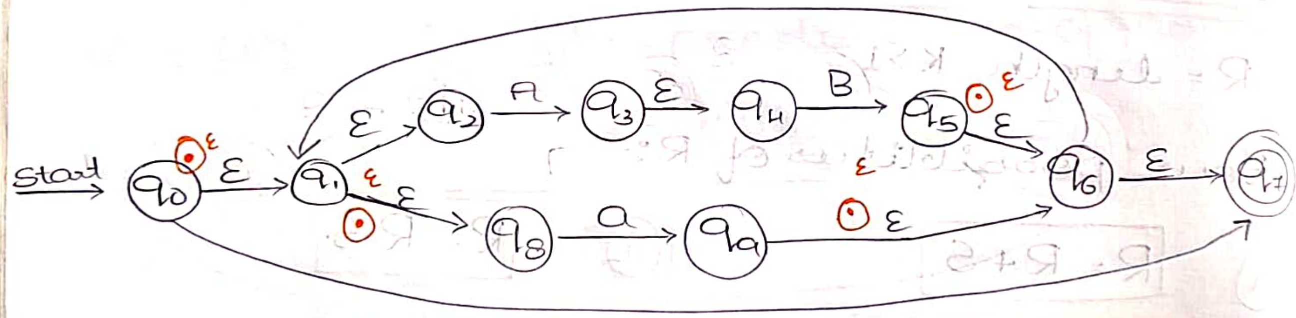
Here I should 2 states after 1 ε after 0

Sol:-

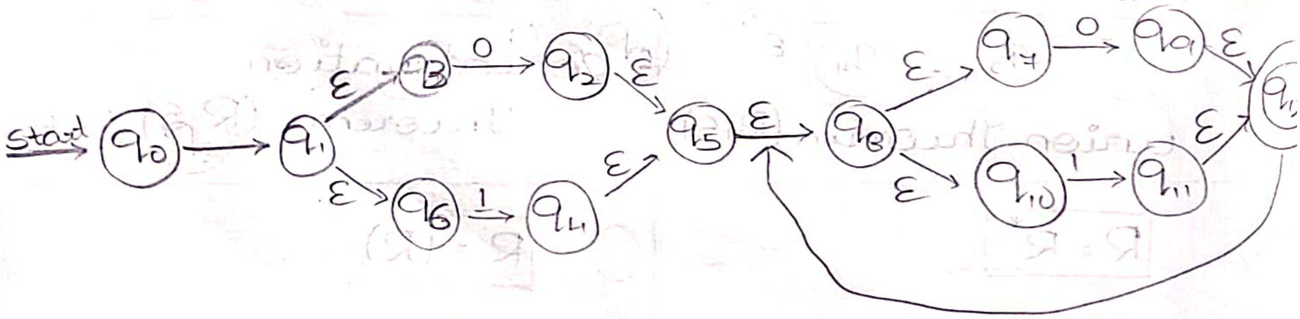
NFA



Prob:-] $(AB + A)^*$



Prob:-] $(0+1)(0+1)^*$



Formulas:-

$$\varepsilon + 1 = 1$$

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

$$R + RS^* = RS^*$$

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

$$\phi R = R\phi = \phi$$

Identity's :-

(i) $\phi + R = R$

(ii) $\phi R = R\phi = \phi$

(iii) $\varepsilon R = R\varepsilon = R$.

4) $\varepsilon^* = \varepsilon$ & $\phi^* = \varepsilon$

5) $R + R = R$

6) $R^* R^* = R^*$

13) $R^* R = R^+$

14)

7) $RR^* = R^* R$

8) $(R^*)^* = R^*$

9) $\varepsilon + RR^* = \varepsilon + R^* R = R^*$

10) $(PQ)^* P = P(Q \cdot P)^*$

11) $(P+Q)^* = (P^* Q^*)^* = (P^* + Q^*)^*$

12) $(P+Q) \cdot R = PR + QR$

$R(P+Q) = RP + RQ$

Arden's Theorem :-

P and Q are 2 RE over Σ; if P does not contain ε, then eq $\boxed{R = Q + R \cdot P}$ has unique sol $\boxed{R = QP^*}$

Proof :- $R = Q + RP$ —— ①.

For $R = QP^*$ is sol

Sub $R = QP^*$.

$\boxed{R^* R = R^+}$

$R = Q + QP^* P$

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

$R = QP^*$

Proof :- $R = Q + RP$.

$R = QP^*$ is unique sol

$= Q + (Q + RP)P$.

$= Q + QP + R \cdot P \cdot P$.

$= Q + QP + (Q + RP) \cdot P^2$

$= Q + QP + QP^2 + RP^3$.

$\vdots$

$= Q + QP + QP^2 + \cdots + QP^n + RP^{n+1}$.

$= Q + QP + \text{——} + QP^n + (QP^*) P^{n+1}$

$= Q[\varepsilon + P + P^2 + \cdots + P^n + P^* P^{n+1}]$

$= QP^*$

4\. Eg:] Prove $(1+00^*1)+(1+00^*1)(0+10^*1)^*(0+10^*1)$

Sol: $(1+00^*1)+(1+00^*1)(0+10^*1)^*(0+10^*1)$

$\Rightarrow (1+00^*1)[\varepsilon + (0+10^*1)^*(0+10^*1)]$

$\Rightarrow (1+00^*1)[\varepsilon + (0+10^*1)^+]$

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

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

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

$\Rightarrow \boxed{1(0)^*(0+10^*1)^*}$

4\. Eg:] Design RE of lang over $\{a,b\}$

(i) Lang accepting string length exactly 2

(ii) Lang accepting string length atleast 2

(iii) Lang accepting string length atmost 2

Sol: $L_1 = \{aa, ab, ba, bb\}$

$R = aa + ab + ba + bb$

$= a(a+b) + b(a+b)$

$= (a+b)(a+b)$

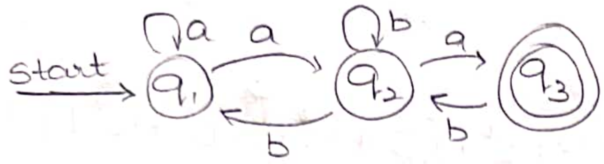
$L_2 = \{\cancel{a}, \cancel{b}, ab, ba, aa, bb, aaa, \cdots\}$

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

$L_3 = \{a, b, aa, ab, ba, bb, \varepsilon\}$

$R = \cancel{(a+b)}\ (\varepsilon+a+b)(\varepsilon+a+b)$

4. Eg:] Find RE for NFA.



Arden's Theorem

Sol:-

$$q_3 = q_2 a \quad \text{---} (1)$$

$$q_2 = q_1 a + q_2 b + q_3 b \quad \text{---} (2)$$

$$q_1 = \varepsilon + q_1 a + q_2 b \quad \text{---} (3)$$

$(1) \Rightarrow \quad q_3 = q_2 a$

$$= (q_1 a + q_2 b + q_3 b) a$$

$$= (q_1 aa + q_2 ba + q_3 ba) \quad \text{---} (4)$$

$(2) \Rightarrow \quad q_2 = q_1 a + q_2 b + q_3 b.$

$$= q_1 a + q_2 b + q_2 ab.$$

$$\underbrace{q_2}_{R} = \underbrace{q_1 a}_{Q} + \underbrace{q_2}_{R} \underbrace{(b+ab)}_{P} \qquad [\because R = Q + RP]$$

$$q_2 = (q_1 a)(b+ab)^* \quad \text{---} (5) \qquad [\because R = QP^*]$$

$(3) \Rightarrow \quad q_1 = \varepsilon + q_1 a + q_2 b.$

Sub (5) $\quad = \varepsilon + q_1 a + (q_1 a (b+ab)^*) b$

$$q_1 = \varepsilon + q_1 (a + ba(b+ab)^*) \qquad [\because R = Q + RP]$$

$$q_1 = \varepsilon \cdot (a + ab(b+ab)^*)^*$$

$$q_1 = (a + ab(b+ab)^*)^* \quad \text{---} (6)$$

Here after finding all states eq we have to sub in final state i.e here it is q_3. (eq (1)

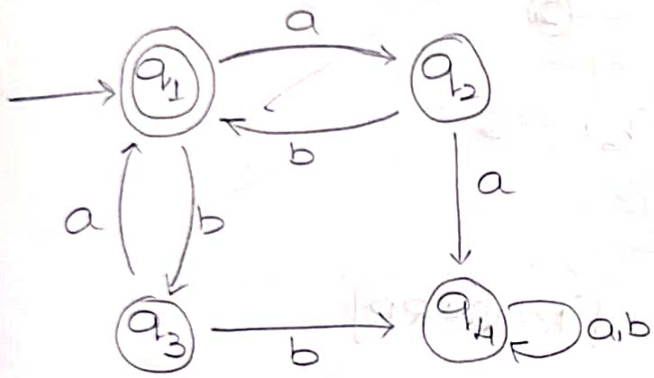
$q_3 = q_2 a$

$= q_1 a (b+ab)^* a$ [eq 5]

$q_3 = (a+a(b+ab)^* b)^* a (b+ab)^* a$ [eq 6].

∴ This is RE of NFA

4-Eg:- Find RE of given ~~NFA~~ DFA



$q_1 = q_2 b + q_3 a + \varepsilon$ —— ①

$q_2 = q_1 a$ —— ②

$q_3 = q_1 b$ —— ③

$q_4 = q_2 a + q_3 b + q_4 a + q_4 b$ —— ④

① ⇒ $q_1 = \varepsilon + q_2 b + q_3 a$

$= \varepsilon + q_1 ab + q_1 ab$.

$\underbrace{q_1}_{R} = \underbrace{\varepsilon}_{Q} + \underbrace{q_1}_{R} \underbrace{(ab+ba)}_{P}$ [∴ R = Q + RP]

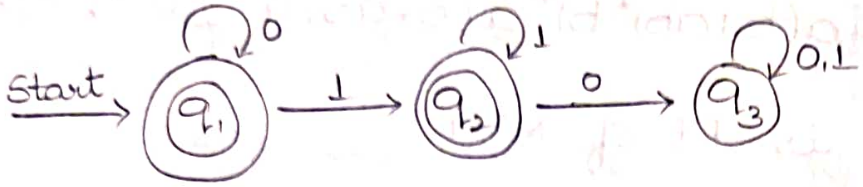
$q_1 = \varepsilon (ab+ba)^*$ [R = QP*]

$q_1 = (ab+ba)^*$ —— ⑤

We have to continue substitution until we get accepting state eq in terms of i/p

4. Eg:- When we get $\boxed{\text{multiple Final}}$ states

Find RE of following $\boxed{\text{DFA}}$.



Sol:-

$$q_1 = q_1 \cdot 0 + \varepsilon \quad \text{——} ①$$

$$q_2 = q_1 \cdot 1 + q_2 \cdot 1 \quad \text{——} ②$$

$$q_3 = q_2 \cdot 0 + q_3 0 + q_3 1 \quad \text{——} ③$$

Final state q_1

$$① \Rightarrow \underset{R}{q_1} = \underset{Q}{\varepsilon} + \underset{R}{q_1} \cdot \underset{P}{0}. \qquad [R = Q + RP]$$

$$\Rightarrow q_1 = \varepsilon \cdot 0^* \Rightarrow \boxed{q_1 = 0^*} \quad \text{——} ④$$

Final State q_2

$$② \Rightarrow q_2 = q_1 \cdot 1 + q_2 \cdot 1$$

Sub ④ $$\underset{R}{q_2} = \underset{Q}{(0^* \cdot 1)} + \underset{R}{q_2} \cdot \underset{P}{1} \qquad [R = Q + RP]$$

$$\boxed{q_2 = (0^* \cdot 1) 1^*} \qquad [R = QP^*]$$

$$\therefore RE = q_1 + q_2$$

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

$$= 0^* (\varepsilon + 1 \cdot 1^*) \qquad [\varepsilon + RR^* = R^*]$$

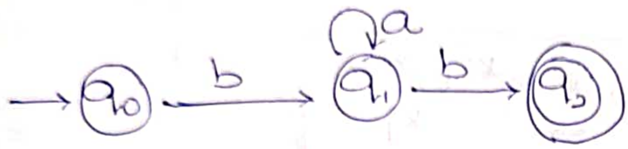
$$\boxed{RE = 0^* 1^*} \qquad [RR^* = R^+]$$

4. Eg:- RE → F.A

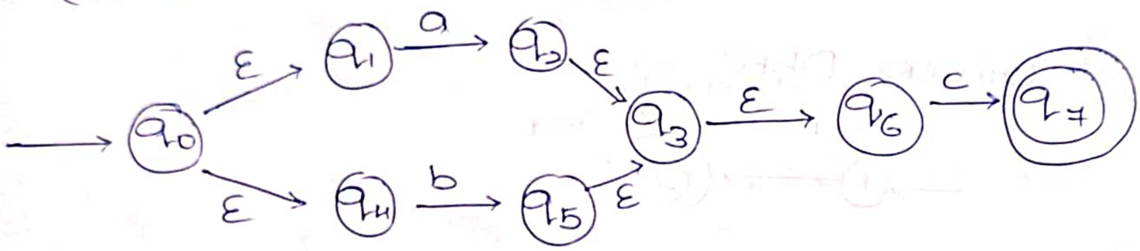
Convert following RE to FA

(i) $b.a^*b$ (ii) $(a+b)c$ (iii) $a(bc)^*$

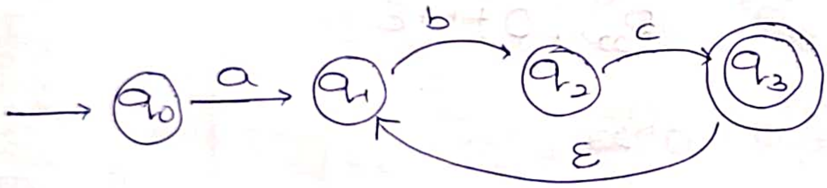
Sol:- (i) ba^*b : {bb, bab, baab, ---}



(ii) $(a+b)c$

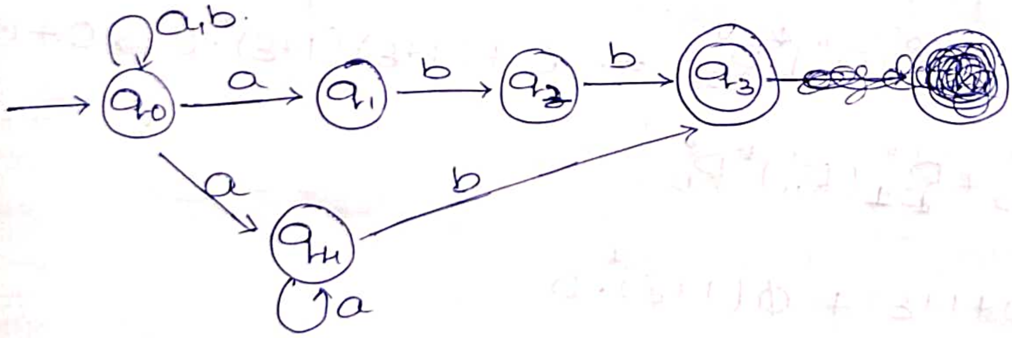


(iii) $a(bc)^*$ = {a, abc, abcbc, ---}



---

4. Eg:- convert RE → F.A $(a|b)^*(abb|a^*b)$



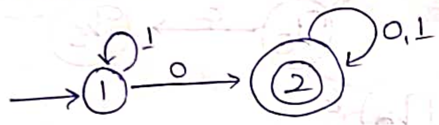
Conversion of DFA $\longrightarrow$ RE (R_{ij}^k method)

R_{ij}^k

- $k \rightarrow$ no. of states
- $R \rightarrow$ Regular Expression
- $i \rightarrow$ Ini state
- $j \rightarrow$ accepting state

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

P.Eg:-] Given DFA is



Sol:-]

R_{11}^0	$1+\varepsilon$	R_{21}^0	ϕ
R_{12}^0	0	R_{22}^0	$0+1+\varepsilon$
$\cancel{R_{13}}$		$\cancel{R_{23}}$	

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

$$R_{12}^1 = R_{12}^0 + R_{11}^0\left(R_{11}^0\right)^* R_{12}^0 \Rightarrow 0 + (1+\varepsilon)(1+\varepsilon)^*\cdot 0 \Rightarrow 0+1\cdot1^*\cdot 0$$

$$\Rightarrow 0+1^*\cdot 0$$

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

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

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

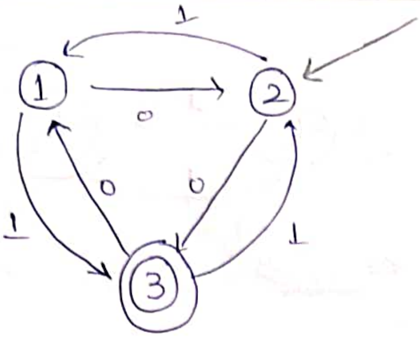
$$\Rightarrow R_{12}^2 = (0+1^*\cdot 0) + (0+1^*\cdot 0)(0+1)^*(0+1)$$

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

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

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

P.Eg:-



$k=0$

R_{11}^0	ε
R_{12}^0	0
R_{13}^0	1
R_{21}^0	1
R_{22}^0	ε
R_{23}^0	0
R_{31}^0	0
R_{32}^0	1
R_{33}^0	ε

$k=1$

	$R_{ij}^1 = R_{ij}^0 + R_{i1}^0 (R_{11}^0)^* R_{1j}^0$	
R_{11}^1	$R_{11}^0 + R_{11}^0 (R_{11}^0)^* R_{11}^0 \Rightarrow \varepsilon + \varepsilon(\varepsilon)^* \varepsilon$	ε
R_{12}^1	$R_{12}^0 + R_{11}^0 (R_{11}^0)^* R_{12}^0 \Rightarrow 0 + \varepsilon(\varepsilon)^* 0$	0
R_{13}^1	$R_{13}^0 + R_{11}^0 (R_{11}^0)^* R_{13}^0 \Rightarrow 1 + \varepsilon(\varepsilon)^* 1$	1
R_{21}^1	$R_{21}^0 + R_{21}^0 (R_{11}^0)^* R_{11}^0 \Rightarrow 1 + 1(\varepsilon)^* \varepsilon$	1
R_{22}^1	$R_{22}^0 + R_{21}^0 (R_{11}^0)^* R_{12}^0 \Rightarrow \varepsilon + 1(\varepsilon)^* 0$	$\varepsilon + 10$.
R_{23}^1	$R_{23}^0 + R_{21}^0 (R_{11}^0)^* R_{13}^0 \Rightarrow 0 + 1(\varepsilon)^* 1$	$0 + 1 \cdot 1$
R_{31}^1	$R_{31}^0 + R_{31}^0 (R_{11}^0)^* R_{11}^0 \Rightarrow 0 + 0(\varepsilon)^* \varepsilon$	0
R_{32}^1	$R_{32}^0 + R_{31}^0 (R_{11}^0)^* R_{12}^0 \Rightarrow 1 + 0(\varepsilon)^* 0$	$1+00$
R_{33}^1	$R_{33}^0 + R_{31}^0 (R_{11}^0)^* R_{13}^0 \Rightarrow \varepsilon + 0(\varepsilon)^* 1$	$\varepsilon+01$

$$R_{23}^2 \Rightarrow R_{23}^1 + R_{22}^1 (R_{22}^1)^* R_{23}^1 = (0+1\cdot1) + (\varepsilon+01)(\varepsilon+10)^*(0+11)$$

$$= (0+11) + (\varepsilon+10)^*(0+11)$$

$$= (0+11)(\varepsilon+10)^*$$

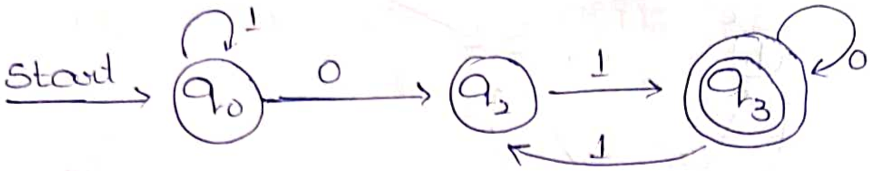
$$R_{33}^2 \Rightarrow R_{33}^1 + R_{32}^1 (R_{22}^1)^* R_{23}^1 = (\varepsilon+10) + (1+00)(\varepsilon+10)^*(0+11)$$

$$R_{23}^3 = R_{23}^2 + R_{23}^2 (R_{33}^2)^* R_{33}^2$$

$$= (0+11)(\varepsilon+10)^* + (0+11)(\varepsilon+10)^* [(10) + (1+00)(10)^*(0+11)]^*$$

$$\Rightarrow [(0+11)(\varepsilon+10)^*][(10)+(1+00)(10)^*(0+11)]^*$$

P. Eg:-]



Sol:-] $R_{ij}^{k} = R_{13}^{3}$

$k=0 \Rightarrow$

R_{11}^{0}	$1+\varepsilon$	R_{21}^{0}	ϕ	R_{31}^{0}	ϕ
R_{12}^{0}	0	R_{22}^{0}	ε	R_{32}^{0}	1
R_{13}^{0}	ϕ	R_{23}^{0}	1	R_{33}^{0}	$\varepsilon+0$

$k=1 \Rightarrow \boxed{R_{ij}^{1} = R_{ij}^{0} + R_{i1}^{0}(R_{11}^{0})^{*}R_{1j}^{0}}$

R_{11}^{1}	$R_{11}^{0}+R_{11}^{0}(R_{11}^{0})^{*}R_{11}^{0}$	$(1+\varepsilon)+(1+\varepsilon)(1+\varepsilon)^{*}(1+\varepsilon)$	$(1+\varepsilon)+(1+\varepsilon)^{*}$
R_{12}^{1}	$R_{12}^{0}+R_{11}^{0}(R_{11}^{0})^{*}R_{12}^{0}$	$0+(1+\varepsilon)(1+\varepsilon)^{*}(0)$	$0+1^{*}0 \Rightarrow 0$
R_{13}^{1}	$R_{13}^{0}+R_{11}^{0}(R_{11}^{0})^{*}R_{13}^{0}$	$\phi+(1+\varepsilon)(1+\varepsilon)^{*}\phi$	ϕ
R_{21}^{1}	$R_{21}^{0}+R_{21}^{0}(R_{11}^{0})^{*}R_{11}^{0}$	$\phi+\phi(1+\varepsilon)^{*}(1+\varepsilon)$	ϕ
R_{22}^{1}	$R_{22}^{0}+R_{21}^{0}(R_{11}^{0})^{*}R_{12}^{0}$	$\varepsilon+\phi(1+\varepsilon)^{*}0$	ε ϕ
R_{23}^{1}	$R_{23}^{0}+R_{21}^{0}(R_{11}^{0})^{*}R_{13}^{0}$	$1+\phi(1+\varepsilon)^{*}1$	1 ϕ
R_{31}^{1}	$R_{31}^{0}+R_{31}^{0}(R_{11}^{0})^{*}R_{11}^{0}$	$\phi+\phi(1+\varepsilon)^{*}(1+\varepsilon)$	ϕ
R_{32}^{1}	$R_{32}^{0}+R_{31}^{0}(R_{11}^{0})^{*}R_{12}^{0}$	$1+\phi(1+\varepsilon)(0)$	1
R_{33}^{1}	$R_{33}^{0}+R_{31}^{0}(R_{11}^{0})^{*}R_{33}^{0}$	$(\varepsilon+0)+\phi(1+\varepsilon)^{*}(\varepsilon+0)$	$\varepsilon+0$

$$\boxed{R_{13}^{3} = R_{13}^{2} + R_{13}^{2}(R_{33}^{2})^{*}R_{33}^{2}} \Rightarrow R_{13}^{3} \cancel{= \phi+\phi(1)^{*}1}$$

$$R_{13}^{2} = R_{13}^{1} + R_{12}^{1}(R_{22}^{1})^{*}R_{23}^{1} \Rightarrow \cancel{\phi+(0+1^{*}0)(\overset{\varepsilon}{\phi})(\overset{1}{\phi})} \Rightarrow \phi.$$

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

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

$$\Rightarrow \varepsilon \cdot 1^{*}0 \cdot 1 \Rightarrow 1^{*} \cdot 0 \cdot 1$$

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

$$\Rightarrow 0 + 1 \cdot 1$$

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

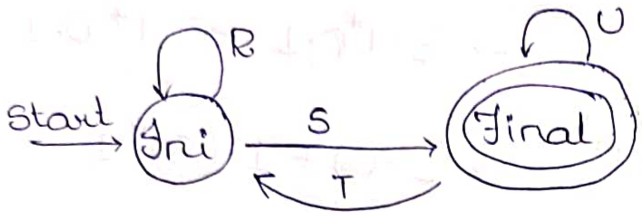
$$= (1^{*}01) + (1^{*}01)(0+1\cdot1)^{*}(0+1\cdot1)$$

$$= (1^{*}01) + (1^{*}01)(0+1\cdot1)^{*}$$

$$= (1^{*}01)(0+1\cdot1)^{*}$$

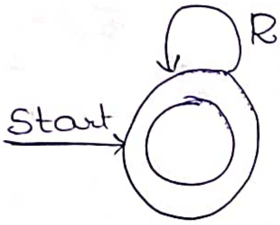
RULES
State Elimina

* When two states are different,



∴ $\boxed{(R+SU^*T)^*SU^*}$

* When only one state means ini & accept state are same.



∴ $\boxed{R^*}$

* When we have one ini state & n accepting states
we consider one accepting & remaining non-accepting.

∴ $\boxed{R_1 \cup R_2 \cup R_3 \cup \dots \cup R_n}$

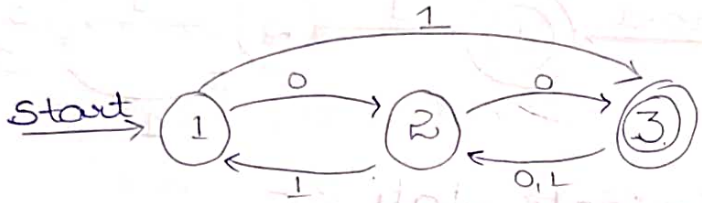
DFA → RE : [State Elimination]

- Eliminating intermediate states & convert to small automata. (2 state / 1 state)

  2 State ⇒ Ini / accept

  1 State ⇒ Ini + accept

P. Eg:-3]



Sol:-

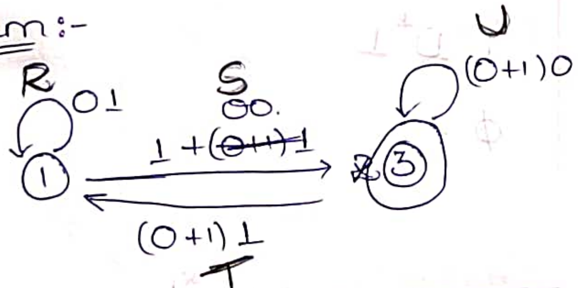
$11 \Rightarrow \phi + 01$

$33 \Rightarrow \phi + (0+1)0$

$13 \Rightarrow 1 + \cancel{(0+1)1}\ 00.$

$31 \Rightarrow \phi + (0+1)1$

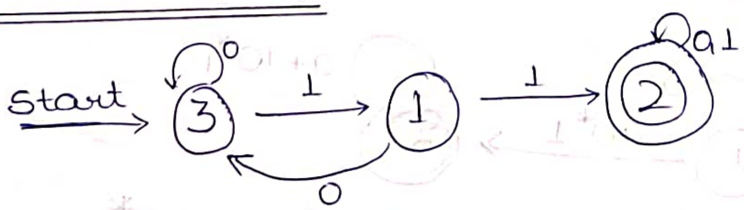
Diagram:-



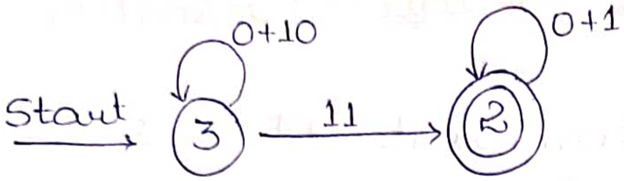
Formula:- $\boxed{(R+SU^*T)^*SU^*}$

$$\Rightarrow 01 + [(1+00)\cdot[(0+1)0]^*((0+1)\cdot 1)]^*(1+00)\cdot[(0+1)0]$$

Eg:-1]

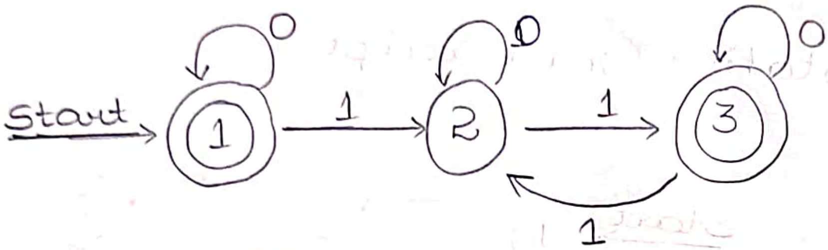


3	3	$0 + 10$
2	2	$\phi + (1+0)$
3	2	$\phi + 11$
2	3	ϕ

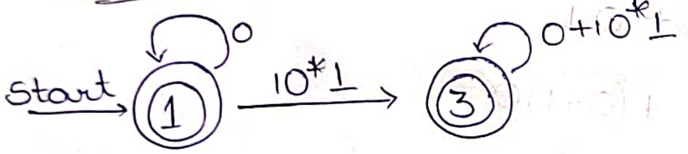


$RE \Rightarrow (0+10)^* 11 (0+1)^*.$

Eg:-2]



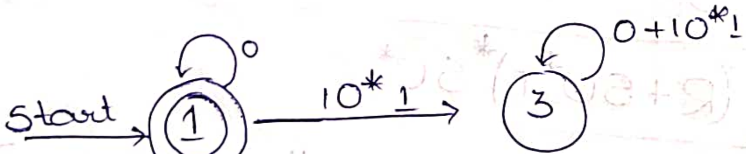
Sol:-] Eliminate state 2:-



Proof:-

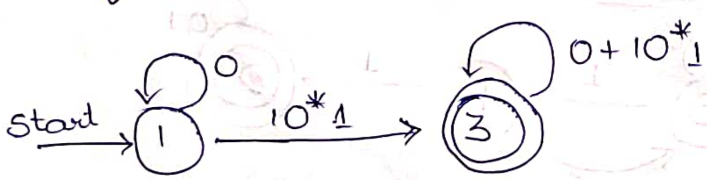
1	1	0
3	3	$0+10^*1$
1	3	10^*1
3	1	ϕ.

Turn of 3:-



$RE_1 = 0^*$

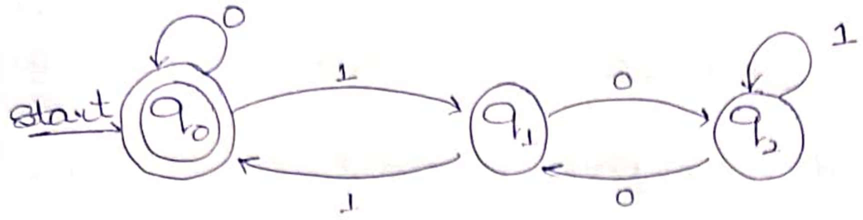
Turn of 1:-



$RE_2 = 0^* \cdot 10^* 1 \cdot (0+10^*1)^*$

$\therefore RE = 0^* 10^* 1 (0+10^*1)^* + 0^*$

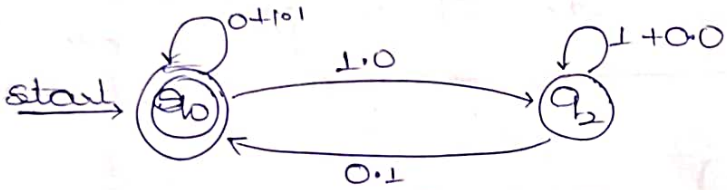
MOD·PAP-7]



Sol:- we have do eliminate q_1 & q_2

q_1:-

q_0	q_0	$0 + 1\cdot1$
q_0	q_2	$\phi + 1\cdot0$
q_2	q_0	$\phi + 0\cdot1$
q_2	q_2	$1 + 0\cdot0$.



q_2:-

q_0	q_0	$(0+1\cdot1) + 1\cdot0\cdot0\cdot1$
q_0	q_2	$\phi + 1\cdot0$
q_2	q_0	$\phi + 0\cdot1$
q_2	q_2	$(1+0\cdot0)+(0\cdot1\cdot1\cdot0)$

$$q_0 = \left[(0+1\cdot1) + 1\cdot0\,(1+0\cdot0)^*\,0\cdot1\right]^*$$

Pumping Lemma.

* It is a methodoly to say Lang is regu/not
* Pumping lemma proves the lang is not regular
* We cannot use pumping lemma to say lang is regular.

* Pumping lemma:-

L = regular language $|L| = \infty$

There exist a positive integer n such that

if $w \in L$ & $|w| \geq n$.

then $w = xyz$

(i) $|y| > 0$ or $y != \varepsilon$ $(y \neq \varepsilon)$

(ii) $|xy| \leq n$

(iii) $xy^kz \in L$ for any $k \geq 0$.

P.Eg:] Show that $L = \{0^n 1^n \mid n \geq 0\}$ is not regular

Sol:] (i) L = regular lang.

(ii) $\omega = 0^3 1^3 \Rightarrow 0^i 1^i$

$|\omega| = 2i \geq n \Rightarrow n = 2i.$

(iii) $\omega = 0^i 1^i$

2 ways $\begin{cases} = 0^{i-k}\, 0^k\, 1^i & [x \quad y \quad z] \\ = 0^k\, 0^{i-k}\, 1^i & \\ \;\; x \quad y \quad z & \end{cases}$

(1) $y \neq \varepsilon$ (2) $|xy| \leq n$

$i \leq 2i.$

(3) $xy^kz \quad k \geq 0.$

(iv) $K = 2$

1st way: $xy^2z \Rightarrow 0^k \cdot 0^{i-k} \cdot 0^{i-k} \cdot 1^i$

$\Rightarrow 0^2\, 0^{i-2}\, 0^{i-2}\, 1^i$

$\Rightarrow 0^{2i-2}\, 1^i \notin L$ (powers are not lang

$\therefore$ Here it is not a regular language

2nd way $xy^2z = 0^{i-k}\, 0^k\, 0^k\, 1^i$

$= 0^{i+k}\, 1^i \notin L.$

$\therefore$ powers are not same

Here it is not regular language

P.Eg:- Show that ~~p~~ Lang L consists of all palindrome over $(0+1)^*$ is not regular.

Sol:- Let us assume given L is regular

$\omega = 0^n 1 0^n$. $\qquad |\omega| = 3n+1 \geq n$. $\qquad \boxed{n = 3n+1}$

$= \underbrace{0^{n-j}}_{x} \underbrace{0^j}_{y} \underbrace{10^n}_{z}$

$k=3$, by pumping lemma.

$\Rightarrow 0^{n-j} 0^j 0^j 10^n$

$\Rightarrow 0^{n+j} 1 0^n \notin L \Rightarrow$ It is not reg lang.

P.Eg:- Show that $L = \{a^i b^j c^k \mid 0 \leq i < j < k\}$ is not regular.

Sol:- $\omega = a^i b^j c^k \Longrightarrow$ ~~$|\omega| = i+j+k \geq n$~~

~~$n = i+j+k$~~

$= a^1 b^2 c^3$.

$= a^n b^{n+1} c^{n+2} \Rightarrow |\omega| = 3n+3 \geq n$

$\boxed{n = 3n+3}$

$x = a^{n-k} \qquad y = a^k \qquad z = b^{n+1} c^{n+2}$.

$k=2 \Rightarrow \quad \omega = a^{n-k} a^k a^k b^{n+1} c^{n+2}$.

$= a^{n+k} b^{n+1} c^{n+2}$.

$k=2 \qquad = a^{n+2} b^{n+2} c^{n+2} \notin L$.

$\therefore$ powers are not eq

$\therefore$ It is not a reg lang.

P.Eg:-) Show that $L = \{a^n b^{2n} \mid n > 0\}$ is not reg.

Sol:-) Let assume L is regular.

'n' is a positive integer

$w = a^n b^{2n}$.

$|w| = n + 2n = 3n$

$|w| >= n \Rightarrow n = 3n$

$w = xy^k z$

$x = a^{n-j} \quad y = a^j \quad z = b^{2n} \quad [j > 0]$

cond (i) $\Rightarrow |y| > 0$.

cond (ii) $\Rightarrow |xy| = |a^{n-j} a^j|$

$= n \leq n$

$= n \leq 3n$

cond (iii) By pumping lemma

$xy^k z = a^{n-j} . a^{jk} b^{2n}$.

where $k=2 \Rightarrow a^{n-j} a^j a^j b^{2n}$

$\Rightarrow a^{n+j} b^{2n} \neq a^n b^{2n}$.

Here $a^{n+j} b^{2n}$ does not belong to language. So it is not a regular Expression.

P.Eg:- Show that $L = \{a^p \mid p \text{ is prime}\}$ is not reg

Sol:- Let us assume L is regular

'p' is prime number

Let $p = r+s+t$

Eg:- $p = 5 \Rightarrow 1+2+2$
$3+1+1$

$w = \{a, aa, aaa, aaaaa, aaaaaaa, \ldots\}$

$|w| = |a^{r+s+t}|$

$= r+s+t$

$|w| >= n \Rightarrow \quad n = r+s+t$

1] $s > 0$ 2] $|xy| \leq n$ 3] $xy^{p+1}z$

$x = a^r \quad y = a^s \quad z = a^t$

By pumping lemma;

$a^r \cdot (a^s)^p a^s \cdot a^t$.

$\Rightarrow a^{r+s+t} \cdot a^{sp}$

$\Rightarrow a^p \cdot a^{sp}$

$\Rightarrow a^{p(1+s)}$

$\Rightarrow$ prime $\times (s+1)$ does not belongs to L
[Some no.]

$\therefore$ Language is not a regular.

P.Eg:- Show that $L = \{0^n \mid n \text{ is perfect sq}\}$ is not regular.

Sol:- Let us assume that lang is regular

Let $w = xyz$ $w = 0^{n^2}$.

$|w| >= n$.

$|xy| <= n$.

$|y| > 0$.

By pumping lemma, $K=2$

$|xy^2z| \Rightarrow |xyz| + |y|$

$\Rightarrow |w| + |y|$

$\Rightarrow n^2 + (1 \leq |y| \leq n)$

$\Rightarrow n^2+1 \leq n^2+|y| \leq n^2+n.$

$\Rightarrow n^2+1 \leq |xyz|+|y| \leq n^2+n$

$\Rightarrow n^2+1 \leq |xy^2z| \leq n^2+n$

$\therefore$ Hence the min & max length are not perfect square, hence language is not regular.

MODULE - 3.

Closure properties of Regular Lang.

1] Union of 2 Reg Lang is regular
2] Intersection of 2 RL is regular
3] Compliment of Reg Lang is regular
4] Diff of 2 Reg Lang is regular
5] Reversal of Reg Lang is regular
6] Closure of Reg Lang is regular
7] Concatination of Reg Lang is regular
8] Homomorphism of Reg Lang is regular
9] Inverse homomorphism of Reg is regular

1] Closure under union:-

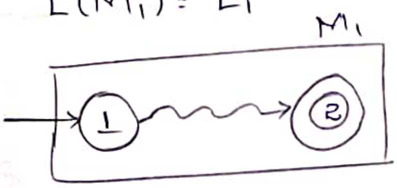
L & M are regular Languages. $L \cup M$.

$L = L(R)$ $\quad$ $M = L(S)$ $\quad$ $\boxed{NFA}$

$$\boxed{L \cup M = L(R+S)}$$

Reg Lang L_1

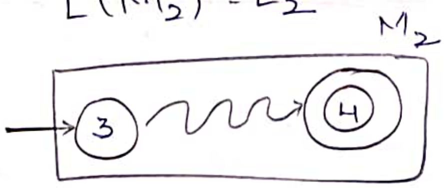
$L(M_1) = L_1$



Single accepting state

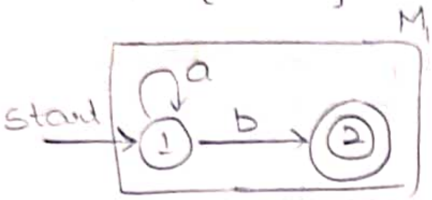
Reg Lang L_2

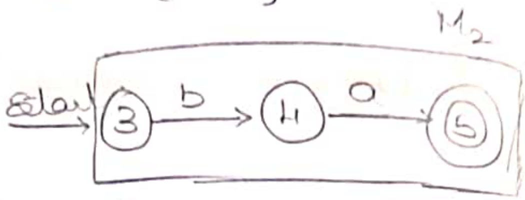
$L(M_2) = L_2$



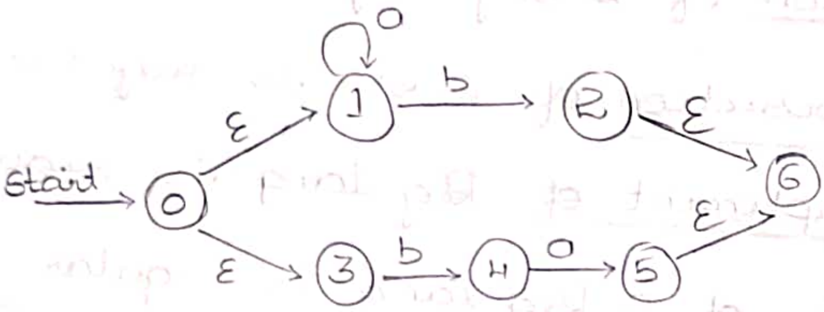
Single accepting State.

Eg:- $L_1 = \{a^n b\}$ $L_2 = \{ba\}$





$L_1 \cup L_2 \Rightarrow$

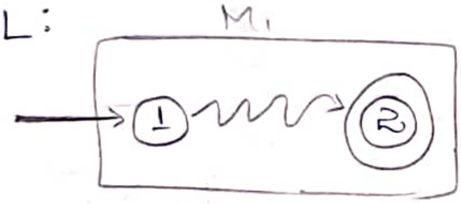


2] Compliment :-

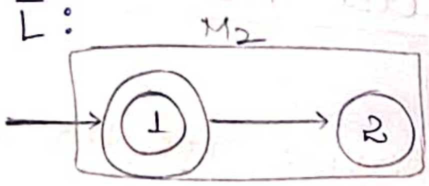
$\bar{L}$ is compliment of L.

- Construct a $\boxed{DFA}$ for Lang.
- Convert accepting State → non accepting States &
  non accepting → accepting

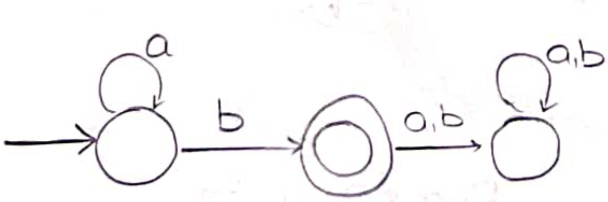
L:

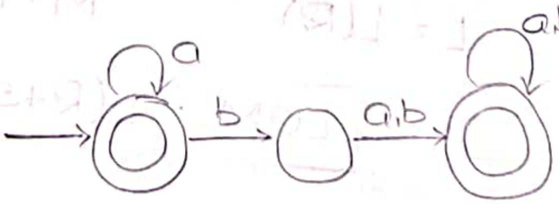


$\bar{L}$:



Eg:- $L_1 = \{a^n b\}$ $\bar{L_1} = \{a,b\}^* - \{a^n b\}$





3] Intersection:-

Demorgan's law $\boxed{L \cap M = \overline{\overline{L} \cup \overline{M}}}$

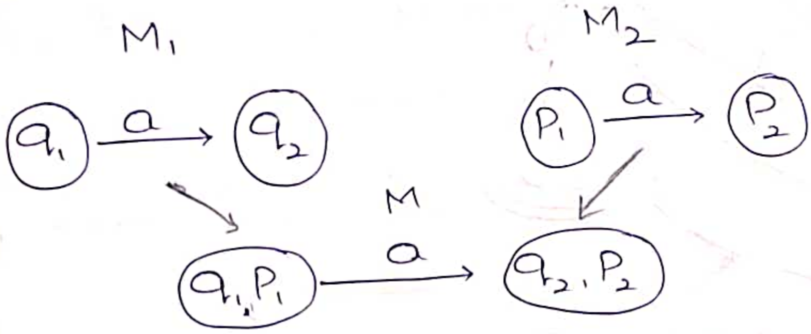
Another proof:-

M_1 $\boxed{\text{DFA for } L_1}$

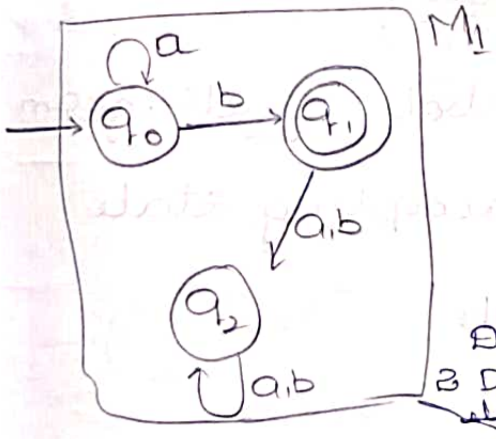
M_2 $\boxed{\text{DFA for } L_2}$

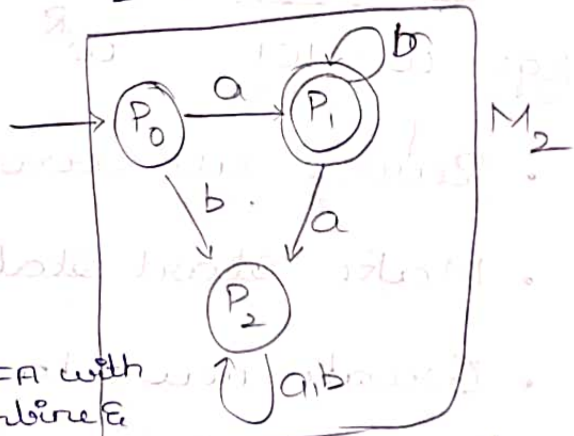
M $\boxed{L_1 \cap L_2}$

Diagram:-

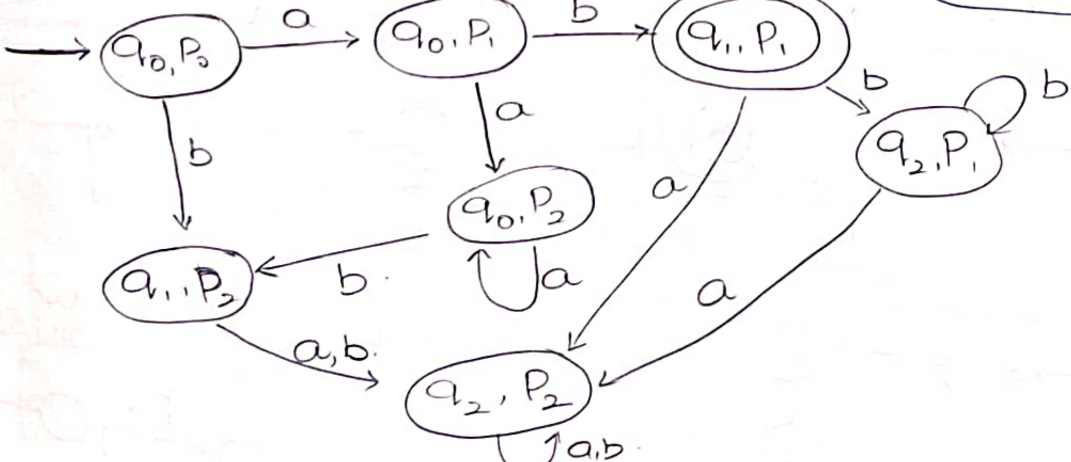


Eg:- $L_1 = \{a^n b\}$ $L_2 = \{ab^m\}$





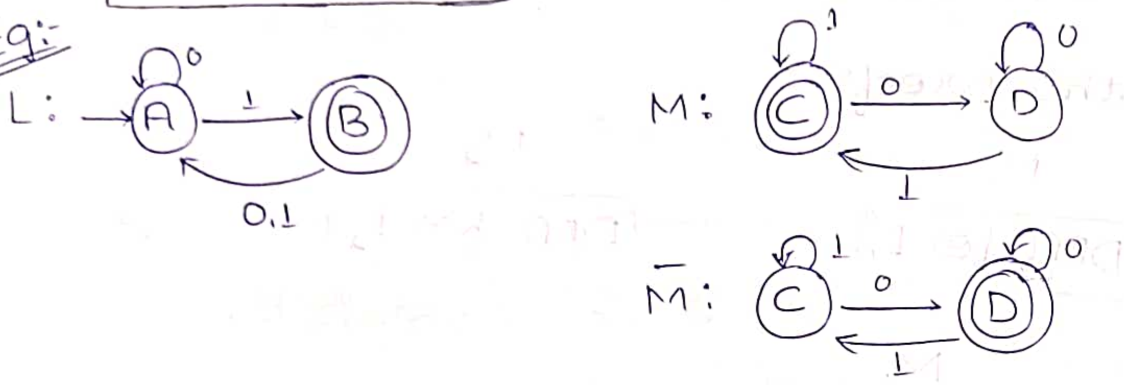
Draw a DFA with 2 DFA & combine & check carefully about creating states.



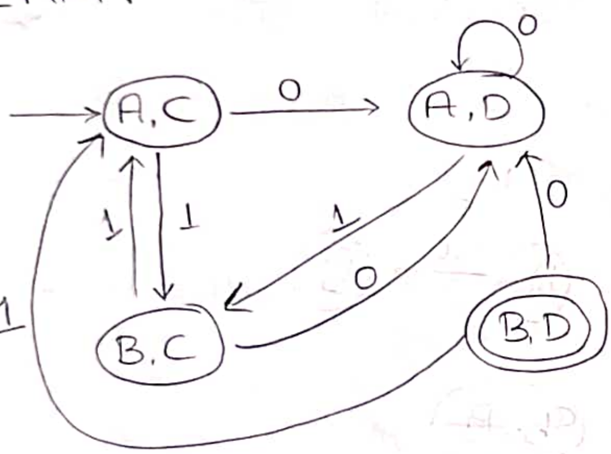
4] Difference:-

$$\boxed{L - M = L \cap \bar{M}}$$

Eg:-



$L \cap \bar{M}$:

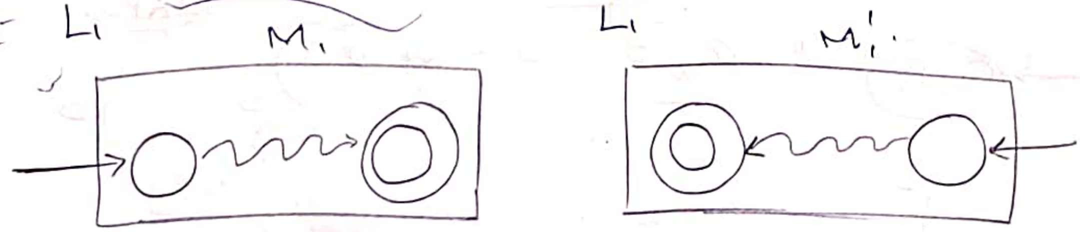


5] Reversal:-

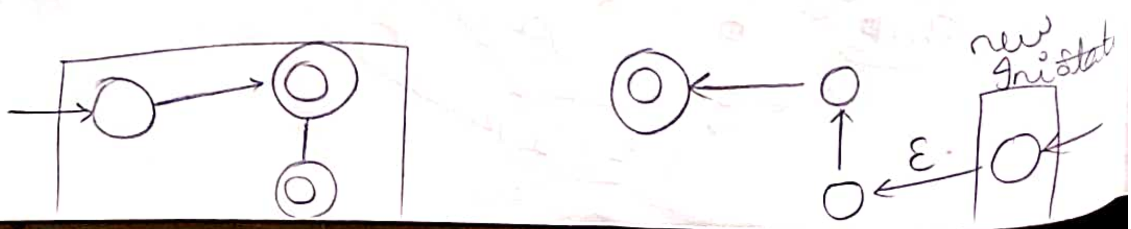
Eg:- $\omega = 1011$ $\omega^R = 1101$

- Reverse all arrow symbols of diagram
- Make start state → accepting state.
- Create new start state

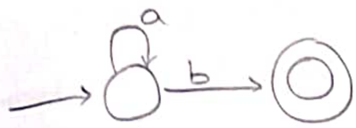
Eg:- L_1 L_1^R

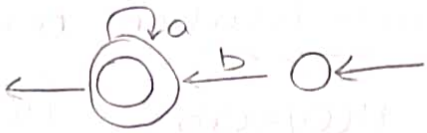


Eg:-

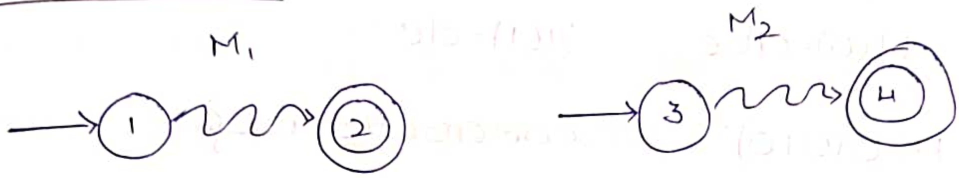


Eg:- $L_1 = \{a^n b\}$ $\qquad$ $L_1^R = \{b a^n\}$



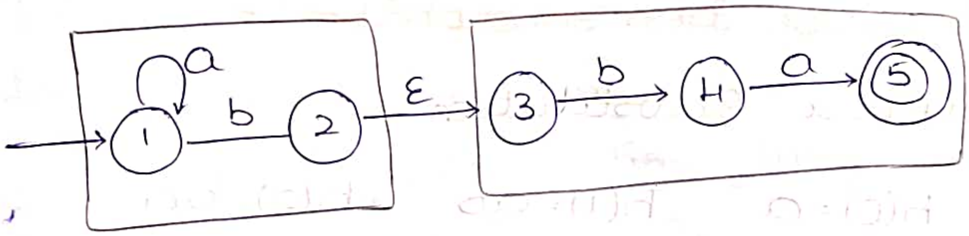


6] **Concatenation:-** Adding both by ε-Trans



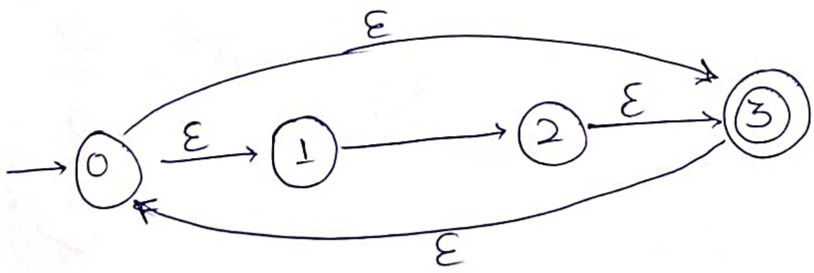
$M_1 . M_2 \Rightarrow$

Eg:- $L_1 = \{a^n b\}$ $\quad$ $L_2 = \{ba\}$ $\quad$ $L_1 L_2 = \{a^n b b a\}$

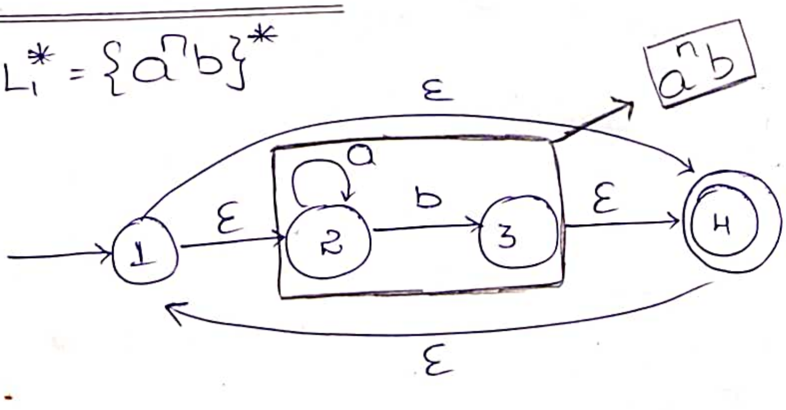


7] **Star operation:-** Here we create new initial & accepting state. Keep ε-Transition from ini → accp & accp → Ini.

L_1^*



Eg:- $L_1^* = \{a^n b\}^*$



8] Homomorphism:-

- Substitution method.

Eg:- $H(0) = ab$ $\quad H(1) = \varepsilon$

$H(01010) = \{ababab\}$

Eg:- $H(0) = abc$ $\quad H(1) = de$.

$H(01010) = \{abcdeabcdeabc\}$

Eg:- $H(0) = a$ $\quad H(1) = ab$ $\quad H(2) = ba$

$H(01^*2) = a(ab)^*ba$.

9] Inverse Homomorphism:-

- Reverse Substitution.

Eg:- $h(0) = a$ $\quad h(1) = ab$ $\quad h(2) = ba$.

ababa

$h^{-1} = \{022, 110, 102\}$

- Result of $\boxed{h^{-1}}$ is also $\boxed{\text{Regular language}}$

Decision Properties of RL:-

① Emptyness	② Membership	③ Conversion
(whether lang producing Empty or not)	checking whether the string belongs to that lang.	(i) NFA → DFA (ii) DFA → NFA (iii) FA → RE (iv) RE → FA

Conversions:-

$\boxed{\begin{array}{l} n = \text{String length} \\ S = \text{States} \end{array}}$

(i) NFA ⟶ DFA :-

NFA with "ε" ⇒ $O(n^3 2^n)$ → ~~states~~ [Takes more time to remov ε & convert to DFA]

NFA without "ε" ⇒ $O(n^3 S)$ → ~~Subset~~

(ii) DFA ⟶ NFA:-

Runtime $O(n)$

(iii) FA ⟶ RE:- [R_{ij}^K | State Elimination]

Runtime $O(n^3 4^n 2^n)$

(iv) RE ⟶ FA:-

Runtime $O(n)$ where string length n^3.

(v) Emptyness:-

Runtime ⇒ $O(n^2)$

(vi) Membership:-

Runtime ⇒ $O(nS^2)$

Chomsky Hierarchy for Language.

• Table.

Type	Language (Grammars)	Form of Production	Accepting Device.
3	Regular	$A \to aB;\ A \to \Lambda$	Finite Auto
2	Context-free	$A \to \alpha$	Pushdown automata
1	Context-Sensitive	$\alpha \to \beta$. with $\lvert\beta\rvert \geq \lvert\alpha\rvert$	LBA (linear bounded automata)
0	unrestricted	$\alpha \to \beta$	Turing machine

Context-Free Grammar :- (CFG).

* In NFA | DFA $\Rightarrow M = \{Q, \Sigma, \delta, q_0, F\}$

* In Grammar; Def has 4 pupils.

$\boxed{G = \{N, T, P, S\}}$ or $\{V, T, P, S\}$.

N = Finite set of non-Terminals = $\{A, B, \ldots, Z\}$

T = Finite set of Terminals. = $\{0, \ldots, 9, a, b, \ldots z\}$.

P = Set of Production Rules.

$\alpha \to \beta$, $\alpha \Rightarrow$ single non-Terminal

$\beta \in (N \cup T)^*$

S : Start symbol $S \in N$ (non-Terminal)

V = Finite set of non variables

* P ⇒ δ - Trans T = input Symbol [lower case & 0-9

S ⇒ Ini state N / V = all upper case letters ($A \to Z$).

* **P:-** $\boxed{N \to (N \cup T)^*}$

Eg:- $A \to AB$, $A \to aAa$, $A \to \varepsilon$.

- $|\alpha| \Rightarrow 1$. (always)

Eg:- 1] Find CFG for $L = \{a^n b^n, n \geq 0\}$

Sol:- $T = \{a, b\}$

$S = \{S\}$ $N = \{S\}$

P: $\boxed{\begin{array}{l} S \to aSb \\ S \to \varepsilon. \end{array}}$ — ① — ②

Derivation:- How to derive string from ini state

① string = aabb.

$S \xrightarrow{*} aSb$ (eq ①)

$S \to aaSbb$ (eq ①)

$\to aa\varepsilon bb$ (eq ②)

$\to \boxed{aabb}$

② string = aaabbb.

$S \xrightarrow{*} aSb$ .. ①

$\to aaSbb$ --- ①

$\to aaaSbbb$ .... ①

$\to \boxed{aaabbb}$ .... ②

Eg:-2] Find CFG for $L = \{a^n b^n, n \geq 1\}$.

Sol:-] $\boxed{\begin{array}{l} S \rightarrow aSb \\ S \rightarrow ab \end{array}}$ $\begin{array}{l} \ldots (1) \\ \ldots (2) \end{array}$ } CFG.

(i) String $= a^2b^2$.

$S \rightarrow aSb$ (eq (1))

$S \rightarrow aabb$ (eq (2)).

Complete Sol:-

CFG; $G = \{N, T, P, S\}$

$N = \{S\}$ $T = \{a, b\}$ $S = \{S\}$

P: $S \rightarrow aSb$ —— (1)

$S \rightarrow ab$ —— (2)

String $w = aaabbb$.

$S \xrightarrow{*} aSb$ (eq (1))

$S \rightarrow aaSbb$ (eq (1))

$S \rightarrow aaabbb$ (eq (2)).

Eg:-3] Construct a CFG for language

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

Sol:- CFG ; $G = \{N, T, P, S\}$.

$T = \{a\}$ $S = \{S\}$ $N = \{S\}$.

$G = \{\{S\}, \{a\}, \{S \rightarrow aa, S \rightarrow aaS\}, \{S\}\}$.

$S \longrightarrow aaS$ .... (1)

$S \longrightarrow aa$ .... (2)

String = w = aaaaaa

$$S \xrightarrow{*} aaS \quad (\text{eq } ①)$$
$$\rightarrow aaaaS \quad (\text{eq } ①)$$
$$\rightarrow aaaaaa \quad (\text{eq } ②)$$

Derivation

Eg:-4] Const CFG $L = \{w \mid w \text{ is odd no. of a's}\}$

Sol:- $G = \{N, T, P, S\}$.

$T = \{a\}$ $\quad S = \{S\}$ $\quad N = \{S\}$

① $G = \{\{S\}, \{a\}, \{S \rightarrow a, S \rightarrow aSa\}, \{S\}\}$.

② $G = \{\{S\}, \{a\}, \{S \rightarrow a, S \rightarrow aaS\}, \{S\}\}$

P: $S \longrightarrow a$ — ①

$S \longrightarrow aSa$ — ②

String w = aaaaa

$$S \xrightarrow{*} aSa$$
$$\longrightarrow aaSaa$$
$$\longrightarrow aaaaa$$

Eg:-5] Const CFG for $L = \{a^i b^j \mid i \neq j\}$.

$T = \{a, b\}$ $\quad S = \{S\}$ $\quad N = \{A, B\}$

$$G = \left\{\{S\}, \{a, b\}, \left\{\begin{array}{l} S \rightarrow aSb \mid A \mid B \\ A \rightarrow Aa \mid a \\ B \rightarrow Bb \mid b \end{array}\right\}, \{A, B, S\}\right\}$$

$S \rightarrow aSb \mid A \mid B$. —①

$A \rightarrow Aa \mid a$ —②

$B \rightarrow Bb \mid b$ —③

String = $w = a^3b^2$ = aaabb.

$S \overset{*}{\rightarrow} aSb$ — ①

$\rightarrow aaSbb$ —①

$\rightarrow aaAbb$ —①

$\rightarrow aaabb$ —②

Eg:-6] Const CFG generates " All positive even integers. upto 998.

Sol:- $w = \{0, 2, 4, 6, \cdots, 998\}$.

$S \rightarrow 0 \mid 2 \mid 4 \mid 6 \mid 8 \mid AB \mid AAB$

$A \rightarrow 0 \mid 1 \mid 2 \mid 3 \mid 4 \mid 5 \mid 6 \mid 7 \mid 8 \mid 9 \mid$

$B \rightarrow 0 \mid 2 \mid 4 \mid 6 \mid 8 \mid$.

wrong.

$S \rightarrow 0 \mid 2 \mid 4 \mid 6 \mid 8$

$S \rightarrow BA$

$B \rightarrow 1 \mid 2 \mid 3 \mid 4 \mid 5 \mid 6 \mid 7 \mid 8 \mid 9$.

$A \rightarrow 0 \mid 2 \mid 4 \mid 6 \mid 8$

$S \rightarrow BCA$

$C \rightarrow 0 \mid 1 \mid 2 \mid 3 \mid 4 \mid 5 \mid 6 \mid 7 \mid 8 \mid 9$.

Correct

7] Const CFG generates "Equal no. of a & b".

Sol:- $w = \{ab, ba, aabb, bbaa, abab, \dots\}$.

$S \longrightarrow aSb \mid bSa$

$S \longrightarrow \varepsilon$

$S \longrightarrow SS$

Production Rules.

$G = \{\{S\}, \{a, b, \varepsilon\}, \{\ \}, \{S\}\}$.

$w = abab$.

$S \rightarrow aSb \mid bSa \rightarrow ①$

$S \rightarrow \varepsilon \rightarrow ②$

$S \rightarrow SS \rightarrow ③$

Derivation:-

$S \longrightarrow SS$

$\longrightarrow aSbS$

$\longrightarrow aSbaSb$

$\longrightarrow aSbab$

$\longrightarrow abab.$

$w = abbbaa.$

LMD.

$S \xrightarrow{lm} SS$ — ③

$\xrightarrow{lm} aSbS$ — ①

$\xrightarrow{lm} abS$ — ②

$\xrightarrow{lm} abbSa$ — ①

$\xrightarrow{lm} abbbSaa$ — ①

$\xrightarrow{lm} abbbaa.$ — ②

RMD

$S \xrightarrow{rm} SS$

$\xrightarrow{rm} SbSa$

$\xrightarrow{rm} SbbSaa$

$\xrightarrow{rm} Sbbaa$

$\xrightarrow{rm} aSbbbaa$

$\xrightarrow{rm} abbbaa.$

Ambigous Grammar:-

* CFG is called Ambigous Grammar; iff there is W ⊂ L(G), such that w has (atleast) two different parse tree with respect to G.

* String W has multiple parse tree
* String W with multiple LMD trees
* String W with multiple RMD trees

P.Eg:-1] Let G; $S \to SbS \mid a$.

string w = abababa

Sol:- LMD:-

LM Parse Tree

Type:-1]

$S \xrightarrow{lm} SbS$

$\xrightarrow{lm} SbSbS$

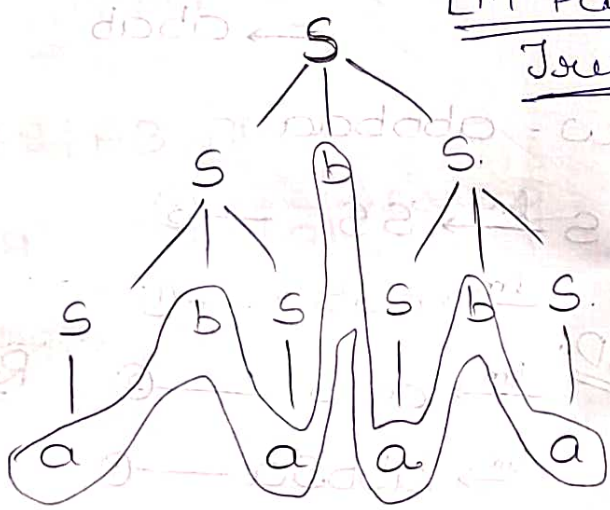
$\xrightarrow{lm} abSbS$

$\xrightarrow{lm} ababS$

$\xrightarrow{lm} ababSbS$

$\xrightarrow{lm} abababS$

$\xrightarrow{lm} ababab a$



leaf nodes ⇒

abababa

Type:-2]

$S \xrightarrow{lm} SbS$

$\xrightarrow{lm} abS$

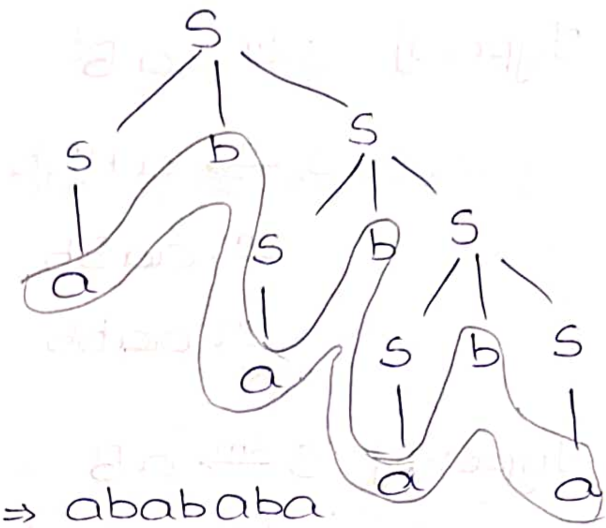
$\xrightarrow{lm} abSbS$

$\xrightarrow{lm} ababS$

$\xrightarrow{lm} ababSbS$

$\xrightarrow{lm} abababS$

$\xrightarrow{lm} abababa$



$\Rightarrow abababa$

$\therefore$ It has multiple parse Trees.

$\therefore$ Ambigous Grammar.

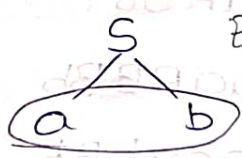
P.Eg:-2] $S \rightarrow aB | ab$

$A \rightarrow aAB | a$

$B \rightarrow ABb | b$ is ambigous Grammar.

Sol:- (i) $\omega = ab$.

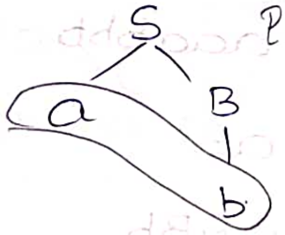
Type:-1] $S \rightarrow ab$



Parse Tree 1

Type:-2] $S \rightarrow aB$

$\rightarrow ab$.

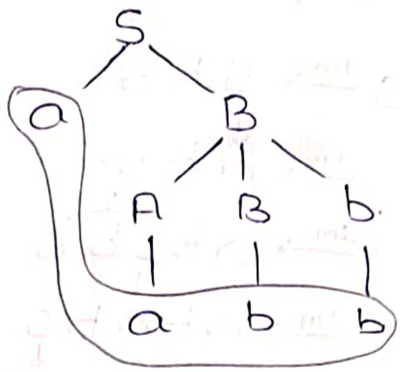


Parse Tree 2.

$\therefore$ Single string $\omega = ab$ it produces two different parse trees.

(ii) $w = aabb$

Type:-1] $S \xrightarrow{lm} aB$



$S \xrightarrow{lm} aABb$

$\xrightarrow{lm} aaBb$

$\xrightarrow{lm} aabb$

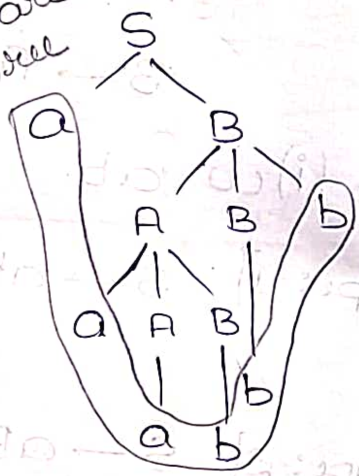
Type:-2] $S \xrightarrow{lm} aB$

$\xrightarrow{lm} aABb$.

(not produces).

(iii) $w = aaabbb$.

left parse Tree



Type :-1] $S \xrightarrow{lm} aB$

$\xrightarrow{lm} aABb$

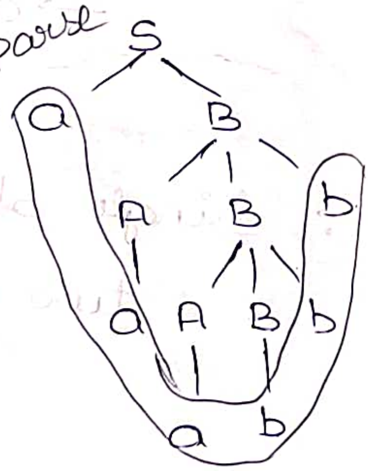
$\xrightarrow{lm} aaABBb$

$\xrightarrow{lm} aaabBb$

$\xrightarrow{lm} aaabbb$.

Type:-2] $S \xrightarrow{lm} aB$.

left parse Tree



$\xrightarrow{lm} aABb$

~~$\xrightarrow{lm} aAABb$~~

$\xrightarrow{lm} aaBb$

$\xrightarrow{lm} aaABbb$

$\xrightarrow{lm} aaabbb$.

$\therefore$ $w = aaabbb$ produces 2 parse Trees

Hence it is Ambigous Grammer.

Purpose of checking Ambigous Grammar:-

Used in Exp evalutation.

Eg:-3] $S \rightarrow a \mid Sa \mid bSS \mid SSb \mid SbS$ is ambigous

Sol:- $w = babaa$

Type:-1]

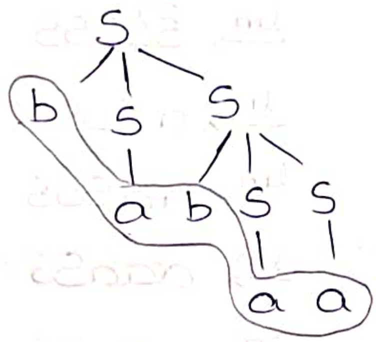
$S \xrightarrow{lm} bSS$

$\xrightarrow{lm} baS$

$\xrightarrow{lm} babSS$

$\xrightarrow{lm} babaS$

$\xrightarrow{lm} babaa$



Type:-2] ~~$S \xrightarrow{lm} b\,SSb$~~

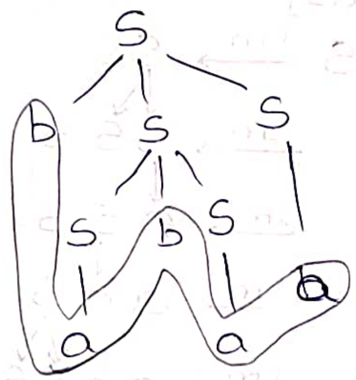
$S \xrightarrow{lm} bSS$

$\xrightarrow{lm} bSbSS$

$\xrightarrow{lm} babSS$

$\xrightarrow{lm} babaS$

$\xrightarrow{lm} babaa$



Prob:-4] $S \rightarrow SS \mid a$.

Sol:-] w = Any string of length of 'a' excluding 'ε'.

$= \{a, aa, aaa, aaaaa, \ldots\}$.

Type:-1] $w = aaaaa$.

$S \xrightarrow{lm} SS$

$\xrightarrow{lm} SSS$

$\xrightarrow{lm} SSSS$

$\xrightarrow{lm} SSSSS$

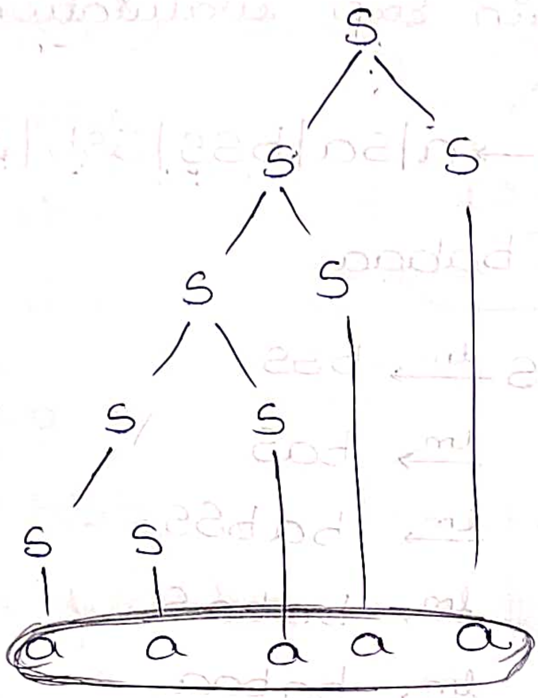
$\xrightarrow{lm} aSSSS$

$\xrightarrow{lm} aaSSS$

$\xrightarrow{lm} aaaSS$

$\xrightarrow{lm} aaaaS$

$\xrightarrow{lm} aaaaa$



Type:-2]

$S \xrightarrow{lm} SS$

$\xrightarrow{lm} SSS$

$\xrightarrow{lm} aSS$

$\xrightarrow{lm} aaS$

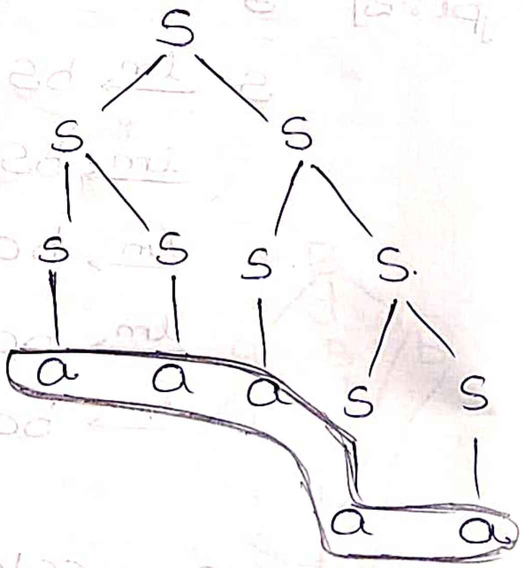
$\xrightarrow{lm} aaSS$

$\xrightarrow{lm} aaaS$

$\xrightarrow{lm} aaaSS$

$\xrightarrow{lm} aaaaS$

$\xrightarrow{lm} aaaaa$



∴ It is an ambigous Grammar

Unambigous Grammar:-

: String that produces unique parse tree

Ambigous $\longrightarrow$ Unambigous:-

Q:- Prev Eg.

Sol:- $S \rightarrow SS \mid a \Rightarrow$ Ambigous.

$L(G) = \{a^n \mid n \geq 1\}$

Convertion of amb $\rightarrow$ unamb also produces same language

$\underbrace{S \longrightarrow aS \mid a}_{\Downarrow}$ [Same lang as above].

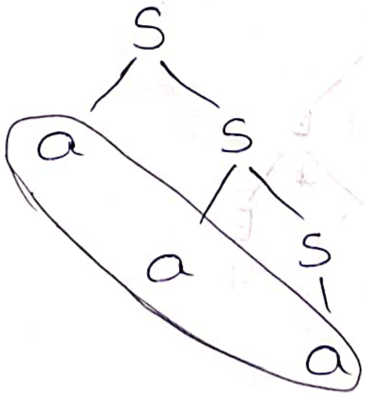
$\therefore$ Unambigous.

$w = aaa$

$S \longrightarrow aS$

$\longrightarrow aaS$

$\longrightarrow aaa$



$\therefore$ It produces only one parse tree for the grammer.

$\therefore$ $\boxed{\text{Unambigous Grammer}}$

Eg:- $G = \{N, T, P, S\}$

$P = \{ S \to \varepsilon,\ S \to SS,\ S \to (S) \}$

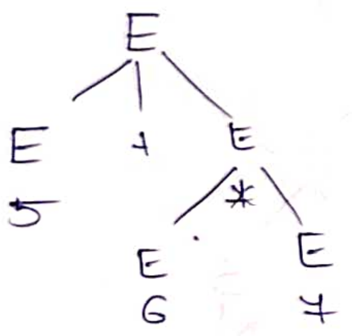
Sol:- $N = \{S\}$ $\quad T = \{ (,) \}$ $\quad S = \{S\}$.

$$
\begin{aligned}
S &\to SS \\
&\to (S) \\
&\to (())
\end{aligned}
$$

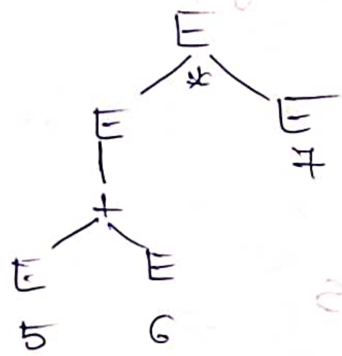
$L(G) = \{$ Set of all strings of balanced left and Right paranthesis $\}$

Eg:- $E \to E + E \mid E * E$ (Exp Grammar)

$w = E + E * E$



47.



77.

MODULE-4

PUSH DOWN AUTOMATA (PDA)

* Implimentation of RE we need Automata (NFA / DFA)
* Implimentation of CFG $\Rightarrow$ $\boxed{\text{Pushdown Automata PDA}}$

Definition :- PDA has 7 tuple machine

$P = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$
Q = Finite set of states Σ = Set of input symbols. Γ = Stack alphabet $\delta : Q \times (\Sigma \cup \varepsilon) \times \Gamma \Rightarrow$ ~~$Q \times \Gamma^*$~~ $\longrightarrow Q \times \Gamma^*$ = Transition Func q_0 = Start state Z_0 = Start sym of stack $F \subseteq Q$ = Set of accepting states Γ^* is string of stack sym

* In F.A ; δ has 2 sym . $(q_0, a) \longrightarrow q_2$
* In PDA ; $\boxed{\delta(q, a, x).}$

* $\delta(q, a, X)$

$q \Rightarrow$ State in Q.

$a \Rightarrow$ either input symbol in Σ

$a = \varepsilon$; where empty string, which is assumed not be an i/p symbol.

$X \Rightarrow$ stack symbol ; $X \in \Gamma$

---

* The output of δ is finite set of pairs

(p, γ)

$\therefore \boxed{\delta(q, a, X) = (p, \gamma)}$

p = new state

γ = string of stack sym. that replaces X at the top of stack.

- If $\gamma = \varepsilon \Rightarrow$ poped (stack is poped)

$\gamma = X \Rightarrow$ unchanged. (stack is unchanged)

$\gamma = yz \Rightarrow \boxed{X = Z}$ & y pushed on the stack.

---

* String Validation :- ID

ID = Instataneous Definition

DFA :- $\hat{\delta}(q_0, w) = q \in F$

NFA :- $\delta(q_0, w) = q \cap F \neq \phi$.

- PDA :-

① $(q_0, w) \rightarrow$ Final State

② $(q_0, w) \rightarrow \phi$ (Empty Stack)

} String accepted

PDA :- FA + STACK.

5 pupils + 2 pupils

$(Q, \Sigma, \delta, q_0, F) + (\Gamma, Z_0)$.

δ: operation :-

* It has 3 pupils

$$\delta(\underset{Q}{q}, \underset{\Sigma}{a}, \underset{\Gamma}{x}) = (p, \gamma)$$

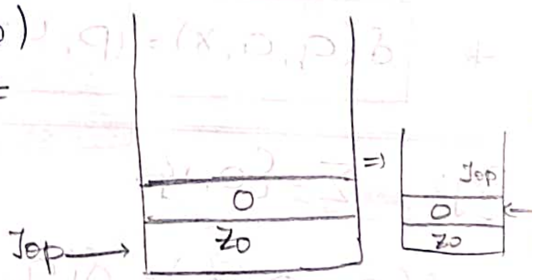
* γ values.

(i) $\gamma = \epsilon \Rightarrow$ stack poped.

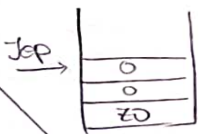
(ii) $\gamma = X \Rightarrow$ stack unchang

(iii) $\gamma = YZ$

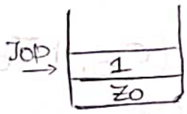
Eg:- $\delta(q_0, 0, \underset{X}{Z_0}) = \delta(q_0, \underset{YZ}{0Z_0})$



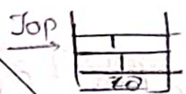
$\delta(q_0, 0, 0) = \delta(q_0, 00)$



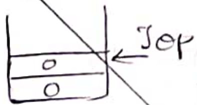
Eg:- $\delta(q_0, 1, Z_0) = (q_0, 1Z_0)$



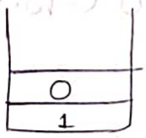
$\delta(q_0, 1, 1) = (q_0, 11)$



Eg:- $\delta(q_0, 0, 0) = (q_0, 00)$



$\delta(q_0, 0, \underset{X}{1}) = (q_0, \underset{YZ}{01})$

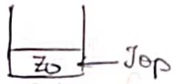


Eg:- $\delta(q_0, 0, \boxed{0}) = (q_0, 00)$ Top

$\delta(q_0, 0, \boxed{1}) = (q_0, 01)$

$\delta(q_0, 0, \boxed{Z_0}) = (q_0, 0Z_0)$

Push:-

* Initially Z_0 is there in stack 

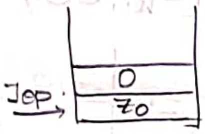
* $\delta(q, a, X) = (p, \gamma)$

$$\begin{bmatrix} \gamma = YZ \\ \gamma = \varepsilon \\ \gamma = X. \end{bmatrix}$$

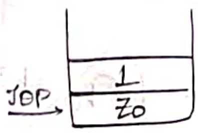
* $\boxed{\delta(q, a, X) = (p, YZ)}$

Eg:- $\Sigma = \{0, 1\}$.

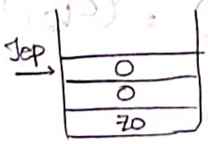
(i) $\delta(q_0, 0, Z_0) = \delta(q_0, 0Z_0)$



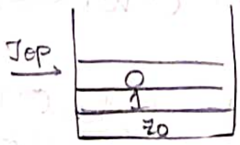
(ii) $\delta(q_0, 1, Z_0) = \delta(q_0, 1Z_0)$



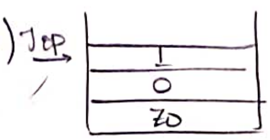
(iii) $\delta(q_0, 0, 0) = \delta(q_0, 00)$



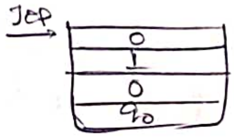
(iv) $\delta(q_0, 0, 1) = \delta(q_0, 01)$



(v) $\delta(q_0, 1, 0) = \delta(q_0, 10)$

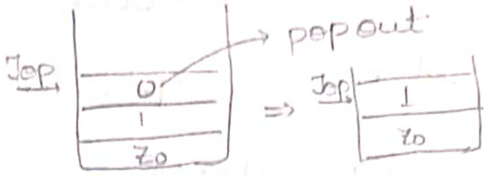


(vi) $\delta(q_0, 0, 1) = \delta(q_0, 01)$

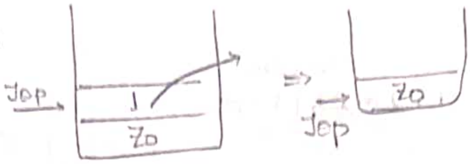


Pop:-

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



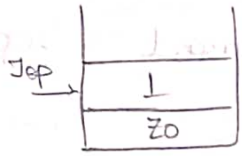
$\delta(q_0, 1, 1) = \delta(q_0, \varepsilon)$



$\delta(q_0, \varepsilon, Z_0) = \delta(q_0, \varepsilon)$

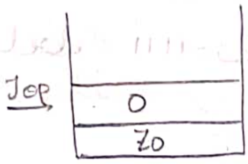
Unchanged:-

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

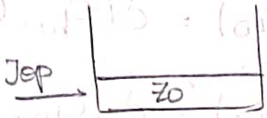


$\boxed{\delta(q_0, a, X) = (p, X)}$

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



$\delta(q_0, 1, Z_0) = (q_0, Z_0)$



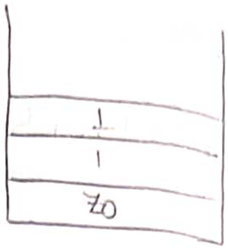
P.Eg:- 1] Design PDA for lang

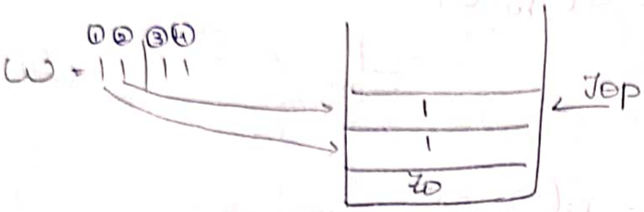
$L = \{ww^R\}$ over string $\{0, 1\}$.

Sol:-] $L = \{\underbrace{11}_{w}\underbrace{11}_{w^R}, \underbrace{10}_{w}\underbrace{01}_{w^R}, \underbrace{01}_{w}\underbrace{10}_{w^R}, \underbrace{001}_{w}\underbrace{100}_{w^R}, \ldots\}$.

For eg $w = \underbrace{11}_{w}\underbrace{11}_{w^R}$ accepted by this PDA

- First we have to push all element upto middle into the stack.

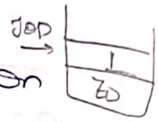




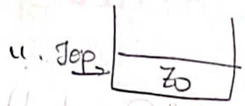
Methodology

- Remaining two elements:-

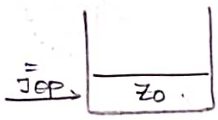
1] If 3^{rd} element = Top ⇒ pop operation



2] If 4^{th} element = Top ⇒ pop



∴ Final State =



∴ String w=1111 belongs to PDA. Ei valid.

Solution:

① $\delta(q_0, 0, Z_0) = \delta(q_0, 0Z_0)$

$\delta(q_0, 1, Z_0) = \delta(q_0, 1Z_0)$

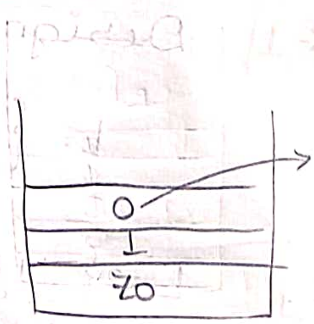
1001

② $\delta(q_0, 0, 0) = (q_0, 00)$

$\delta(q_0, 1, 0) = (q_0, 10)$

$\delta(q_0, 0, 1) = (q_0, 01)$

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



③ $\delta(q_0, \varepsilon, Z_0) = (q_1, Z_0)$

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

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

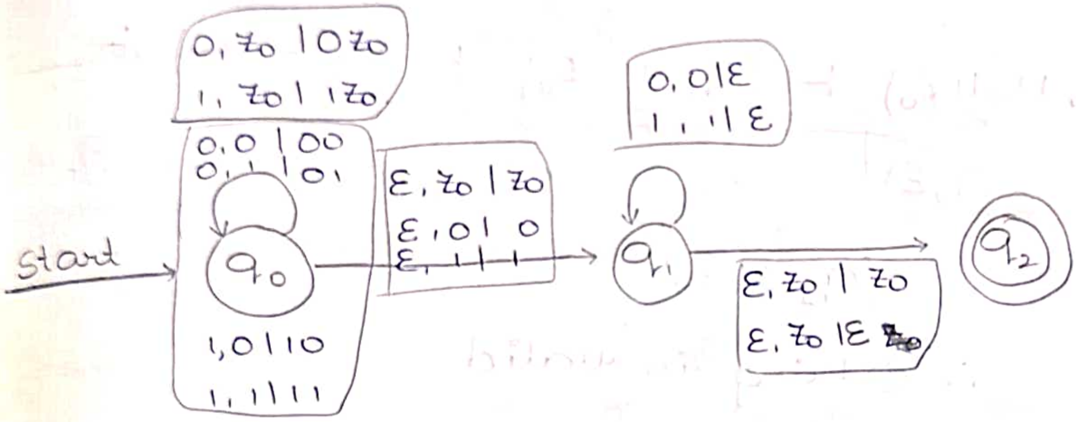
(11) $\delta(q_1, 0, 0) = (q_1, \varepsilon)$

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

(5) $\delta(q_1, \varepsilon, Z_0) = (q_2, Z_0)$

Accepting state.

$\delta(q_1, \varepsilon, Z_0) = (q_2, \varepsilon) \rightarrow$ [Empty stack].



Instantaneous Description (ID):-

* $\hat{\delta} \Rightarrow$ NFA | DFA

ID $\Rightarrow$ PDA

(we can take one sym at a time)

* 2 Types for validations:-

(i) reaches accepting state.

(ii) Stack empty

* PDA by a triple (q, w, γ) where

$q \Rightarrow$ state.

$w \Rightarrow$ remaining output

$\gamma \Rightarrow$ stack contents

Suppose $\delta(q, a, X)$ contains (p, α). Then for all strings w in Σ^* & β in Γ^*

$$\boxed{(q, aw, X\beta) \vdash (p, w, \alpha\beta)}$$

Proof for prev Example:-

$L = \{ww^R\}$.

$w = \underbrace{11}_{w}\ \underbrace{11}_{w^R}$

$(q_0, 1111, Z_0) \vdash (q_0, 111, 1Z_0) \vdash (q_0, 11, 11Z_0) \vdash$

$\delta(q_0, \underset{\text{inp}}{1}, \underset{\text{top}}{Z_0}) = (q_0, 1Z_0)$ | $\delta(q_0, 1, 1) = (q_0, 11)$ | $\delta(q_0, \varepsilon, 1) = (q_1, 1)$

$(q_1, 11, 11Z_0) \vdash (q_1, 1, 1Z_0) \vdash (q_1, \varepsilon, Z_0) \vdash$

$\delta(q_1, 1, 1) = (q_1, \varepsilon)$ | $\delta(q_1, 1, 1) = (q_1, \varepsilon)$ | $\delta(q_1, \varepsilon, Z_0) = (q_2, Z_0)$

(q_2, ε, Z_0)

$\therefore$ String is valid.

Another method:-

$(q_0, 1111, Z_0)$ → $(q_1, 1111, Z_0) \rightarrow (q_2, 1111, Z_0)$

↓

$(q_0, 111, 1Z_0)$ → $(q_1, 111, 1Z_0) \rightarrow (q_1, 11, 1Z_0)$ ↓ $(q_2, 11, Z_0)$

↓

$(q_0, 11, 11Z_0)$ → $(q_1, 11, 11Z_0) \rightarrow (q_1, 1, 1Z_0)$ ↓ (q_1, ε, Z_0) ↓ (q_2, ε, Z_0)

↓

$(q_0, 1, 111Z_0)$ → $(q_1, 1, 111Z_0)$ ↓ $(q_1, \varepsilon, 11Z_0)$ ✗

↓

$(q_0, \varepsilon, 1111Z_0)$

↓

$(q_1, \varepsilon, 1111Z_0)$

Acceptance by Final state:-

PDA = $\mathcal{E}$-NFA + STACK

(i) $\boxed{\{w \mid (q_0, w, Z_0) \vdash_P^* (q, \mathcal{E}, \alpha)\}}$ → any stack symbol

P(F) = accepted by reaching final state

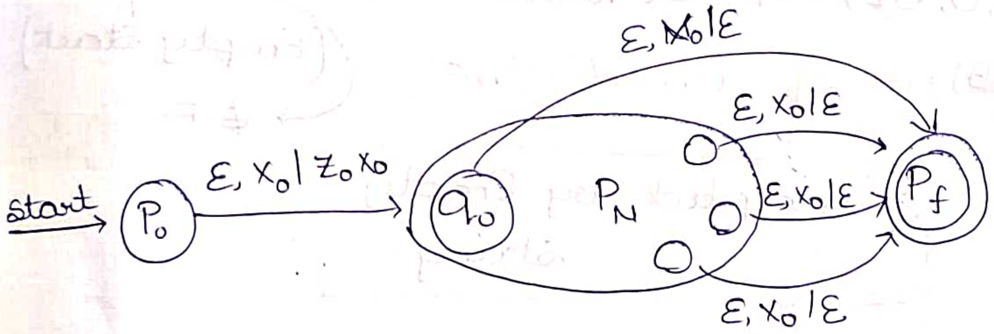
belongs to final state

(ii) $\boxed{\{w \mid (q_0, w, Z_0) \vdash^* (q, \mathcal{E}, \mathcal{E})\}}$ → must be '$\mathcal{E}$'

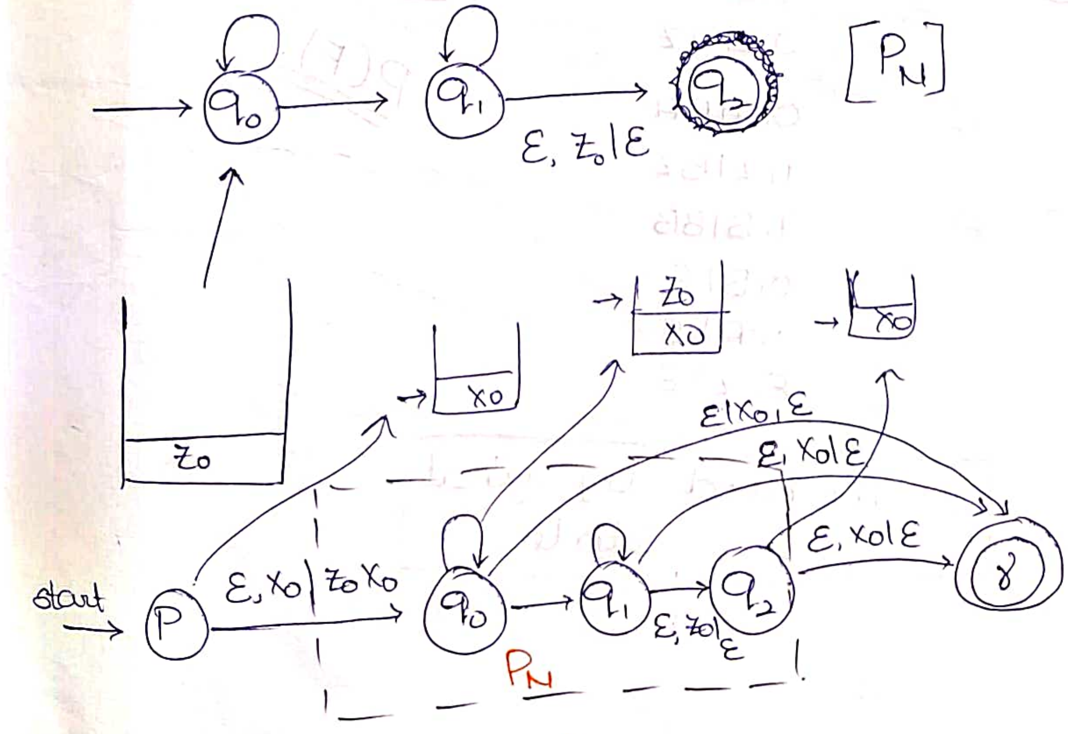
P(N) = accepting by empty stack

need not to be a final state

Eg:- Conversion of Empty stack to Final state:



Eg:- $P_N \longrightarrow P_F$ $L = \{ww^R\}$



P_F

Eg:-

0, Z | AZ

0, A | AA

1, Z | BZ

1, B | BB

P(N)

0, B | ε

1, A | ε

ε, Z | ε

ID:-

$\delta(q_0, \boxed{1}010, Z) \vdash \delta(q_0, \boxed{0}10, BZ) \vdash \delta(q_0, \boxed{1}0, Z) \vdash$

$(q_0, 1, Z) = (q_0, BZ)$ | $(q_0, 0, B) = (q_0, \varepsilon)$ | $(q_0, 1, Z) = (q_0, BZ)$

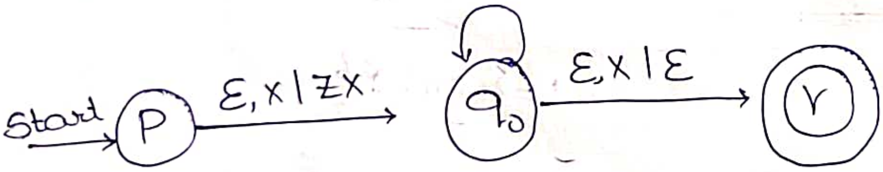
$\delta(q_0, 0, BZ) \vdash \delta(q_0, \varepsilon, Z) \vdash \delta(q_0, \varepsilon, \varepsilon)$

$(q_0, 0, B) = (q_0, \varepsilon)$ | $(q_0, \varepsilon, Z) = (q_0, \varepsilon)$

(Empty stack)

$\notin F$.

$\boxed{\therefore \text{ Accepted by Empty string.}}$



0, Z | AZ

0, A | AA

1, Z | BZ

1, B | BB

0, B | ε

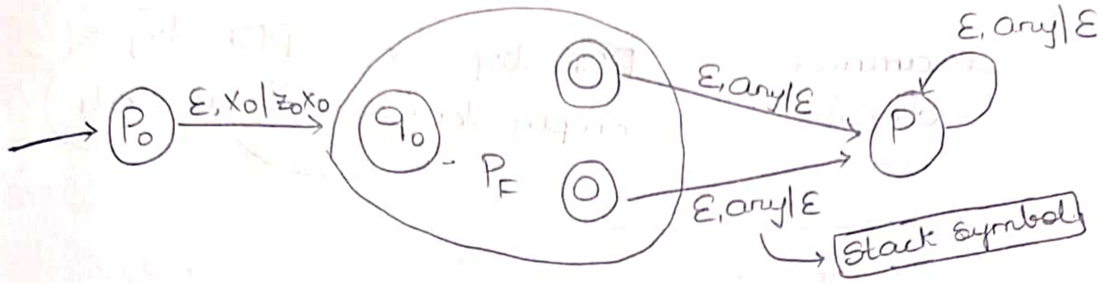
1, A | ε

ε, Z | ε

P(F).

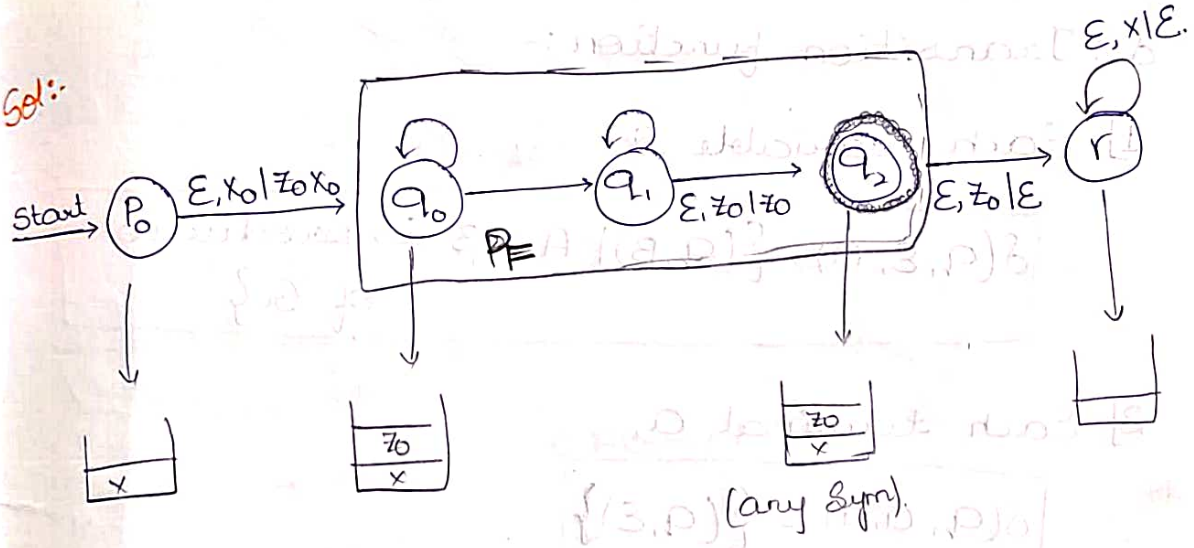
$\boxed{\therefore \text{ Accepted by final State}}$

Final state → Empty stack Conversion:-



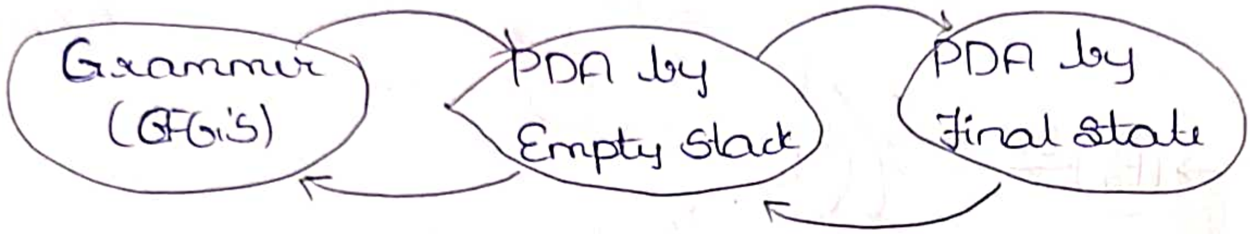
Eg:- $P_F \rightarrow P_N$

Sol:-



∴ Idea is P_N.

Equivalence of PDA's and CFG's :-



CFG ⟶ PDA :-

CFG ⇒ $G = \{V, T, Q, S\}$

PDA ⇒ P accepts L(G) by empty stack.

$P = \{\{q\}, T, V \cup T, \delta, q, S)$

δ : Transition function :-

1] Each variable A

$$\boxed{\delta(q, \varepsilon, A) = \{(q, \beta) \mid A \to \beta \text{ is production of } G\}}$$

2] Each terminal a,

$$\boxed{\delta(q, a, a) = \{(q, \varepsilon)\}}$$

* Only single state automata.

CFG → PDA(P_N):-

Ex:-1] $S \rightarrow 0S1 \mid A$

Sol:- $A \rightarrow 1A0 \mid S \mid \varepsilon$

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

$= \{\{S, A\}, \{0, 1\}, P, \{S\}\}$

(i) $\underset{A}{S} \rightarrow \underbrace{0S1}_{\beta_1} \mid \underbrace{A}_{\beta_2} \Rightarrow \delta(q, \varepsilon, S) = \{(q, 0S1), (q, A)\}$

$\underset{A}{A} \rightarrow \underbrace{1A0}_{\beta_1} \mid \underset{\beta_2}{S} \mid \underset{\beta_3}{\varepsilon} \Rightarrow \delta(q, \varepsilon, A) = \{(q_0, 1A0), (q, S), (q, \varepsilon)\}.$

(ii) $\delta(q, 0, 0) = \{(q, \varepsilon)\}$

$\delta(q, 1, 1) = \{(q, \varepsilon)\}$

Eg:-2] $S \rightarrow aAa$

$A \rightarrow Sb$ CFG → PDA

$A \rightarrow bCC$

$C \rightarrow abb.$

Sol:- $V = \{S, A, C\}$ $T = \{ab\}$ $S = \{S\}$

Variables:-

1] $S \rightarrow aAa \Rightarrow \delta(q, \varepsilon, S) = \{(q, aAa)\}$

2] $A \rightarrow Sb \Rightarrow \delta(q, \varepsilon, A) = \{(q, Sb)\}$

3] $A \rightarrow bCC \Rightarrow \delta(q, \varepsilon, A) = \{(q, bCC)\}$

4] $C \rightarrow abb \Rightarrow \delta(q, \varepsilon, C) = \{(q, abb)\}$

Terminals:-

(i) $\delta(q, a, a) = (q, \varepsilon)$

(ii) $\delta(q, b, b) = (q, \varepsilon)$

PDA(P_N) $\longrightarrow$ Grammar:-

(Single state automata.

Start Symbol.

$S \longrightarrow [q_0 Z_0 P]$, $P \in Q$

ini state, top of stack, no. of state

if $Q = \{q_0, q_1\}$;
$S \to [q_0 Z_0 q_0] | [q_0 Z_0 q_1]$.

Rule 1:

$$\boxed{\begin{aligned} &\delta(q, \underset{\text{input}}{a}, Z) = (p, \underset{\text{stack symbol}}{X}) \\ &[q Z \underset{\text{no. of states}}{r}] \longrightarrow a[p X r] \end{aligned}}$$

Rule 2:

$$\boxed{\begin{aligned} &\delta(q, a, Z) = (q, XZ) \\ &[q Z r] \longrightarrow a[q X \underset{\text{states}}{k}][k Z r] \end{aligned}}$$

$r, k \in Q$

Eg:- 1] Convert PDA $P = \{\{p, q\}, \{0, 1\}, \{X, Z\}, \delta, \{q\}, Z\}$

States, i/p, Stack Sym, δ, ini/ ~~final~~ State, Start Stack Sym.

Sum 6

Sol:- Start Symbol:-

$S \longrightarrow [q Z q] | [q Z p]$.

~~Rule 2~~:- ① $\delta(q, 1, Z) = (q, XZ)$

$[q Z q] \longrightarrow 1[q X q][q Z q] \to 1AS$

$[q Z q] \longrightarrow 1[q X p][p Z q] \to 1BD$

$[q Z p] \longrightarrow 1[q X q][q Z p] \to 1AS$

$[q Z p] \longrightarrow 1[q X p][p Z p] \to 1BE$

Rule 2:

$\delta(q, a, Z) = (q, XZ)$

$[q Z r] \longrightarrow a[q X k][k Z r]$.

② $\delta(q, 1, X) = (q, XX)$

$[qXq] \rightarrow 1[qXq][qXq]$

$[qXq] \rightarrow 1[qXp][pXq]$

$[qXp] \rightarrow 1[qXq][qZp]$

$[qXp] \rightarrow 1[qXp][pZp]$

③ $\delta(q, 0, X) = (p, X)$

$[qXq] \rightarrow 0[pXq]$

$[qXp] \rightarrow 0[pXp]$

$Z = X.$

$[qZr] \rightarrow \bar{a}[pXr].$

④ $\delta(q, \varepsilon, X) = (q, \varepsilon)$

$[qXq] \rightarrow \varepsilon$

⑤ $\delta(p, 1, X) = (p, \varepsilon)$

$[pXp] \rightarrow 1$

⑥ $\delta(p, 0, Z) = (q, Z)$

$[pZq] \rightarrow 0[qZq]$

$[pZp] \rightarrow 0[qZp]$

$S \rightarrow [qZq][qZp]$

$A \rightarrow [qXq]$

$B \rightarrow [qXp]$

$C \rightarrow [pXp]$

$D \rightarrow [p\cancel{X}^{Z}q]$

$E \rightarrow [pZp]$

$F \rightarrow [qZq].$

① $S \rightarrow 1AS \mid 1BD \mid 1AS \mid 1BE$

② $S \rightarrow 1AA \mid 1BF \mid 1AS \mid 1SE$

③ $A \rightarrow 0F \quad B \rightarrow 0C$

④ $A \rightarrow \varepsilon$

⑤ $C \rightarrow 1$

⑥ $D \rightarrow 0S, \; E \rightarrow 0S.$

Deterministic PDA:-

1] $\delta(q, a, X)$ has almost one member for any q in Q, a in Σ (or) $a = \epsilon$ & X in Γ

2] If $\delta(q, a, X)$ is nonempty, for some a in Σ, then $\delta(q, \epsilon, X)$ must be empty.

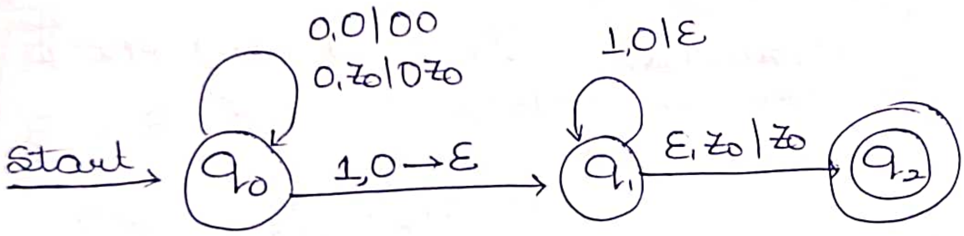
Eg:- $L = \{ww^R\} \Rightarrow$ NPDA

$L = \{wcw^R\} \Rightarrow$ DPDA

Push centre Pop

* DPDA & NPDA both accept ε-Transition

Eg:- $L = \{0^n 1^n, n \geq 1\}$



$w = 0011$

$(q_0, 0011, Z_0) \vdash (q_0, 011, 0Z_0) \vdash (q_0, 11, 00Z_0) \vdash$

$(q_1, 1, 0Z_0) \vdash (q_1, \epsilon, Z_0) \vdash (q_2, \epsilon, Z_0)$

Simplification of Grammar.

* Elimination of useless symbol
  (or)
* Const of Reduced grammar
* Elimination of ε/null production
* Elimination of unit prod.

For eg:- $G = \{N, T, P, S\}$

$$P: \{ S \to AB$$
$$A \to a$$
$$B \to b$$
$$B \to C \quad \text{useless.}$$
$$E \to c \mid \varepsilon \}$$

$w \in T \notin N.$

$$\boxed{\begin{array}{l} A \to B \text{ (unit prod)} \\ S \to \varepsilon \text{ (null}/\varepsilon \text{ prod)} \end{array}}$$ we have to remove

$G' = \{N', T', P', S\}.$

$P' \Rightarrow \{ S \to AB$ $\quad N' = \{S, A, B\}$
$A \to a$ $\quad T' = \{a, b\}.$
$B \to b.$
$\}.$

Theorem 1:- Const of Reduced Grammar:-

$G = \{N, T, P, S\}$ $\quad G' = \{N', T', P', S\}.$

Step-1] Constr of N':-

Grammar G' is constructed equi to G. every variable in G' derives some terminal string.

Step-2] Const of P':-

Every sym in N' should derive sentential form to reach the terminal string are considered as productions of G'.

$$S \overset{*}{\Rightarrow} \alpha X \beta \overset{*}{\Rightarrow} w \quad w \in T^*$$

P.Eg:- Let $G = \{N, T, P, S\}$

$S \to aS|A|C$

$A \to a$

$B \to aa$ Eliminate useless sym.

$C \to aC$

Sol:- (i) Finding N'

$W_1 = \{A, B\}$ [which has terminals (small letters) in right side]

$W_2 = \{W_1 \cup A_1 \to \alpha,\ \alpha \in (W_1 \cup T)^*\}$

$(A, B, a)^* \Rightarrow$ we have to take rules which A, B, a lies right side

$W_2 = \{S, A, B\}$

$W_3 = \{W_2 \cup A_1 \to \alpha,\ \alpha \in (W_2 \cup T)^*\}$

$(S, A, B, a)^*$

$W_3 = \{S, A, B\}$

$$\Rightarrow \boxed{\begin{array}{l} S \to aS|A \\ A \to a \\ B \to aa \end{array}}$$

(ii) Finding P' :-

$W_1 = \{S\}$ $S \to aS|A$

$W_2 = \{a, A, S\}$ $A \to a$

$W_3 = \{S, a, A\}$

$\therefore$ $\boxed{\begin{array}{l} S \to aS|A \\ A \to a \end{array}}$ $\Rightarrow$ Simplified CFG.

Theorum: 2] Elimination of Null production:-

- If a CFG production has a $A \to \varepsilon$ (null prod) can eliminated where A is non Terminal.

$$\boxed{L(G') = L(G) \sim \varepsilon}$$

Definition :-

- A variable (A) in a CFG is nullable if

$$\boxed{A \xrightarrow{*} \varepsilon}$$

Eg:-
- $S \to \varepsilon$ [$\therefore$ S is nullable].
- $S \xrightarrow{*} A$, $A \to \varepsilon$ — Nullable = $\{S, A\}$.
- $S \to A$, $A \to B$, $B \to \varepsilon$ — Nullable = $\{S, A, B\}$

- Any Production of CFG of form $A \to \varepsilon$ is calle[d] ε- Production.

Steps :-

1] Find set of nullable variables of G ; for all productions of form

$A \rightarrow \varepsilon$; put A into nullable

if $B \rightarrow A_1 A_2 \dots A_n$

where $A_1, A_2, \dots A_n$ are nullable then put B is also Nullable

2] Const a new set of Production P'.

P.Eg :- Eliminate ε-prod from grammar G.

$S \rightarrow abB$

$B \rightarrow Bb \mid \varepsilon$

Sol :- B produces ε ⇒ B is Nullable

step :- 1] $N_{nullable} = \{B\}$.

step :- 2] Construct P'

$$\left.\begin{aligned} S &\rightarrow abB \mid ab \\ B &\rightarrow Bb \mid b \end{aligned}\right\} \Rightarrow P^1$$

P.Eg :- Eliminate ε-production from grammar G

$S \rightarrow BabC$

$C \rightarrow b \mid \varepsilon$

$B \rightarrow a \mid \varepsilon$

Sol:- Step-1] $N_{nullable} = \{B, C\}$

both(ε)

Step-2] $S \rightarrow BabC \mid abC \mid Bab \mid ab$

$C \rightarrow b$

$B \rightarrow a$

P.Eg:- Eliminate ε-prod from grammar G

$S \rightarrow bS \mid AB$

$A \rightarrow \varepsilon$

$B \rightarrow \varepsilon$

$D \rightarrow a$

Sol:- Step-1] $N_{nullable} = \{A, B\}$

Step-2] $S \rightarrow bS \mid B \mid A \mid AB$

$D \rightarrow a$

Theorem-3] Removal of Unit Productions:-

Def:- A prod in CFG of form $A \rightarrow B$ where $A, B \in N$ is called Unit Production

Step-1] For each variable A & B such that

$A \stackrel{*}{\Rightarrow} B$

The new grammar G' is generated by

(i) Putting all non unit productions of P into P'.

(ii) For all A & B if $A \stackrel{*}{\Rightarrow} B$.

add $A \rightarrow x_1 x_2 \cdots x_n$.

$B \rightarrow x_1 x_2 \cdots x_n$

where x_i should not be single variable

P.Eg:- Consider prod rule and eliminate unit

P:- $S \to bS \mid A$

$A \to a \mid B$

$B \to aa$

Sol:- Unit prod :- $S \to A$

$A \to B$.

(i) $P' = \left\{ \begin{array}{l} S \to bS \\ A \to a \\ B \to aa \end{array} \right\}$ all non-unit prod rules

(ii) $S \to A$ X removed $\Rightarrow S \to a \mid B$.

$A \to B$ X removed $\Rightarrow S \to a \mid aa$

$A \to a \mid aa$

$$\therefore \boxed{\begin{array}{l} P': \; S \to bS \mid a \mid aa \\ \quad A \to a \mid aa \\ \quad B \to aa \end{array}}$$

P.Eg:- $S \to AaB \mid C$ Remove all unit prod

$A \to a \mid bc \mid B$

$C \to a$

$B \to A \mid bb$

Sol:- Unit production :- $S \to C$

$A \to B$

$B \to A$

(i) p' = $S \to AaB$

$A \to a|bc$

$C \to a$

$B \to bb$

(ii) $S \to C \Rightarrow S \to a$

$A \to B \Rightarrow A \to bb$

$B \to A \Rightarrow B \to a|bc$

$$\therefore \boxed{\begin{aligned} p':- \quad & S \to AaB|a \\ & A \to a|bb|bc \\ & C \to a \\ & B \to bb|a|bc \end{aligned}}$$

P.Eg:- Remove the unit prod

$S \to AB$

$A \to a$

$B \to C$

$B \to b$

$C \to D$

$D \to E$

$E \to a$

Sol:- Unit prod :- $B \to C$ $C \to D$ $D \to E$

p' : $S \to AB$	$B \to C$	$C \to D$	$D \to E$
$A \to a$	$B \to D$	$D \to E$	$D \to a$
$B \to b$	$B \to E$	~~$E \to a$~~	
$E \to a$	$B \to a$	$C \to a$	

$$\therefore \boxed{\begin{aligned} p' \Rightarrow \quad & S \to AB \qquad D \to a \\ & A \to a \qquad E \to a \\ & B \to a|b \\ & C \to a \end{aligned}}$$

4. Eg of Reduced grammar:-

Const Reduced grammar equivalent to gramma

P: $S \rightarrow AC \mid B$

$A \rightarrow a$

$C \rightarrow c \mid BC$

$E \rightarrow aA \mid e$

Sol:- (i) $T = \{a, c, e\}$

$W_1 = \{A, C, E\}$

$W_2 = \{S, A, C, E\}$

$W_3 = \{S, A, C, E\}$

$\Rightarrow G' = \{(A, C, E, S), (a, c, e), P', (S)\}$

P':
$S \longrightarrow AC$

$A \longrightarrow a$

$C \longrightarrow c$

$E \longrightarrow aA \mid e$

(ii) $\omega_1 = \{S\}$

$\omega_2 = \{S, A, C\}$

$\omega_3 = \{S, A, C, a, c\}$

$\omega_4 = \{S, A, C, a, c\}$

$\therefore$ Simplified Grammar.

$G'' = \{(A, C, S), \{a, c\}, P, \{S\}\}$

P: $S \rightarrow AC$

$A \rightarrow a$

$C \rightarrow c$

1. Eg of Removal of Unit Productions:-

$$
\begin{aligned}
P:\ & S \to XY \\
& X \to a \\
& Y \to Z \mid b \\
& Z \to M \\
& M \to N \\
& N \to a
\end{aligned}
$$

Sol:-

$$
\left.\begin{array}{l} Y \to Z \\ Z \to M \\ M \to N \end{array}\right\} \text{Unit prod.}
$$

(i) $Y \to Z$ $\quad$ $Z \to M$ $\quad$ $M \to N$

$Y \to M$ $\quad$ $Z \to N$ $\quad$ $\therefore M \to a$

$Y \to N$ $\quad$ $Z \to N$

$Y \to a$ $\quad$ $\therefore Z \to a$

$\therefore Y \to a$

$$
\begin{aligned}
\therefore\ P':\ & S \to XY \\
& X \to a \\
& Y \to a \mid b \\
& Z \to a \\
& M \to a \\
& N \to a
\end{aligned}
$$

4. Eg of Removal of Null Productions:-

$S \to ABAC \quad A \to aA \mid \varepsilon \quad B \to bB \mid \varepsilon \quad C \to c$

Sol:- $N_{nullable} = \{A, B\}$

$$
\begin{aligned}
S &\to ABAC \mid BC \mid AAC \mid ABC \mid AC \mid C \\
A &\to aA \mid a \\
B &\to bB \mid b \\
C &\to c
\end{aligned}
$$

MODULE-5

Normal forms for CFG's.

* It is simplifing the CFG.
* Normal forms are used to reduce the complexity of a CFG.
* This CNF & GNF has minimum stack operal

max no. of stack in CNF=2

* Min no. of stack operation:-

If we have min no. of stack operation then we can reduce time ~~com~~ complexity means running time of prog is reduced.

Chomsky Normal Form [CNF]:-

A CFG G is in CNF of every production is of form

$$\boxed{\begin{array}{l} A \longrightarrow a \\ A \longrightarrow BC \end{array}} \qquad \begin{array}{l} A, B, C \in N \\ a \in T \end{array}$$

$S \rightarrow \varepsilon$ is in G if $\varepsilon \in L(G)$

when ε is in $L(G)$ we assume that S does not appear on any RHS productio

Reducing CFG to CNF :-

Step-1:-

→ Eliminate all ε-productions

→ Eliminate unit productions

→ Eliminate useless symbols

Step-2:-

→ After performing the step-1; if all the productions in G is in the form

$$\boxed{\begin{array}{c} A \rightarrow BC \text{ or } A \rightarrow a \\ \therefore G = CNF \end{array}}$$

otherwise perform following steps

Step-3:-

$A \rightarrow A_1 A_2 A_3 \cdots A_n$ $[n \geq 3]$

→ Now Introduce new productions & varia

→ Restrict the length of RHS as 2 NonTer

(or) a single terminal such as

$A \rightarrow A_1 D_1$

$D_1 \rightarrow A_2 D_2$

$D_2 \rightarrow A_3 D_3$

$D_3 \rightarrow A_4 D_4$

⋮

$\boxed{D_{m-1} \rightarrow A_m D_m}$

$A \rightarrow ab$

$D_1 \rightarrow a$

$D_2 \rightarrow b$

$A \rightarrow D_1 D_2$

Step-4:-

→ If $A \rightarrow A_1 A_2 \cdots A_n$ $[A_i = \text{terminals}]$

→ Introduce new variables such that

$\boxed{D_i \rightarrow t}$ where $\boxed{D_i \in N}$

P.Eg:-1] Convert the following CFG to CNF.

$S \to AAC$

$A \to aAb|\epsilon$

$C \to aC|a$

Sol:- Step:-1] ε-prod

Nullable = {A}

$P' = S \to AAC|AC|C$

$A \to aAb|ab.$

$C \to aC|a.$

Unit prod:-

~~There are no Unit Prod~~

$S \to C$

$C \to a|aC$

$\therefore \boxed{S \to a|aC}.$

$\therefore P' = S \to AAC|AC|a|aC$

$A \to aAb|ab$

$C \to aC|a$

Reduced Grammar:-

There are no useless symbols.

Proof:-

$N' \Rightarrow w_1 = \{S, A, C\}$

$P' \Rightarrow w_1 = \{S\}$

$w_2 = \{S, A, C, a\}$

$\therefore$ no useless symbols.

$\therefore S \to AAC|AC|aC|a$

$A \to aAb|ab$

$C \to aC|a.$

Step 2:- $P_1: S \rightarrow AAC$

$\therefore \boxed{\begin{array}{l} S \rightarrow AD_1 \\ D_1 \rightarrow AC \end{array}}$

$P_2: \boxed{S \rightarrow AC}$

$P_3: S \rightarrow aC$

$\boxed{\begin{array}{l} D_2 \rightarrow a \\ S \rightarrow D_2C \end{array}}$

$P_7: C \rightarrow aC$

$\boxed{C \rightarrow D_2C}$

$P_8: \boxed{C \rightarrow a}$

$P_4: \boxed{S \rightarrow a}$

$P_5: A \rightarrow aAb$

$D_4 \rightarrow b$

$A \rightarrow D_2 A D_4$

$D_3 \rightarrow A D_4$

$\boxed{A \rightarrow D_2 D_3}$

$P_6: A \rightarrow ab$

$\boxed{A \rightarrow D_2 D_4}$

CNF :-

P.Eg:-

$S \rightarrow abAB$

$A \rightarrow bAB \mid \varepsilon$

$B \rightarrow BAa \mid A \mid \varepsilon$.

Step-1] ε-prod.

$N_{nullable} = \{A, B\}$

P': $S \rightarrow abAB \mid abB \mid abA \mid ab$

$A \rightarrow bAB \mid bB \mid bA \mid b$

$B \rightarrow a \mid Aa \mid Ba \mid BAa \mid$

Unit Prod:- No unit Productions

Reduced:- $W_1 = \{S, A, B\}$.

$\therefore P' \Rightarrow S \rightarrow abAB \mid abB \mid abA \mid ab$

$A \rightarrow bAB \mid bB \mid bA \mid b$

$B \rightarrow a \mid Aa \mid Ba \mid BAa$

(1) $S \to abAB$

$$\boxed{\begin{aligned} &S \to D_1 D_2 AB \\ &S \to D_3 D_4 \\ &D_3 \to D_1 D_2 \\ &D_4 \to AB \\ &D_1 \to a \\ &D_2 \to b \end{aligned}}$$

(2) $S \to abA$

$$\boxed{\begin{aligned} &S \to D_1 D_2 A \\ &S \to D_3 A \end{aligned}}$$

(3) $S \to abB$

$$\boxed{S \to D_3 B}$$

(4) $S \to ab$

$$\boxed{S \to D_1 D_2}$$

(5) $A \to bAB$

$$\boxed{A \to D_2 D_4}$$

(6) $A \to bA$

$$\boxed{A \to D_2 A}$$

(7) $A \to bB$

$$\boxed{A \to D_2 B}$$

(8) $\boxed{A \to b}$

(9) $\boxed{B \to a}$

(10) $B \to Aa$

$$\boxed{B \to AD_1}$$

(11) $B \to Ba$

$$\boxed{B \to BD_1}$$

(12) $B \to BAa$

$$\boxed{\begin{aligned} &D_5 \to BA \\ &B \to D_5 D_1 \end{aligned}}$$

Griebach Normal Form
(GNF)

* Context Free Grammar G is reduced to GNF of every production of form

$\boxed{A \to a\alpha}$ a = terminal
α = Non-term | ε

if $\alpha = \varepsilon \Rightarrow A \to a$

* It consists of two lemma.

Lemma-1]:-

→ $G = \{N, T, P, S\}$ is a CFG & having production as $\boxed{A \to \cancel{\alpha_1 B}\ \alpha_1 B \alpha_2}$

$B = \beta_1 | \beta_2 | \beta_3 | \cdots | \beta_n$.

G' can construct by replacing B by prod

$L(G) = L(G_1)$

Lemma:2]:-

→ $G = \{N, T, P, S\}$ is a CFG; have productions

$\boxed{A \to A\alpha_1 | A\alpha_2 | \cdots | A\alpha_m | \beta_1 | \beta_2 | \cdots | \beta_n}$

* Introduce a new variable (X).

Let $G_1 = (V, T, P, S) \Rightarrow V_1 = V \cup \{X\}$.

$$\boxed{\text{① } \left.\begin{array}{l} A \to \beta_i \\ A \to \beta_i X \end{array}\right\} 1 \le i \le n}$$

$$\boxed{\text{② } \left.\begin{array}{l} X \to \alpha_i \\ X \to \alpha_i X \end{array}\right\} 1 \le i \le n}$$

$$\boxed{\text{GNF:- } \begin{array}{l} A \to a \\ A \to a\underbrace{A_1 A_2 B S}_{\in N|V} \end{array}}$$

Lemma 1:

Eg:- $A \rightarrow \alpha_1 B \alpha_2$

$A \rightarrow B \alpha_1 \alpha_2$

$B \rightarrow \beta_1 | \beta_2 | \cdots | \beta_m$

$A \rightarrow \alpha_1 \beta_1 \alpha_2 | \alpha_1 \beta_2 \alpha_2 | \cdots | \alpha_1 \beta_m \alpha_2$

$A \rightarrow \beta_1 \alpha_1 \alpha_2 | \beta_2 \alpha_1 \alpha_2 | \cdots | \beta_m \alpha_1 \alpha_2$

Eg:- $A \rightarrow A_3 B | A_3 a | b$

$A_3 \rightarrow abA | aS$

$A \rightarrow abAB | abAa | aSB | aSa | b$

Lemma 2:

Eg:- $A \rightarrow \underline{A\alpha_1} | A\alpha_2 | \cdots | A\alpha_m | \beta_1 | \beta_2 | \cdots | \beta_n$

$A_1 \rightarrow A_1 \overset{\alpha_1}{\boxed{aS}} | A \overset{\alpha_2}{\boxed{b}} | A_1 \overset{\alpha_3}{\boxed{AB}} | \overset{\beta_1}{aSa} | \underset{\beta_2}{ab} | \underset{\beta_3}{BA_1A_2}$

Rules:-

1] $A \rightarrow \beta_i$ $\quad A \rightarrow \beta_i X$

2] $X \rightarrow \alpha_i$ $\quad X \rightarrow \alpha_i X.$

X = Non-terminal

$A_1 \rightarrow aSa | ab | BA_1A_2$

$\Rightarrow A_1 \rightarrow aSaX | abX | BA_1A_2X.$

$X \rightarrow aS | b | AB$

$\Rightarrow X \rightarrow aSX | bX | ABX$

Conversion of CFG to GNF:-

* To Reduce a Grammar into GNF

① Convert CFG to CNF. [CFG $\rightarrow$ CNF $\rightarrow$ GNF

② Rename variables in V as $\{A_1, A_2, \cdots, A_n\}$ with start symbol as A_1.

Step-1] Modify productions such that

$A_i \rightarrow A_j \}$ where $i < j$.

Step-2]

→ If $A_k \rightarrow A_j \gamma$ is production with $j < k$; general a new set of productions by substituting for A_j, the right hand side of A_j production according to Lemma 1

→ By replacing this we obtain the productions of form;

$$A_k \rightarrow A_l^{\gamma} \; ; \; l \geq k$$

[* If $l = k$ are replaced acc to Lemma 2 by introducing new variable]

Convertion of GNF:-

① CFG → CNF → GNF

② Eg: $\overset{A_1}{S} \rightarrow \overset{A_2}{A}\overset{A_3}{B}$

$A \rightarrow a | aS$

$B \rightarrow b$

$\underline{A_1}^{i} \rightarrow \underline{A_2}^{j} A_3$ [i<j]

$A_2 \rightarrow a | aA_1$

$A_3 \rightarrow b$

Eg:- $\underline{A_2}^{i} \rightarrow \underline{A_1}^{j} A_3$ [$i \nless j$ ⇒ we use lemma 1]

Eg:- $\underline{A_1}^{i} \rightarrow \underline{A_2}^{j} A_3$ (i<j) ✓

$\underline{A_3}^{i} \rightarrow \underline{A_4}^{j} A_5 A_2$ (i<j) ✓

Eg:- $\underline{A_2}^{i} \rightarrow \underline{A_1}^{j} A_3$ [$i \nless j$ ⇒ Sub A_1 by prod rule and recheck whether (i<j)]

$A_4 \rightarrow A_2 A_1 A_2$

Prob :- 1] Construct GNF for given G.

$S \to AA \mid a$

$A \to SS \mid b$

Sol :- (1) Given grammar G is in CNF format.

Rename the variable $S \to A_1$

$A \to A_2$

$P: A_1 \to A_2A_2 \mid a$

$A_2 \to A_1A_1 \mid b$

(2) Prod rule (1) ⇒ $\underline{A_1^i} \to \underline{A_2^j}A_2 \mid a$ $(i<j) \Rightarrow$ no char

Prod rule (2) ⇒ $\underline{A_2^i} \to \underline{A_1^j}A_1 \mid b$ $(i<j) \Rightarrow$ False.

∴ we use Lemma 1 for rule 2.

By Lemma 1; $\underline{A_2^i} \to \underline{A_2^j}A_2A_1 \mid aA_1 \mid b$

(3) $\underline{A_2^i} \to \underline{A_2^j}A_2A_1 \mid aA_1 \mid b$ $(i<j) =$ False

$(i=j) \Rightarrow$ we apply Lemma 2.

$A_2 \to A_2\overset{\alpha_1}{\boxed{A_2A_1}} \mid \underbrace{aA_1}_{B_1} \mid \underbrace{b}_{B_2}$ [new variable is B_2 $(X=B_2)$]

Lemma 2 ⇒ $A_2 \to aA_1 \mid b$

$A_2 \to aA_1B_2 \mid bB_2$

∴ $\boxed{A_2 \to aA_1 \mid b \mid aA_1B_2 \mid bB_2}$ ⇒ GNF format

$B_2 \to A_2A_1$

$B_2 \to A_2A_1B_2$

∴ $\boxed{B_2 \to A_2A_1 \mid A_2A_1B_2}$

Here in (3) all prod rule satisfing GNF

(4) All the productions of A_2 in form of GNF

[Sub prod A_2 in A_1]

$A_1 \rightarrow A_2A_2 \mid a$

GNF

$$\boxed{A_1 \rightarrow aA_1A_2 \mid bA_2 \mid aA_1B_2A_2 \mid bB_2A_2 \mid a}$$

(5) All productions of A_1 are in GNF

[Sub prod A_2 & A_2 in B_2]

$B_2 \rightarrow A_2A_1 \mid A_2A_1B_2$

GNF

$$\boxed{B_2 \rightarrow aA_1A_1 \mid bA_1 \mid aA_1B_2A_1 \mid bB_2A_1 \mid aA_1A_1B_2 \mid bA_1B_2 \mid aA_1B_2A_1B_2 \mid bB_2A_1B_2}$$

$\therefore B_2$ is in GNF.

$\therefore$ GNF:-

$A_1 \rightarrow aA_1A_2 \mid bA_2 \mid aA_1B_2A_2 \mid bB_2A_2$

$A_2 \rightarrow aA_1 \mid b \mid aA_1B_2 \mid bB_2.$

$B_2 \rightarrow aA_1A_2 \mid bA_1 \mid aA_1B_2A_1 \mid bB_2A_1 \mid aA_1A_2B_2 \mid$
$bA_1B_2 \mid aA_1B_2A_1B_2 \mid bB_2A_1B_2.$

Prob:-2]
$S \rightarrow AB$
$A \rightarrow BS \mid b$
$B \rightarrow SA \mid a$

Sol:-

$$\left.\begin{array}{l} S \rightarrow A_1 \\ A \rightarrow A_2 \\ B \rightarrow A_3 \end{array}\right\} \Rightarrow \begin{array}{l} A_1 \rightarrow A_2A_3 \\ A_2 \rightarrow A_3A_1 \mid b \\ A_3 \rightarrow A_1A_2 \mid a \end{array}$$

① $A_1 \longrightarrow A_2A_3$

$i=1 \quad j=2 \Rightarrow (i<j) \Rightarrow$ True

② $A_2 \longrightarrow A_3A_1 | b$

$i=2 \quad j=3 \Rightarrow (i<j) \Rightarrow$ True

③ $A_3 \longrightarrow \underline{A_1}A_2 | a \Rightarrow (i \not< j) \Rightarrow$ False

$A_3 \rightarrow \underline{A_2}A_3A_2 | a$

$A_3 \rightarrow A_3A_1\underline{A_3A_2} | a | b\underline{A_3A_2}$

$(i=j)$

$A_3 \longrightarrow A_3 \underbrace{A_1A_2A_2}_{\alpha_1} | \underbrace{a}_{\beta_1} | \underbrace{bA_3A_2}_{\beta_2}$

* $\boxed{A_3 \rightarrow \cancel{B}\, aB_2 | a | bA_3A_2 | bA_3A_2B_2}$

$\cancel{A_3} \rightarrow \boxed{B_2 \longrightarrow A_1A_2A_2 | A_1A_2A_2B_2}$

Sub A_3 in A_2 ~~Sub A_2 in A_1~~

$A_2 \longrightarrow A_3A_1 | b$

* ~~$A_1 \rightarrow aB_2A_1A_3 | bA_3 | aA_3$~~

* $\boxed{A_2 \longrightarrow aB_2A_1 | b | aA_1 | bA_3A_2B_2A_1 | bA_3A_2A_1}$

$\boxed{A_1 \longrightarrow bA_3A_2A_1A_3 | aA_1A_3 | bA_3A_2B_2A_1A_3 | aB_2A_1A_3 | bA_3}$

* $B_2 \rightarrow aB_2A_1A_3A_2A_2 | aB_2A_1A_3A_2A_2B_2 |$

$bA_3A_2A_2 | bA_3A_2A_2B_2 | aA_3A_2A_2 |$

$bA_3A_2A_1A_3A_3A_2 | aA_1A_3A_3A_2 | aA_3A_2A_2B_2 |$

$\cancel{A_1} \rightarrow bA_3A_2B_2A_1A_3A_3A_2 | aB_2A_1A_3A_3A_2 | \ldots$

Pumping lemma for CFG's

Let L is a CFG

Then there is a const n such that if

$z \in L$ & $|z| \geq n$

then we can write $z = uvwxy$;

(i) $|vx| > 0$

(ii) $|vwx| \leq n$.

(iii) $k \geq 0 \Rightarrow uv^k wx^k y \in L$

Eg:-1] $L = \{0^n 1^n 2^n, n \geq 1\}$ is not CFL.

:- Let L = CFG.

$z \in L \Rightarrow 0^i 1^i 2^i$ $\quad |z| = i+i+i$

$= 3i \geq n \Rightarrow \boxed{n = 3i}$

$$z = \underbrace{0^{i-j}}_{u}\underbrace{0^{j}}_{v}\underbrace{1^{i}}_{w}\underbrace{2^{i-j}}_{x}\underbrace{2^{j}}_{y}$$

i) $|vx| > 0 \Rightarrow |\cancel{j} + i - \cancel{j}| = i > 0$

ii) $|vwx| \Rightarrow |j + i + i - j| = (2i) \leq (3i)$

iii) $uv^k wx^k y$.

$= 2$ $\quad 0^{i-j} 0^j 0^j 1^i 2^{i-j} 2^{i-j} 2^j$

$0^{i+j} 1^i 2^{2i-j} \notin L \Rightarrow$ Not a CFG.

Prog-2] Let 'L' be a lang $\{x \in \{0,1\}^*\}$; $|x| \Rightarrow$ perf sq

Sol:- L be a CFG.

$z = 0^{n^2}$ $\quad |z| = n^2 \quad \Rightarrow m = n^2$.

$z = uvwxy$

$|vwx| \leq m$

$|vx| \neq \varepsilon$

$k=2 \Rightarrow \quad |uv^2wx^2y| = |uvwxy| + |vx|$

$= n^2 + |vx|$

$= n^2 + (1 \leq |vx| \leq m)$

$= n^2+1 \leq n^2 + |vx| \leq n^2+m$.

$= \underbrace{(n^2+1)}_{\text{greater than } n^2} \leq |uv^2wx^2y| \underbrace{\leq n^2+m}_{\text{greater than } n^2}$.

$\therefore$ upper & ~~to~~ lower limits are not perf sq.

$\therefore$ L is not CFG.

Prob:-3] Let $L : \{0^n | n \text{ is prime}\}$ is not CFL

Sol:- Assume L is a CFL

$z = 0^n \Rightarrow n = a+b+c+d+e$.

$z = 0^a 0^b 0^c 0^d 0^e$

$|bd| \neq \varepsilon \qquad |bcd| \leq n$

$k=2 \quad 0^a 0^b 0^b 0^c 0^d 0^d 0^e$

$\Rightarrow 0^{(a+b+d)} \notin L$.

$\therefore$ It is not a CFL

MODULE-6

Turing Machine & Universal Lang

* It has 7 tuples:

$$T = (Q, \Sigma, \Gamma, \delta, q_0, B, F) :-$$

- Q → Set of States
- Σ → input sym
- Γ → Tape alpha
- q_0 → Start State
- B → blank sym
- F → Final state

~~δ: Q×Γ~~ $\delta : Q \times \Gamma \times \{L, R\} \leftarrow Q \times \Gamma$

* Turing Machine: ① String validation
  ② computable operation

Prob:-1] $L = \{0^n 1^n, n \geq 1\}$ Design TM.

Sol:-] Methodology:-

Γ = Tape Symbo

$\Gamma = \{0, 1, X, Y, B\}$

...	B	0	0	1	1	B	B	...

X ⟶

(i) 0011 ⇒ XOY1
  X ⟶ Y

(ii) XOY1 ⇒ XXY1
  ⟵

(iii) XXY1 ⇒ XXYY
  ⟶

(iv) XXYY
  ⟵ no sym

(v) XXYY reaches blank & accepting symbol
  ⟶ no sym

ID:- Eg:-

q_7
↓ Tape head

B	1	1	0	1	0	0	0	1	1	0	B

$\delta(q_7, 0) = (p, a, L) \Rightarrow 11010\,q_7 0\,0110 \vdash 1101\,p0a\,01$

$\delta(q_7, 0) = (p, a, R) \Rightarrow 11010\,q_7 0\,0110 \vdash 110100\,ap\,0110$

Eg:- $L = \{0^n 1^n \mid n \geq 1\}$ TM.

δ	0	1	X	Y	B
q_0	(q_1, X, R)	–	–	(q_3, Y, R)	–
q_1	$(q_1, 0, R)$	(q_2, Y, L)	–	(q_1, Y, R)	–
q_2	$(q_2, 0, L)$		(q_0, X, R)	(q_2, Y, L)	–
q_3	–	–	–	(q_3, Y, R)	(q_4, B, H)
q_H	–	–	–	–	–

$\omega = 000111.$

q_0 ↓

B	0	0	0	1	1	1	B	B

$(q_0 000111) \vdash (q_1 X00111) \vdash$

q_0 ↓

B	X	0	0	1	1	1	B

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

$(X q_1 00111) \vdash (X0 q_1 0111) \vdash (X00 q_1 111) \vdash (X0 q_2 0Y11) \vdash$

$\delta(q_1, 0) = (q_1, 0, R)$ | $\delta(q_1, 0) = (q_1, 0, R)$ | $\delta(q_1, 1) = (q_2, Y, L)$ | $\delta(q_2, 0) = (q_2, 0, L)$

$(q_2 X00Y11)$.

~~$(X q_2 00Y11) \vdash (X$~~

~~$\delta(q_2, X) = (q_0, X, R)$~~

$(q_2 X00Y11) \vdash (X q_0 00Y11) \vdash (XX q_1 0Y11) \vdash (XX0 q_1 Y11)$

$(q_2, X) = (q_0, X, R)$ | $(q_0, 0) = (q_1, X, R)$ | $(q_1, 0) = (q_1, 0, R)$ | ~~$(q_1, Y) = (q_1, Y, R)$~~

$(XX0Y q_1 11) \vdash (XX q_2 0YY1) \vdash (XX q_2 0YY1) \vdash (X q_2 X0YY1)$

$(q_2, Y) = (q_2, Y, L)$ | $(q_2, 0) = (q_2, 0, L)$ | $(q_2, X) = (q_0, X, R)$

$(XX q_0 0YY1) \vdash (XXX q_1 YY1) \vdash (XXXY q_1 Y1) \vdash (XXXYY q_1 1)$

$(q_0, 0) = (q_1, X, R)$ | $(q_1, Y) = (q_1, Y, R)$ | $(q_1, Y) = (q_1, Y, R)$ | $(q_1, 1) = (q_2, Y, L)$

$(XXXY q_2 YY) \vdash (XXX q_2 YYY) \vdash (XX q_2 XYYY) \vdash (XXX q_0 YYY)$

$(q_2, Y) = (q_2, Y, L)$ | $(q_2, Y) = (q_2, Y, L)$ | $(q_2, X) = (q_0, X, R)$ | $(q_0, Y) = (q_3, Y, R)$

$\underbrace{(XXXY q_3 YY) \vdash (XXXYY q_3 Y) \vdash (XXXYYY q_3)}_{(q_0, Y) = (q_3, Y, R) \text{ \& } (q_3, Y) = (q_3, Y, R)} \vdash (q_3, B) \Rightarrow (q_4, B, H)$

$\Rightarrow \boxed{XXXYYY q_4} \in$

Prob:-2] $L = \{a^n b^n c^n \mid n \geq 1\}$. Const Turing Machine

B	a	a	b	b	c	c	B

States	a	b	c	X	Y	Z	B
$\rightarrow q_0$	(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_2, b, R)	(q_3, Z, L)			(q_2, Z, R)	
q_3	(q_3, a, L)	(q_3, b, L)		(q_0, X, R)	(q_3, Y, L)	(q_3, Z, L)	
q_4					(q_4, Y, R)	(q_4, Z, R)	$(q_5,$ [illegible]
$* q_5$							

$q_0abbcc \vdash Xq_1bbcc \vdash XYq_2bcc \vdash XYbq_2cc \vdash$

~~$XYbZq_3$~~ $XYq_3bZc \vdash Xq_3YbZc \vdash q_3XYbZc \vdash$

$Xq_0YbZc \vdash XYq_4bZc$ ⇒ not reaching affected state

∴ abbcc is invalid.

w = abc

$q_0abc \vdash Xq_1bc \vdash XYq_2c \vdash Xq_2YZ \vdash$

w = aabbcc

$q_0aabbcc \vdash Xq_1abbcc \vdash Xaq_1bbcc \vdash XaYq_2bcc \vdash$

$XaYbq_2cc \vdash XaYq_3bZc \vdash Xaq_3YbZc \vdash Xq_3aYbZc \vdash$

$q_3XaYbZc \vdash Xq_0aYbZc \vdash XXq_1YbZc \vdash XXYq_1bZc \vdash$

$XXYYq_2Zc \vdash XXYYZq_2c \vdash XXYYq_3ZZ \vdash XXYq_3YZZ \vdash$

$XXq_3YYZZ \vdash Xq_3XYYZZ \vdash XXq_0YYZZ \vdash XXYq_4YZZ \vdash$

$XXYYq_4ZZ \vdash XXYYZq_4Z \vdash XXYYZZq_5$.

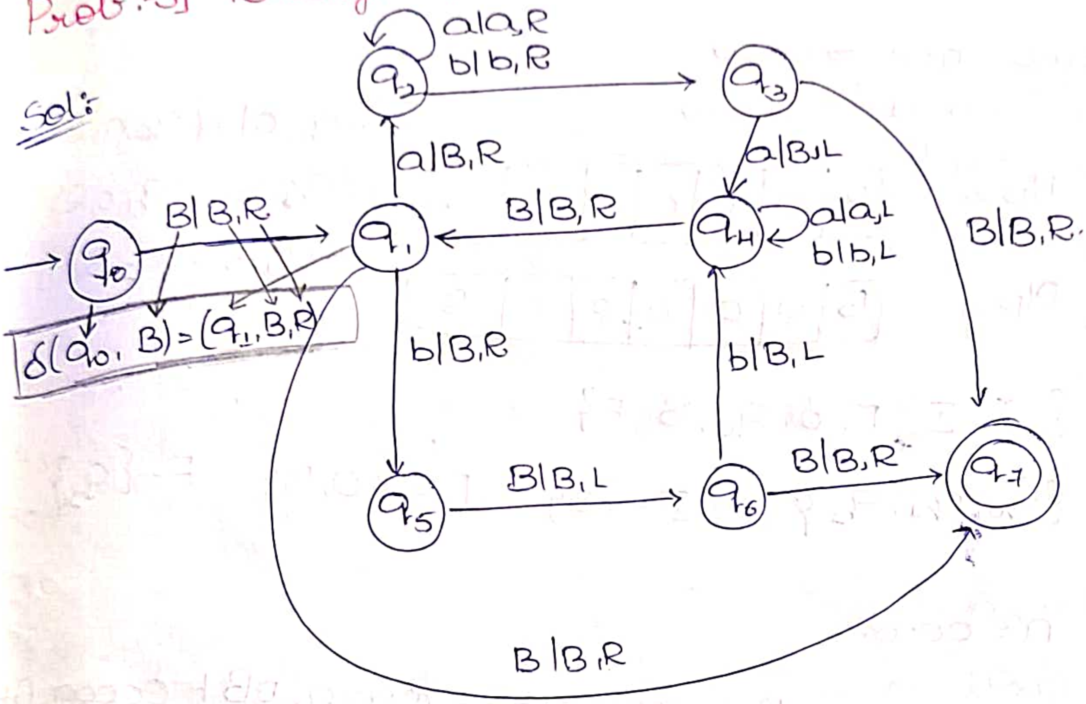
∴ reaches q_5 ⇒ accepted.

$w = aaabbbccc.$

$q_0aaabbbccc \vdash Xq_1aabbbccc \vdash Xaq_1abbbccc$

Prob:-3] Design TM accepts L = palindrome{a,b}

Sol:-



$w = q_0Baba.$

$\Rightarrow Bq_1aba \vdash BBq_2ba \vdash BBbq_3a \vdash \cancel{BBb}\ BBq_4bB \vdash$

$Bq_4BbB \vdash BBq_1bB \vdash BBBq_5B \vdash BBq_6BB \vdash$

$BBBq_7B \Rightarrow$ Reaches Final state.

$w = q_0Baaaa$:-

$Bq_1aaaa \vdash BBq_2aaa \vdash BBaq_2aa \vdash BBaaq_2a \vdash BBaaaq_2$

Computable Languages & Functions:-

(1) Let 'f' be successor function $f(n) = n+1$ for each $n \in N$. Design a TM that computes f. $\boxed{\Sigma = \{0}$

means $n = 4 \Rightarrow 0000$

$n+1 \Rightarrow 00000$

Sol:- i/p ⇒

B	0	0	0	0	B	B

o/p ⇒

B	0	0	0	0	0	B

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

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

$\delta(q_1, B) = (q_2, B, H)$

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

$Q = \{q_0, q_1, q_2\}$ $\Sigma = \{0\}$ $\Gamma = \{0, B\}$ $F = \{q_2\}$.

ID:- $n = 0000$.

$q_0 0000B \vdash 0q_0 000B \vdash 00q_0 00B \vdash 000q_0 0B \vdash 0000q_0 B \vdash$

$00000q_1 B \vdash 00000\underline{q_2} B \Rightarrow$ Reaches final state.

Prob:-2] Let 'f' be function $f(n) = n+3$. Design TM for f. $\Sigma = \{0\}$

$n = 000$

$f(n) = 000000$.

Sol:- i/p ⇒

B	B	B	0	0	0	B

o/p ⇒

0	0	0	0	0	0	B.

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

$\delta(q_a, B) = (q_1, 0, L)$

$\delta(q_1, B) = (q_2, 0, L)$

$\delta(q_2, B) = (q_3, 0, H)$

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

$Q = \{q_0, q_a, q_2, q_3\}$ $\Sigma = \{0\}$ $\Gamma = \{0, B\}$ $F = \{q_3\}$.

Prob:-3] Design TM for add funct $f(m,n) = m+n$ is compoutable.

$\overset{m}{0^3}\,\overset{n}{0^4} \Rightarrow \overset{m+n}{0^7}$ $\underbrace{0000}_{m}\,1\,\underbrace{000}_{n} \Rightarrow 0000000.$

Sol:- i/p

B	0	0	0	0	1	0	0	B

$\uparrow q_0$

o/p

B	B	0	0	0	0	0	0	0	B

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

$w = q_0B\,00\,1\,000\,B. \vdash$

$Bq_0001000B \vdash BBq_1\cancel{0}1000B \vdash BB0q_1\,1000B \vdash BB00q_2000B \vdash BB000q_200B \vdash BB0000q_2\,0B \vdash$
$BB00000q_2B \vdash BB00000q_3B \Rightarrow$ Reaches final State

Type:-2]

B	0	0	0	0	1	0	0	0	B

B	0	0	0	0	0	0	0	B	B

$\delta(q_0, B) = (q_0, B, R)$
$\delta(q_0, 0) = (q_0, 0, R)$
$\delta(q_0, 1) = (q_1, 0, R)$
$\delta(q_1, 0) = (q_1, 0, R)$
$\delta(q_1, B) = (q_2, B, L)$
$\delta(q_2, 0) = (q_3, B, H) \rightarrow$ Final

$w = q_0B\underbrace{000}_{3}\,1\,\underbrace{0}_{1}B \vdash^* Bq_00001\,0B \vdash$
$B0q_000\,10B \vdash B00q_00\,10B \vdash$
$B000q_0\,10B \vdash B0000q_1\,0B \vdash$
$B00000q_1B \vdash B0000q_2\,0B \vdash$
$\underbrace{B0000}_{4}\overset{B}{\cancel{0}}q_3B.$

Prob-4] Design Turning machine $f(m,n) = m-n.$

Sol:- Two cases $m>n$ $m<n$

$m \dot{-} n\ [m>n]$ proper sub

	0	1	B
$\rightarrow q_0$	(q_1, B, R)	(q_5, B, R)	-
q_1	$(q_1, 0, R)$	$(q_2, 1, R)$	-
q_2	$(q_3, 1, L)$	$(q_2, 1, R)$	(q_4, B, L)
q_3	$(q_3, 0, L)$	$(q_3, 1, L)$	(q_0, B, R)
q_4	$(q_4, 0, L)$	(q_4, B, L)	$(q_0, 0, R)$
q_5	(q_5, B, R)	(q_5, B, R)	(q_6, B, R)

$\underline{\underline{w = 010}}$ [m=n]

$q_0010 \vdash Bq_110 \vdash B1q_20 \vdash q_311 \vdash q_3B11 \vdash q_011 \vdash Bq_51$

$BBq_5B \vdash BBB\boxed{q_6} \dashv$ Final State -

$\underline{w = 0100}$ [m<n]

$q_00100 \vdash Bq_1100 \vdash B1q_200 \vdash Bq_3110 \vdash q_3B110 \vdash Bq_0110 \vdash$

$BBq_510 \vdash BBBq_50 \vdash BBBBB\boxed{q_6} \dashv$ Final State.

$w = 0010$ [m>n]

$q_00010 \vdash Bq_1010 \vdash B0q_110 \vdash B01q_20 \vdash B0q_311 \vdash Bq_3011 \vdash$

$q_3B011 \vdash Bq_0011 \vdash BBq_111 \vdash BB1q_21 \vdash BB11q_2B \vdash BB1q_41B \vdash$

$BBq_41BB \vdash Bq_4BBBB \vdash B0q_{r_0}BBB$

Turing Machine with Subroutines:-

Prob:- 1] Multiplication using subroutines.

Methodology :- $3 \times 4 = 12$

$3+3+3+3 = 12$

$4+4+4 = 12$

[Inside multiplication Subroutine there are multiple Addition]

$\overbrace{111}^{a}\,0\,\overbrace{1111}^{b}\,0\,\overbrace{111111111111}^{o/p}$

Sol:-

a				b																	
1	1	1	0	1	1	1	1	0	B	B	B	B	B	B	B	B	B	B	B	B	B

B   4 4 4 4 → 1 1 1 1

Come back (1 1 1 1)

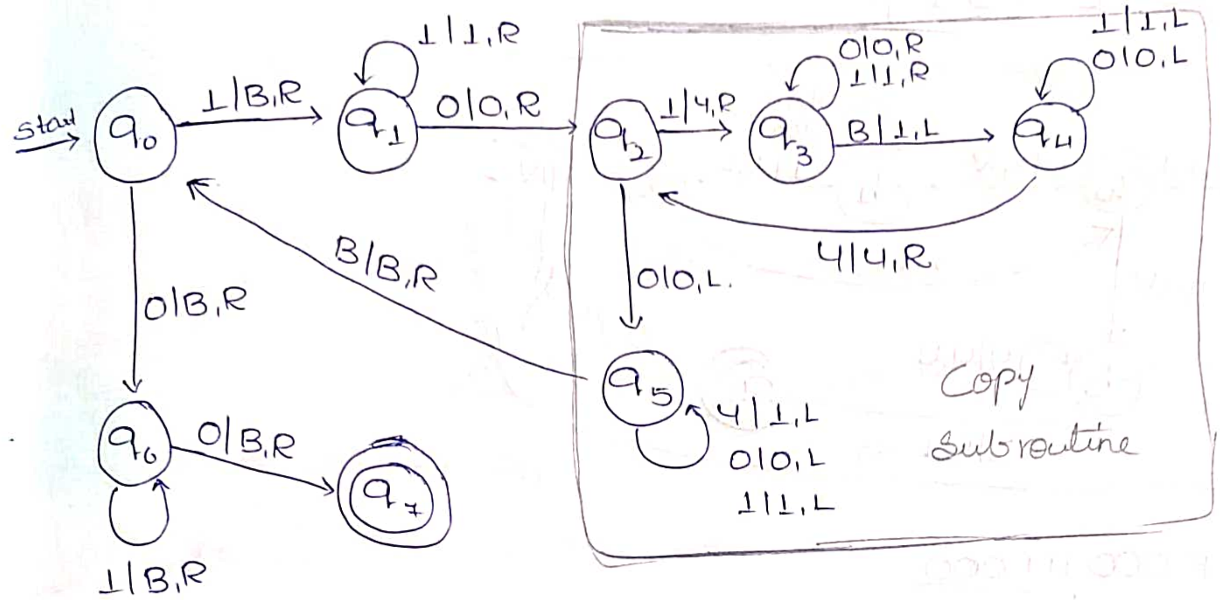
B B   4 4 4 4   1 1 1 1 1 1 1 1

Come back (1 1 1 1)

B B B   4 4 4 4   1 1 1 1 1 1 1 1 1 1 1 1

2x4

1	1	0	1	1	1	1	0	B	B	B	B	B	B	B	B	



Turing machine Programming Techniques

Inserting centre
Eg:- Palindrome wcw^R
multi 0010010000
Sub also

Prob:-1] How to find left end of Tape sym of TM?

Sol:- For eg: $\Sigma = (a, b)$

left b/c Σ has a.

$\Gamma \Rightarrow$

X	4	Z	B	B	a	b	a	b	b	b	-	-	-	

$\Rightarrow$

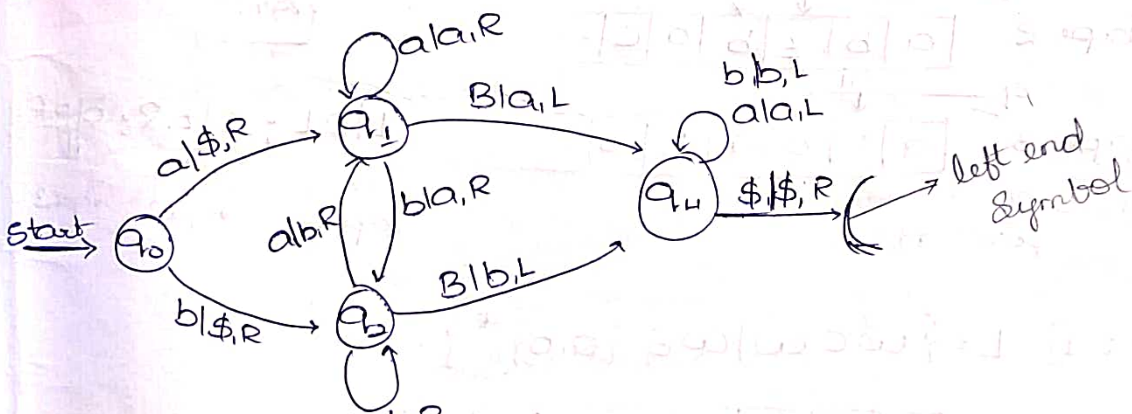
\$	a	b	a	b	b	-	-

Eg:-

B	B	b	a	a	b	a	B	B

$\Rightarrow$

\$	b	a	a	b	a	B

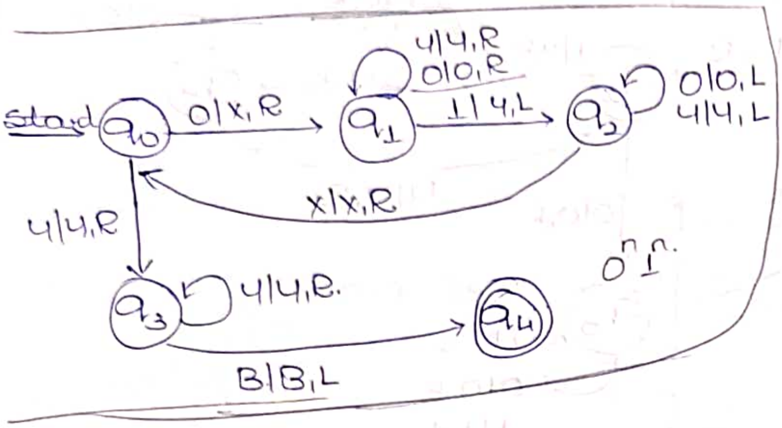


W: baaba

ID: $q_0$baaba ⊢ \$$q_2$aaba ⊢ \$b$q_1$aba ⊢ \$ba$q_1$ba ⊢ \$baaq_2a ⊢ \$baabq_1B ⊢ \$baabq_4a ⊢ \$baaba ⊢ q_4\$baaba

Prob:-2] Design TM $L=\{0^n1^n0^n, n\geq 1\}$ $\Sigma=\{0,1\}$

Sol:- $L=\{0^n1^n\}$.



i/P 000 111 000

first o/P x x x $\boxed{444\ 000}$

AAA CCC

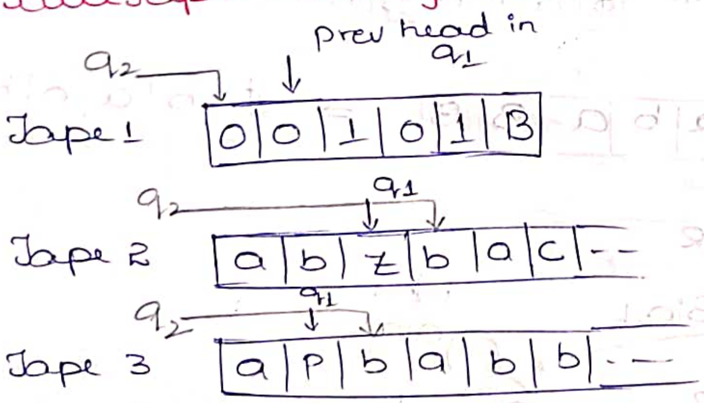
Prob:-3] $L=\{wcw \mid w\in(a,b)\}$

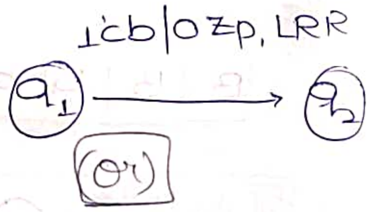
$w=\{abcab,\ abacaba,\ c, \text{---}\}$

Sol:- we have to verify that the string is same or not we have to construct TM that accepts 2 similar strings.

MultiTape Turing Machine:-

Eg:-





1|0,L ; c|z,R ; b|p,R

⇓ Tape1 ⇓ Tape2 ⇓ Tape3

(K=3) ⇒ no. of Tapes

Prob:-1] $L=\{wcw \mid w\in\{a,b\}^*\}$.

Tape 1 (T_1)

b	b	a	a	C	b	b	a	a	B	..

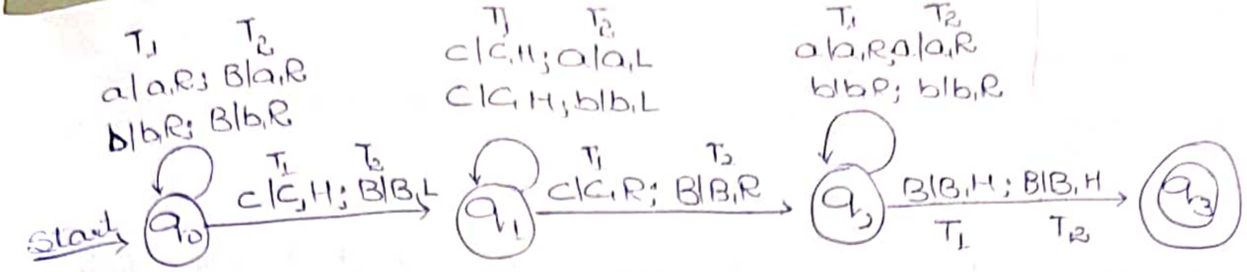
↑ $q_3 \in F$

Tape 2 (T_2)

B	b	b	a	a	B	B	B	B	-	-

↑

$q_3 \in F$



Here for valid string TM reaches $q_3 \in F$. else it will stop in middle.

Undecidality of Turning Machine

* Undecidability of Halting Prob :- (HP)

→ H(P, I) (TM, Prog, i/p)

- Halt
- Not Halt

→ C(X)

if (H(X,X) == Halt)) } Means if it is Halt it goes to infinite loop & it will not Halt/terminate
loop forever;

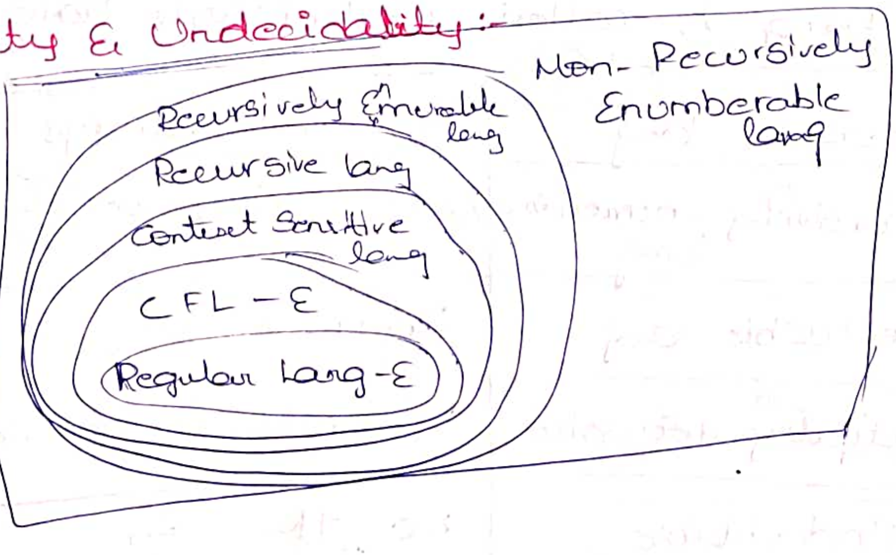
else
Return; } not Halt means it will terminate

→ C(C)

- H(C,C) == Halt → Not Halt
- H(C,C) == Not Halt → Halt

} we can prove this by contradiction method.

Decidability & Undecidability :-



Recursive Language:-

A Lang 'L' is said to be Recursive Lang if there exists a TM such that it will accept all strings of 'L' & reject all string not belong to 'L'.

→ It will halt everytime & give an answer (accepted/not) for each & every string I/p.

Recursively Enumerable Lang:-

A Lang 'L' is said to be recursively Enum lang if there exists a TM which will accept all the Lang & halt & may or may not reject non-accepted strings.

Decidable Language:- A lang 'L' is ~~said~~ deciable if a recursive language. Means all decidable are recursive lang.

Partially Decidable Lang:- It is vice-versa of Recursively Enumerable lang.

Undecidable Language:-

The lang which is not decidable & it maybe partially decidable language.

* If a Lang does not have TM then that Lang is called Undecidable Lang.

Recursive lang	TM will always Halt
Recursively Enumerable lang.	may / may not Halt
Decidable lang	Recursive lang
Partially decidable	Recursively Enumerable lang
Undecidable	no TM for that lang