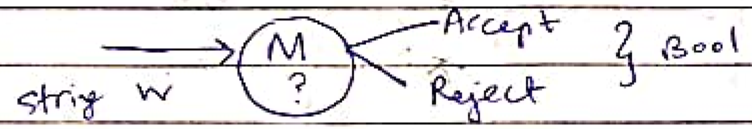
ToC : Theory of Computation

DFA: Deterministic Finite Automata

L : Language Σ : Symbol. (input alphabet)
↳ non empty $\Sigma \neq \emptyset \neq \{\}$.

$L = \{w_1, w_2, \ldots\}$



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

$L_1 = \{w \mid w \in \{a,b\}^*$ & w ends with $ab\}$

$= \{ab, aab, bab, aaab, baab, abab, bbab, \ldots\}$

$L_2 = \{w \mid w \in \{ab\}^*$ & w is a palindrome$\}$.

- Regular Languages - Recognisable by DFA.

Beginning state only one.
Final state - accepting state : one or more.

<u>DFA</u>

$M = (Q, \Sigma, \delta, q_0, F)$ 5 attributes.

- Q : finite set of states.
- Σ : i/p alphabet
- δ : mapping function
- $q_0 \in Q$ initial state
- $F \subseteq Q$ set of accepting states.

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

$\delta(q, a) = P$

present state : q next state : P.
present input symbol : a

classmate
Date 28/10/2024
Page 02

Q1. $\Sigma = \{a, b\}$
$Q = \{q_0, q_1, q_2\}$ $\}$ $2 \times 3 = 6$

$\delta(q_0, a) = q_1$
$\delta(q_0, b) = q_0$
$\delta(q_1, a) = q_1$
$\delta(q_1, b) = q_2$
$\delta(q_2, a) = q_1$
$\delta(q_2, b) = q_0$

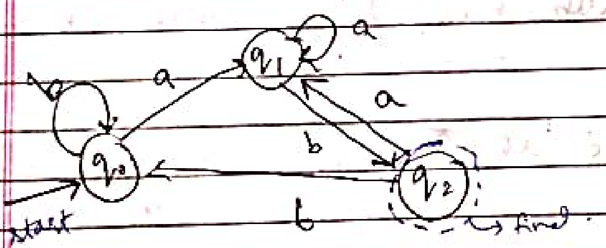
columns are symbols

rows are states

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

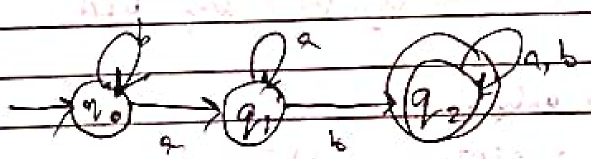
transition table.

transition diagram.



- start at initial state,
- process entire string
- end at final state.

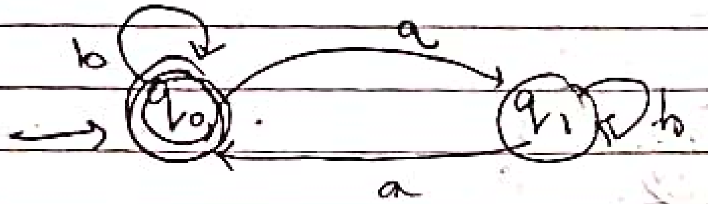
Q2.



↳ ab in substring.
$L = \{ w \mid w \in \{a, b\}^* \text{ \& } w \text{ has substring } ab \}$

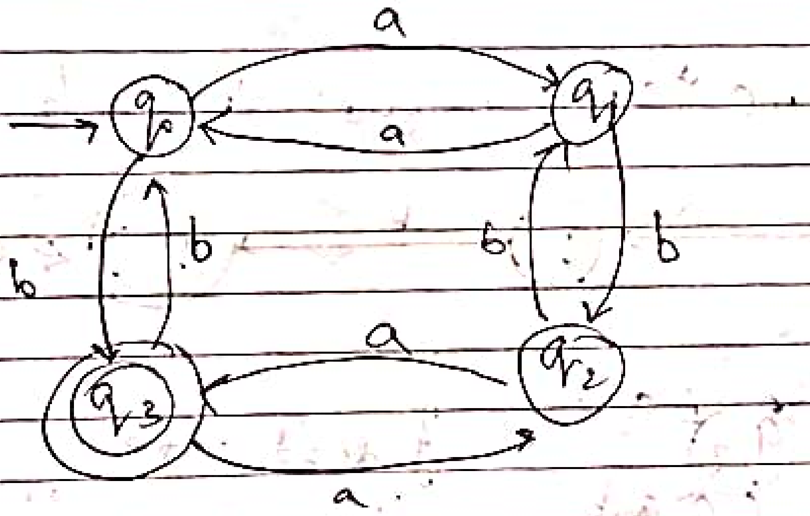
If M, a DFA is given,
$L(M) = \{ W \mid W \in \Sigma^* \text{ \& } \delta^*(q_0, w) \in F \}$
(definition)

L : string should have even number of a's.

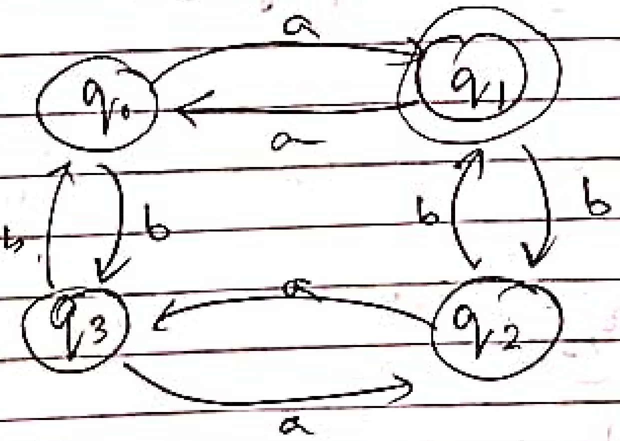


null string ϵ is accepted. (0 a's - even).
so starting & final state same

L : str should have even a's, odd b's.

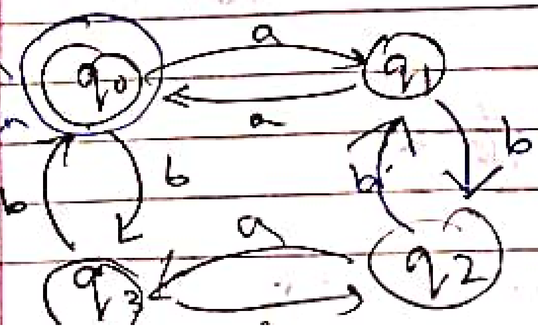


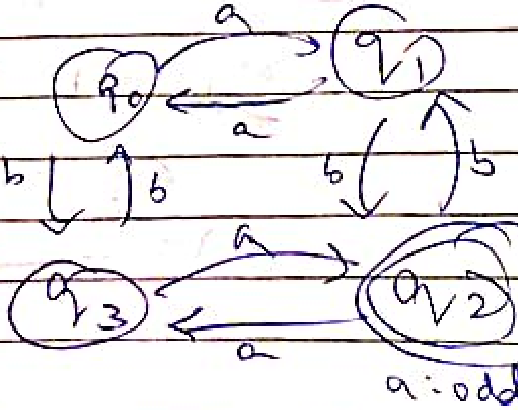
a : even
b : odd.



a : odd
b : even.

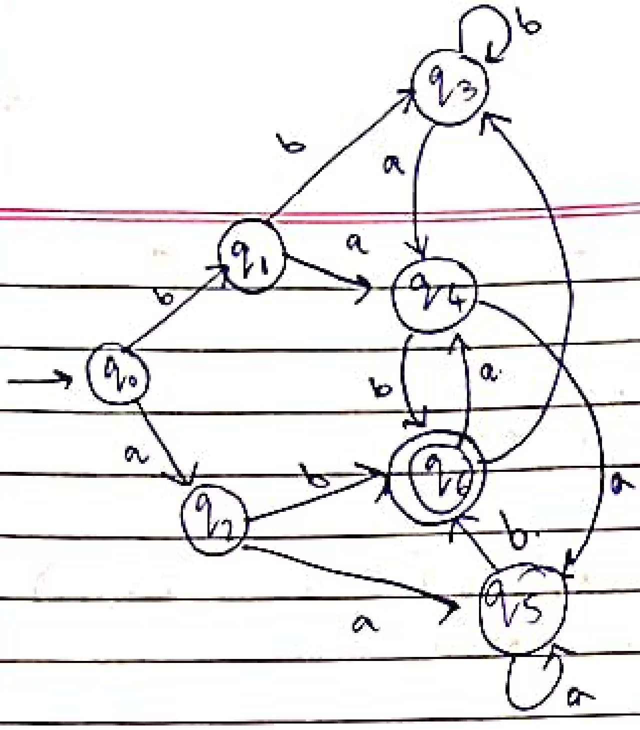
a : even
b : even





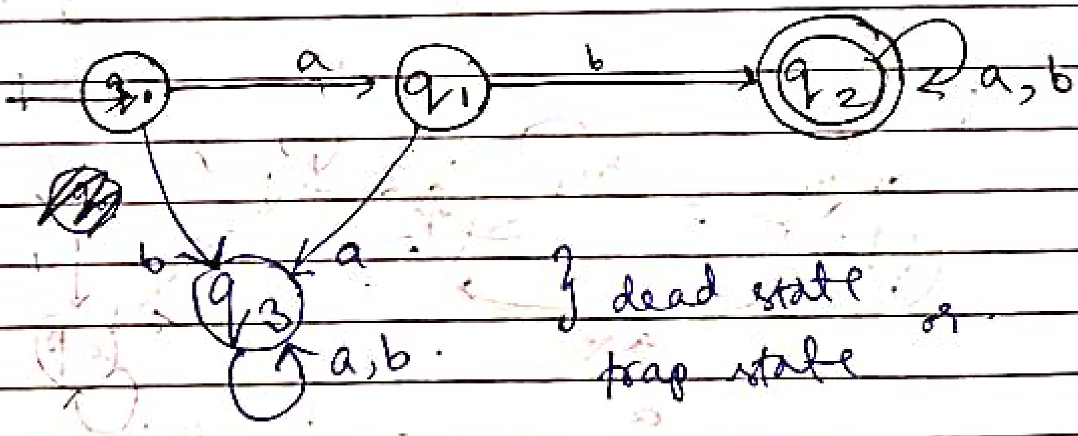
a : odd b : odd

Q)



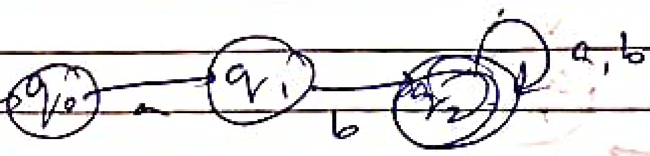
$L = \{ w \mid w \in \{a,b\}^* \text{ \& } w \text{ ends with } ab \}$

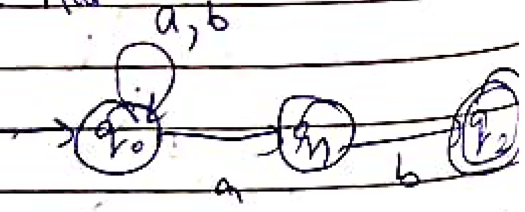
Q) $L = \{ w \mid w \in \{a,b\}$ begins with $ab \}$



} dead state or
trap state

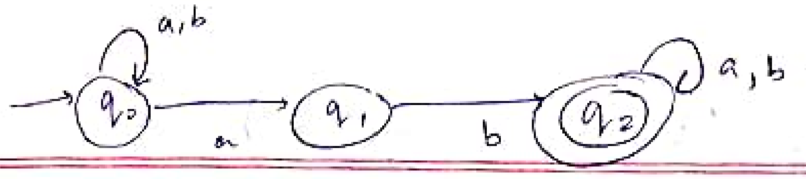
NFA Non Deterministic Finite Automata





~~ends~~ begins with ab . ends with ab .

substring is ab .



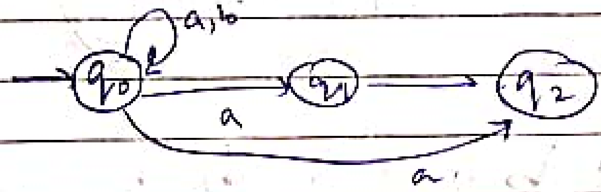
$\delta(q_0, a) = \{q_0, q_1\}$
$\delta(q_0, b) = \{q_0\}$
$\delta(q_1, a) = \phi$
$\delta(q_1, b) = \{q_2\}$
$\delta(q_2, a) = \{q_2\}$
$\delta(q_2, b) = \{q_2\}$

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

$M = (Q, \Sigma, \delta, q_0, F\}$
↳ different in DFA, NFA.

$\delta : Q \times \Sigma \rightarrow Q \quad \rightarrow$ (DFA)
$\delta : Q \times \Sigma \rightarrow 2^Q \quad \rightarrow$ (NFA)
(each state : yes or no).

DF



so each state can use same symbol to go to 0, 1, 2, ..., all states.
(so each state has 2 options like that)

DFA : $\delta^*(q_0, aY)$
$\delta^*(\delta(q_0, a), Y)$.

NFA ↳ $\delta^*(\{p, q, r\}, Y\}$.
(do it for all 3 then).

$= \cancel{\delta^*(p,q)}\ \delta^*((\delta\{p, q, r\}, b), X)$ $\quad Y = bX$.

$\delta(p, b) \cup \delta(q, b) \cup \delta(r, b)$

continue until all symbols are processed.
finally results will be sets.
if a set contains any one final state : ACCEPT.

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

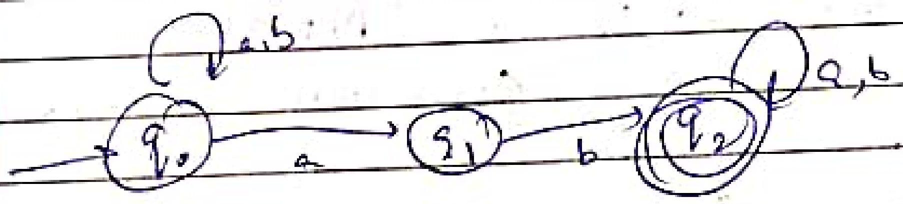
If M (DFA) is given:

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

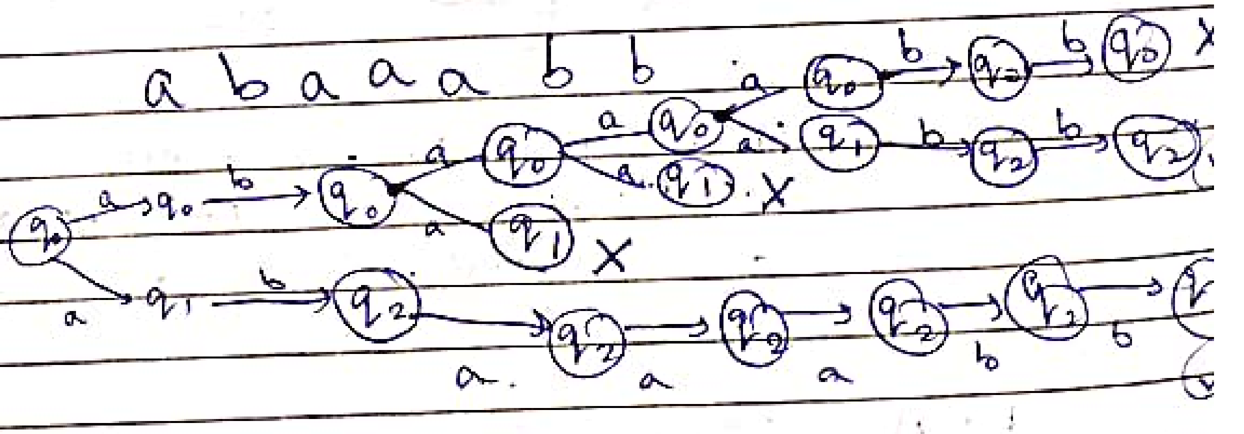
NFA:

$L(M) = \{ W \mid W \in \Sigma^* \ \& \ \underbrace{\delta^*(q_0, W)}_{\text{set of states}} \cap \underbrace{F}_{\text{final state}} \neq \phi \}$

in the result, we should have at least one of the final states.



a b a a a b b



11/11/2024

Epsilon NFA.

$\rightarrow (p) \xrightarrow{\epsilon} (q)$

ϵ (null string) – NFA:

$\epsilon a = a$

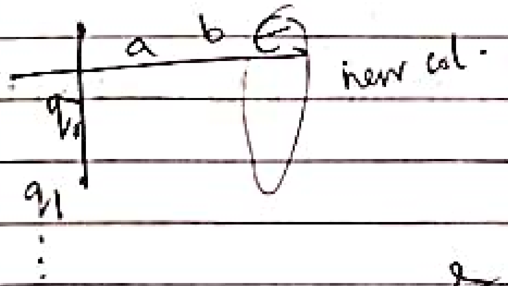
$\rightarrow M = (Q, \Sigma, \delta, q_0, F)$

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

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

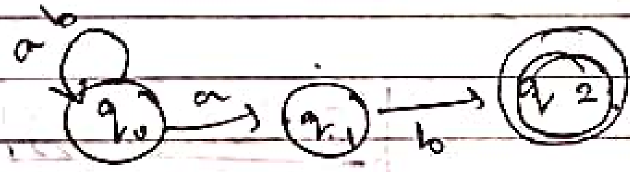
Σ : a, b, ϵ

ENFA $\delta : Q \times \Sigma \cup \{\epsilon\} \rightarrow 2^{Q}$

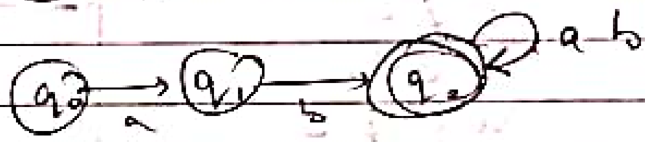


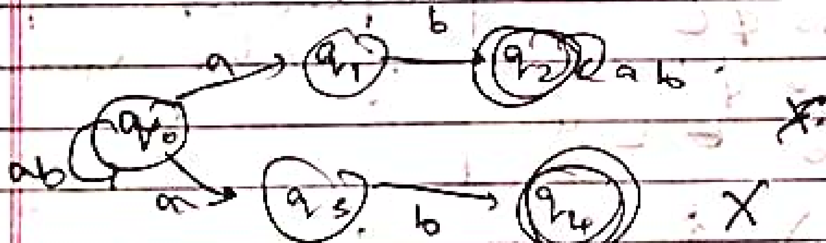
Q) Begins with ab ~~or~~ & ends with ab.

L = {w | w ∈ {a,b}* w - starts & ends with ab.



aab abb

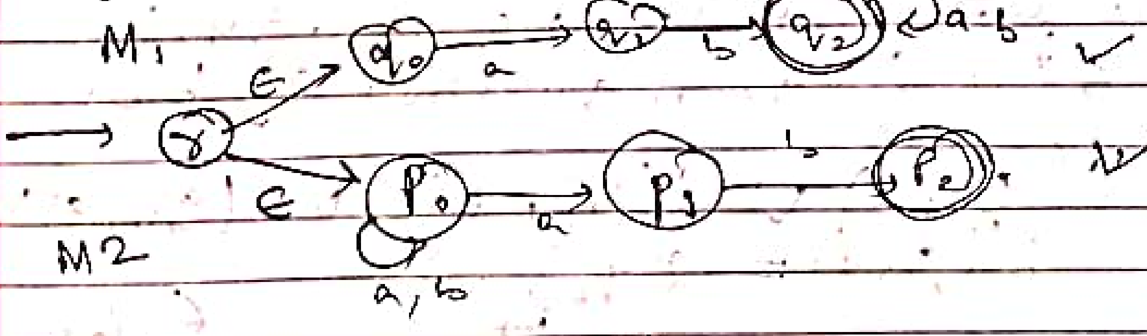




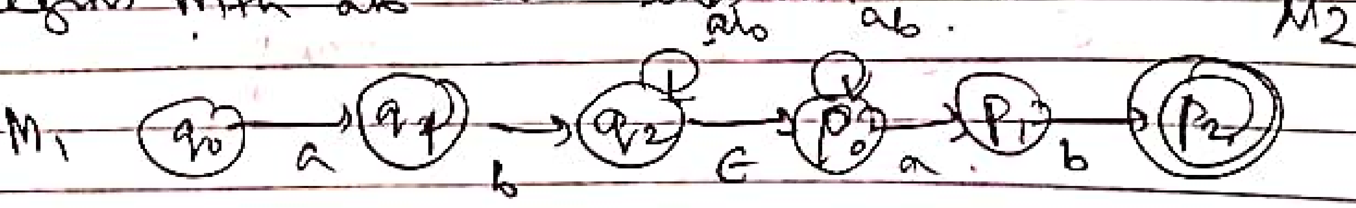
gives correct answer but it is correct for wrong answer also.

bb a b a a

→ Begins with ab or ends with ab.



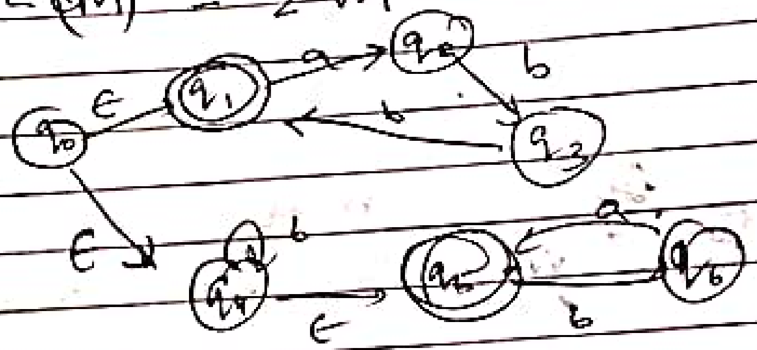
→ ~~Begins with ab~~ and ends with ab.



Date 11/11/2024
Page 08

Language accepted by NFA

$L(M) = \{ w \mid w \in \Sigma^* \ \& \ \delta^*(q_0, w) \cap F \neq \varphi \}$



$L(M) = \{ \epsilon$ ✓
a ∉ L
b ∈ L
aa ∉ L
ab ∉ L
b,a ∉ L
bb ∈ L $\}$.

∈ closure(q)
set of all states which are reachable upon consumption of only epsilon

∈ closure $(q_0) = \{q_0, q_1, q_4, q_5\}$
∈ closure $(q_1) = \{q_1\}$
∈ closure $(q_4) = \{q_4, q_5\}$

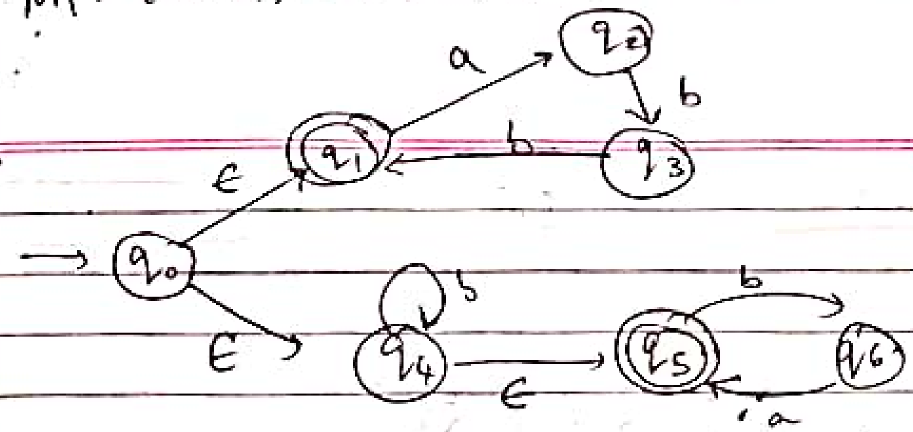
$\delta^*(q \in \lambda^{\epsilon} b^{\epsilon} b^{\epsilon})$ → do this
∈ closure $(q_0) = \{q_0, q_1, q_4, q_5\}$.
$\delta^*(\delta\{q_0\ q_1\ q_4\ q_5\}, a)\ bb)$

$\delta(q_0, a) \cup \delta(q_1, a) \cup \delta(q_4, a) \cup$
$\delta(q_5, a) = \{q_2\}$
↳ Compute ∈ closure

∈ closure (p, q) = ∈ closure(p) ∪ ∈ closure(q)

$M_1 = (Q_1, \Sigma_1, \delta_1, q_1, F_1)$ be given ϵ NFA.

$\boxed{M1}$



ϵ closure $(q_0) = \{q_0, q_1, q_4, q_5\}$.

$\delta_1(q_0, a) = \delta_1(q_0, a) \cup \delta_1(q_1, a) \cup \delta_1(q_4, a) \cup \delta_1(q_5, a)$.

$= \phi \cup \cancel{\phi \cup} \{q_2\} \cup \phi \cup \phi$

$= \epsilon$ closure $(\{q_2\}) = \{q_2\}$.

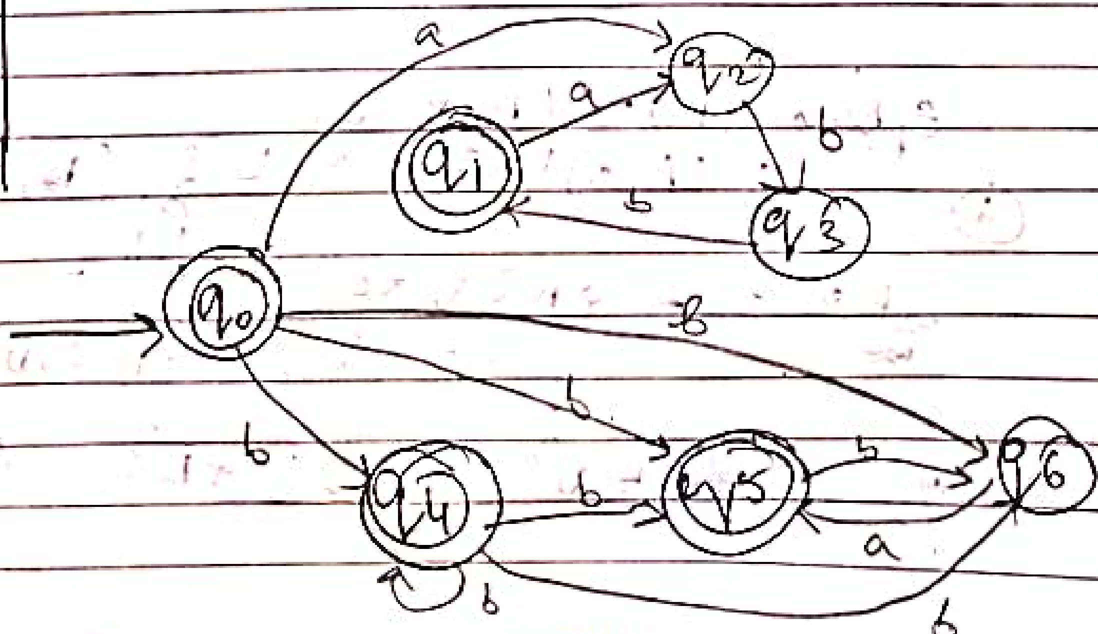
similarly, $\delta_1(q_0, b) = \delta_1(q_0, b) \cup \delta_1(q_1, b) \cup \delta_1(q_4, b) \cup \delta_1(q_5, b)$

$= \phi \cup \phi \cup \{q_4\} \cup \{q_6\}$

$= \epsilon$ closure $(\{q_4, q_6\}) = \{q_4, q_5, q_6\}$

$\Rightarrow$

	a	b
q_0	$\{q_2\}$	$\{q_4, q_5, q_6\}$.
q_1		
⋮		
q_6		



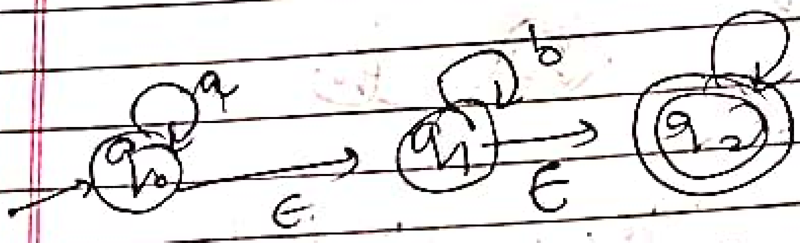
$\boxed{M2}$

classmate Date 11/11/2024 Page 10

Q) $\Sigma_1 = \{a,b,c\}$

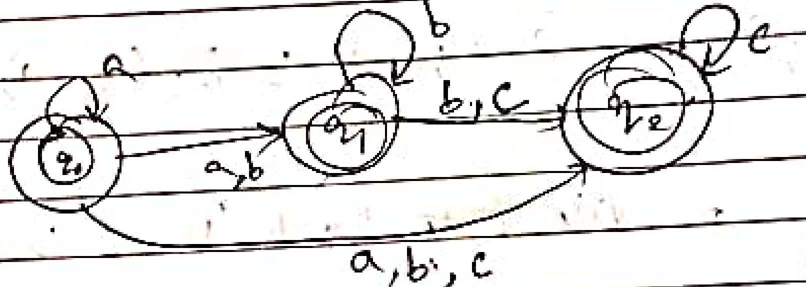
aa ------ bbb ------ cccc

only one state X
$\rightarrow$ q_0 a,b,c
violates the order a,b,c



$\therefore$ 3 states

$\Updownarrow$

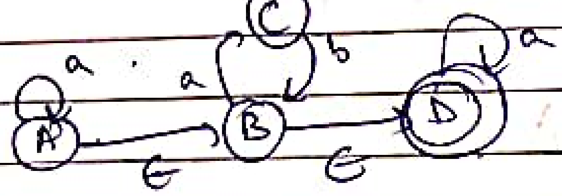


all 3 are final states as each state's ϵ closure contains final state q_2

1. consider each state - each symbol transition
2. for q_0, a check $\delta(q_0,a) \cup \delta(q_1,a) \cup \delta(q_2,a)$ then take epsilon closure of result and mark the transitions

ex: aab ✓
abcabc X

13/4/2024



ϵ closure (A) = {A, B, D}
(B) = {B, D}
(C) = {C}
(D) = {D}

Epsilon(A) = (A, B, D)

1. $\delta_1(A,a) \cup \delta_1(B,a) \cup \delta_1(D,a)$

$\delta_2(A,a)$ = ? $= \{A\} \cup \{C\} \cup \{D\} = \{A, C, D\}$

ϵ closure {A, C, D}

2. {A, B, D} ∪ {C} ∪ {D} = {A, B, C, D}

Equivalent NFA .

$M_1 = (Q_1, \Sigma_1, \delta_1, q_1, F_1)$ → given NFA

$M_2 = (Q_2, \Sigma_2, \delta_2, q_2, F_2)$ $Q_1 = Q_2$ $\Sigma_1 = \Sigma_2$ $q_1 = q_2$

Find final states :

∈ closure of each state. in that if there is a final state, then that also becomes final state.

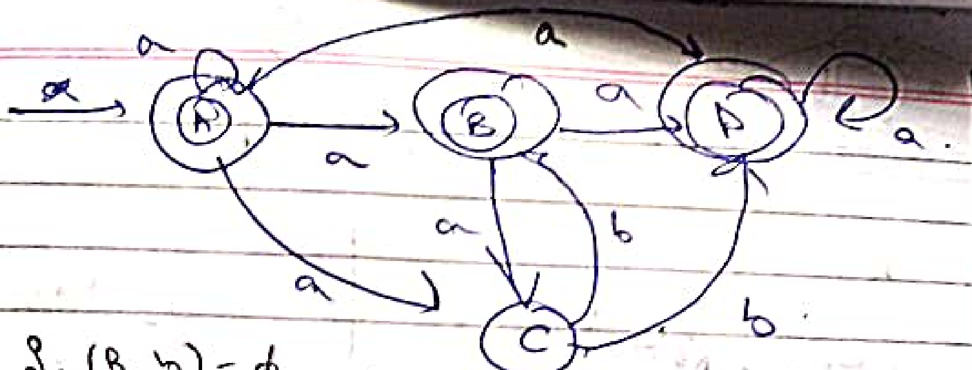
$\epsilon closure(q_0) = \{q_0, q_1, q_4, q_5\}$

q_1 & q_5 are final states here

∴ q_0 is also final.

ϵ closure$(q_4) = \{q_4, q_5\}$ → final state

∴ q_4 is also final.



$\delta_2(A,b) = \delta_1(A,b) \cup \delta_1(B,b) \cup \delta_1(D,b)$
$= \phi \cup \phi \cup \phi$

$\delta_2(B,a) = \delta_1(B,a) \cup \delta_1(D,a)$
$= \{C\} \cup \{D\}$

ϵ closure $\{C,D\}$
$\Rightarrow \{C,D\}$

$\delta_2(B,b) = \phi$

$\delta_2(C,a) = \phi$

$\delta_2(C,b) = B$ ϵ closure of $B = \{B,D\}$

$\delta_2(D,a) = D \rightarrow \epsilon$ closure of $\{D\} = D$.

$\delta_2(D,b) = \phi$.

NFA to DFA

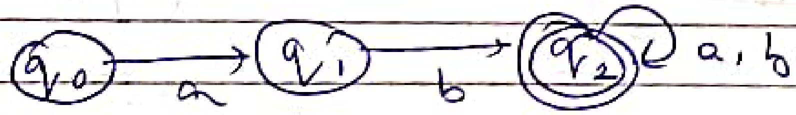
$M_1 = (Q_1, \Sigma, \delta_1, q_i, F_1)$

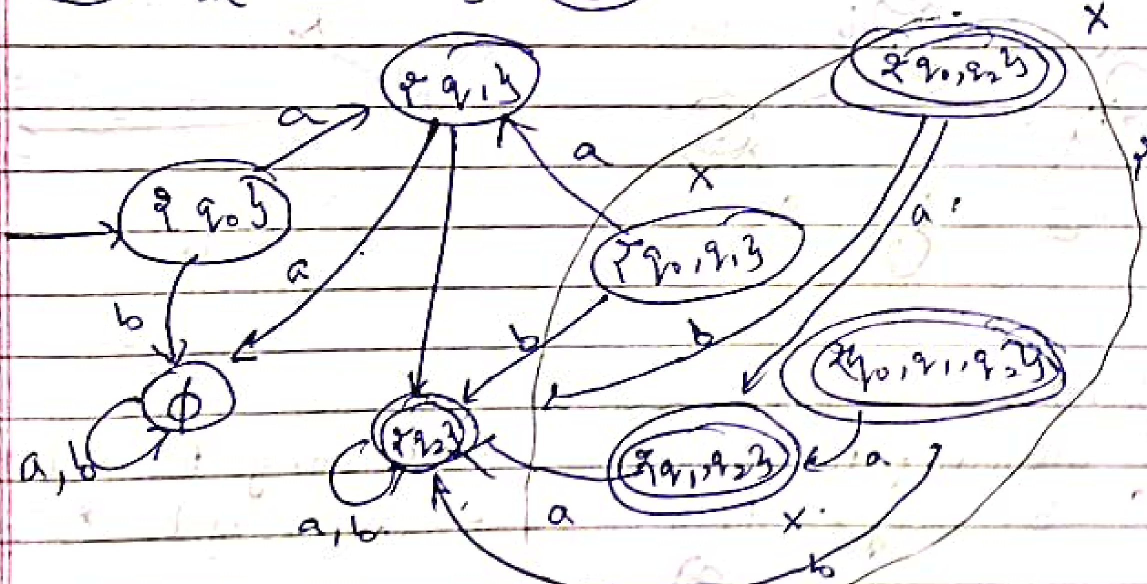
$M_2 = (Q_2, \Sigma, \delta_2, q_2, F_2)$.

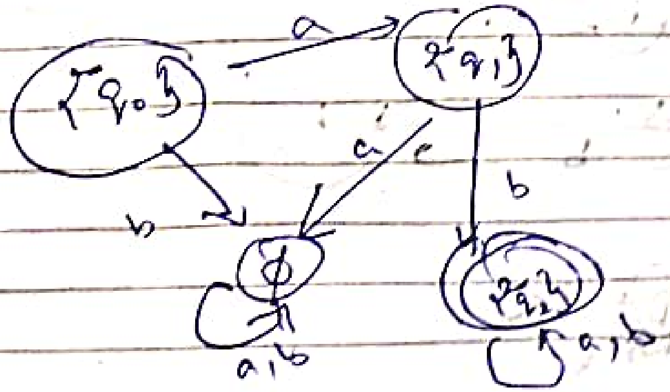
$Q_2 = 2^{Q_1}$

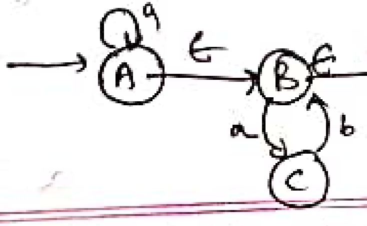
$F_1 = \{q_1\}$

$F_2 = \{p, q, r\} \cap F_1 \neq \phi$

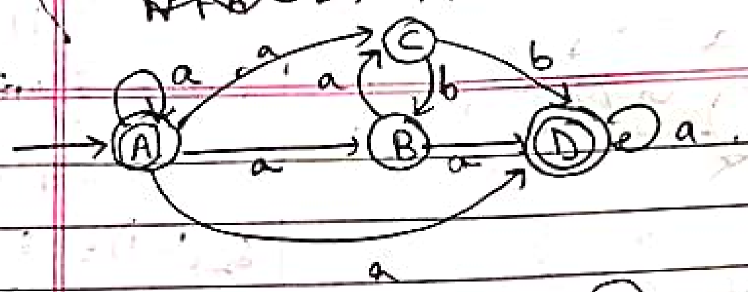


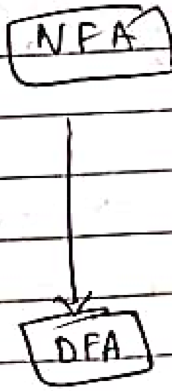


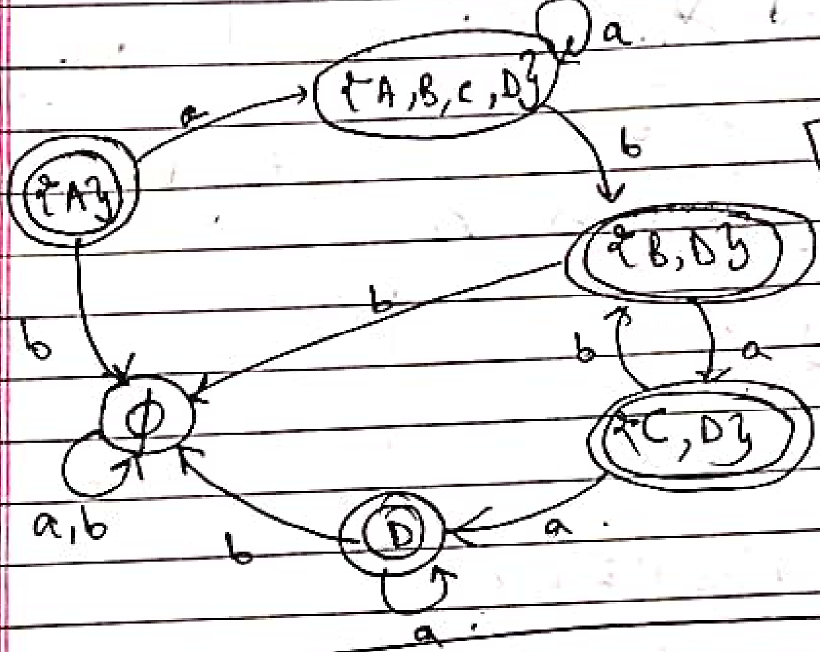




Q1) Ex:- NFA → DFA.







$\delta_2(A,a) = \{A,B,C,D\}$

$\delta_2(A,b) = \phi$

$\delta_2(B,a) = \{C,D\}$

$\delta_2(B,b) = \phi$

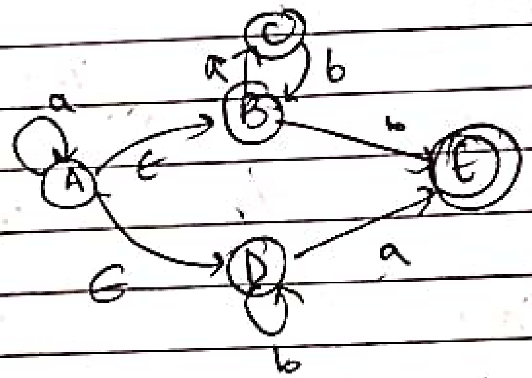
$\delta_2(C,a) = \phi$

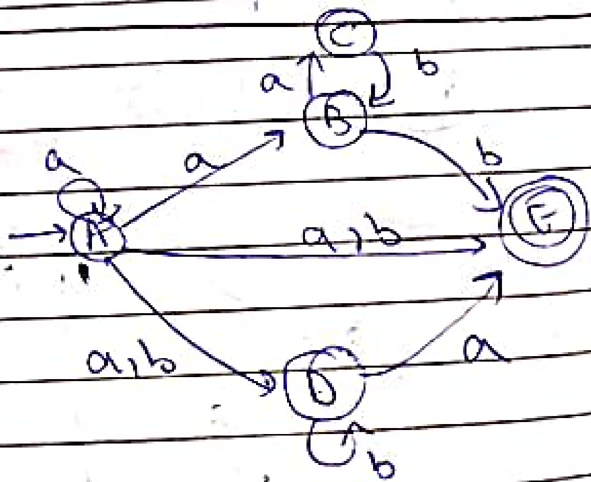
$\delta_2(C,b) = \{B,D\}$

$\delta_2(D,a) = D$

$\delta_2(D,b) = \phi$

Q2)





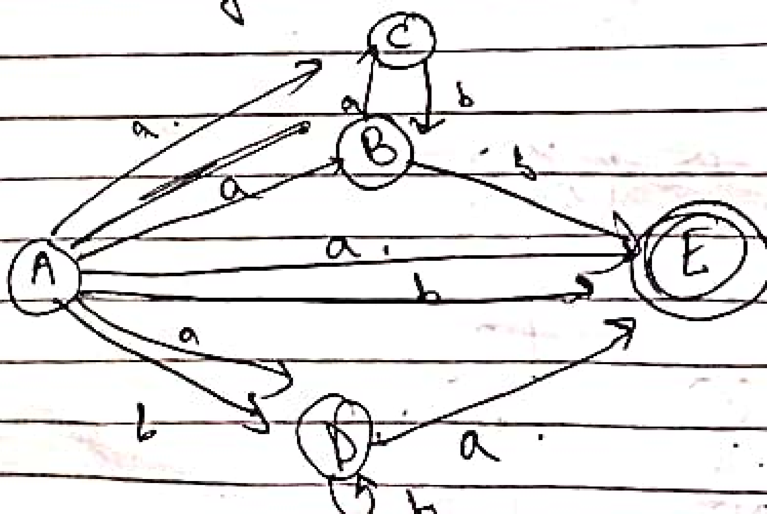
ϵ closure of $A = \{A, B, \cancel{C} D\}$.

ϵ closure of $B = \{B\}$

ϵ closure of $C = \{C\}$

ϵ closure of $D = \{D\}$

ϵ closure of $E = \{E\}$.



$\therefore \delta_2(A, a) = \delta_1(A, a) \cup \delta_1(B, a) \cup \delta_1(D, \phi)$
$= A \cup C \cup E$

ϵ closure of $(A, C, E) = \{A, B, C, D, E\}$

$\delta_2(A, b) = \delta_1(A, b) \cup \delta_1(B, b) \cup \delta_1(D, b) = \phi \cup E \cup D$

ϵ closure of $(E, D) = \{E, D\}$

$\delta_2(B, a) = \delta_1(B, a) = C$

ϵ closure of $C = \{C\}$

$\delta_2(B, b) = \{E\}$ $\quad$ ϵ closure of $\{E\} = E$

$\delta_2(C, a) = \phi$

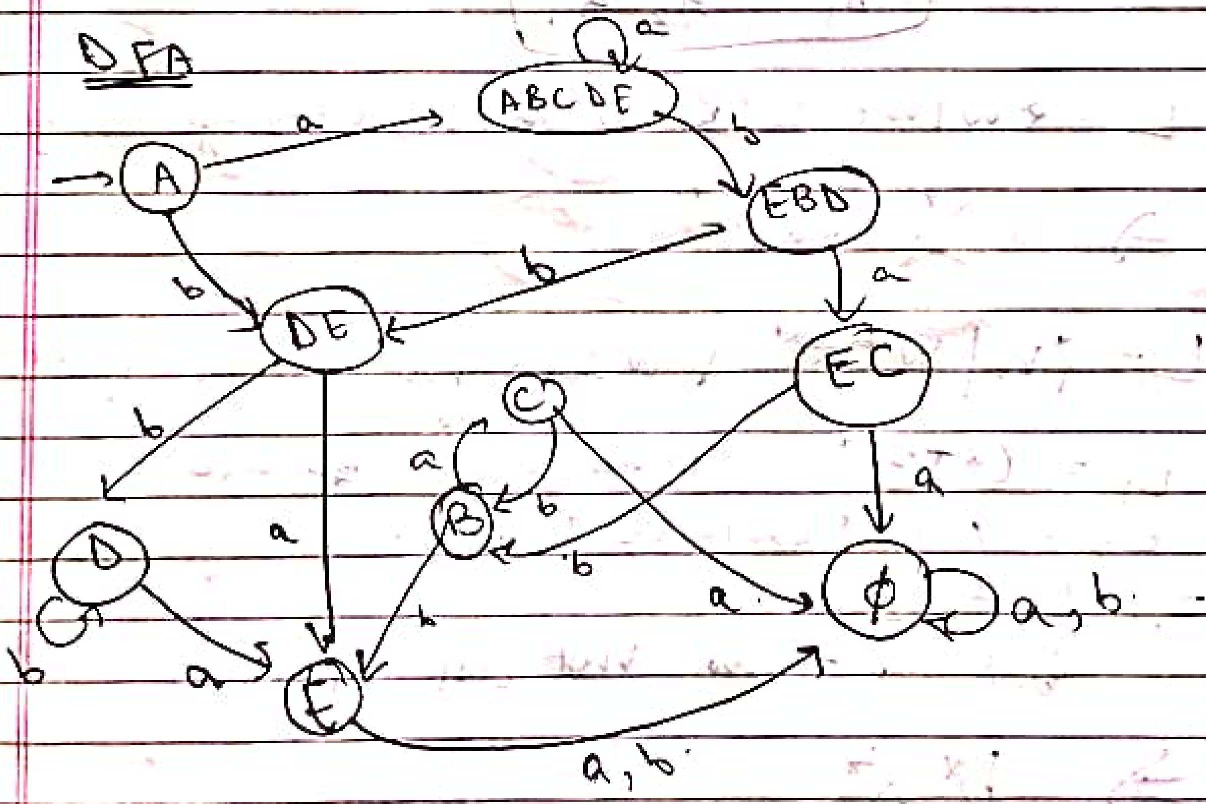
$\delta_2(C, b) = B \rightarrow B$

$\delta_2(D, a) = E \rightarrow E$

$\delta_2(D, b) = D \rightarrow D$

~~$\delta_2(E, a)$~~ $\delta_2(E, a) = \phi$ $\quad$ $\delta_2(E, b) = \phi$

<u>DFA</u>



only these 3 operations

Regular expression

+ union

· concatenation

a^* - unary ~~op~~ repeat n no. of times

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

① L = $\{w \mid w \in \Sigma^*$ w ends with ab$\}$

$r_1 = a \cdot b$. $r_2 = a^* \rightarrow r_2 \cdot r_1$ ✓

$r = a^* ab$ ✓ ~~r_1~~ $= b^* ab$ ✓

$\Rightarrow$ ~~r~~ $\boxed{\underset{r_3}{(a+b)^*} \cdot \underset{r_2}{a} \cdot \underset{r_1}{b}}$ ✓

② L = $\{w \mid w \in \Sigma^*$ & w has substring ab$\}$.

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

③ L = $\{w \mid w \in \Sigma^*$ & w has no substring ab$\}$

$L_1 = (a+b)^*$ → all possible comb^n s of a, b

$L_2 = (a+b)^* ab (a+b)^*$ → substring ab.

$\Rightarrow L_1 - L_2$

but, - is not an operator

$\Rightarrow b^* a^*$

$(b+a)^*$ is ✗ same as $(a+b)^*$

$(ba)^*$ ✗
↳ 0, ba, baba
b, a etc. missing

$b^* + a^*$
✗
ba together not possible

Q. Write the reg expⁿ to generate strings of even length.

↳ $(aa + bb + ab + ba)^*$

or

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

* $a+b = b+a$ commutative. union.
* $a \cdot b \neq b \cdot a$ not commutative over .:

* $(a+b)+c = a+(b+c)$ } Associative.
* $a\cdot(b\cdot c) = (a\cdot b)\cdot c$

* $a(b+c) = a\cdot b + a\cdot c$ } Distributive.
* $(a+b)c = a\cdot c + b\cdot c$

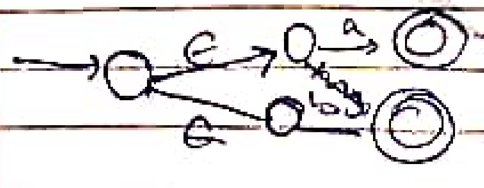
* $\epsilon + a \neq a$ ✗ ϵ is not an implicit symbol (LHS, RHS)
  $\epsilon \cdot a = a \cdot \epsilon = a$ ✓

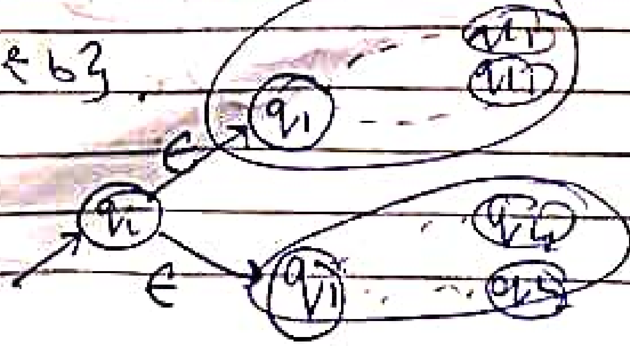
①  ϕ . null set.

②  $\{\epsilon\}$ set has member-null string.

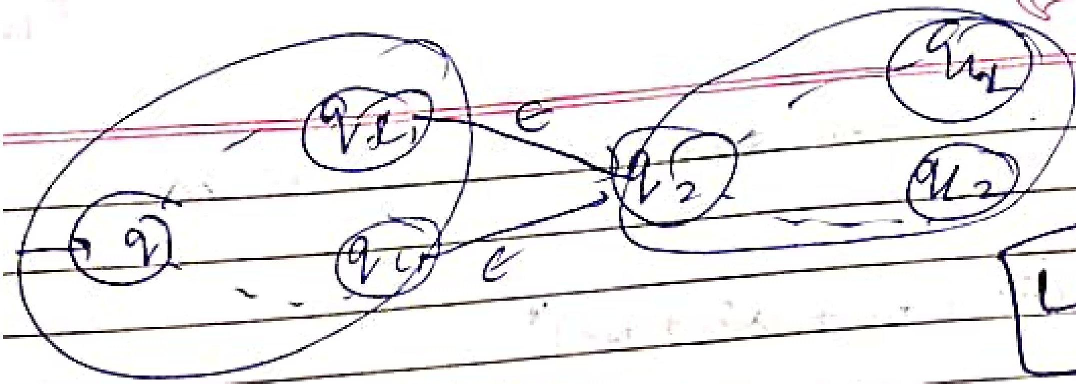
③ M1  $L = \{a\}$

M2  $L_2 = \{b\}$.





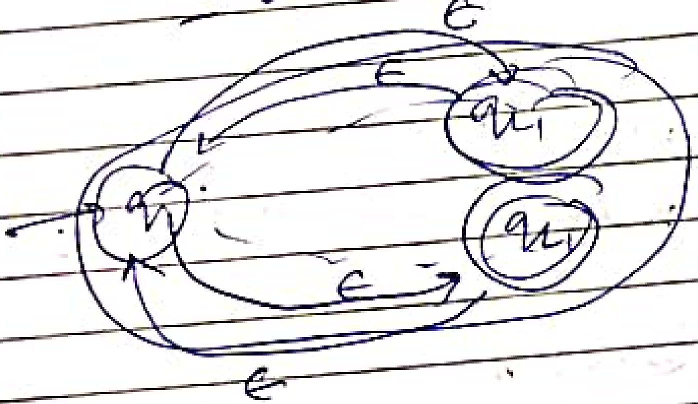
$L_1 \cup L_2$



$L_1 \cdot L_2$

$w_1 \epsilon w_2$.

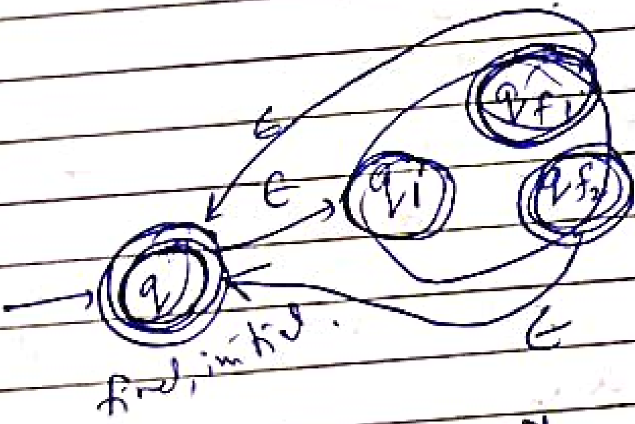
$= w_1 \cdot w_2$



L^*

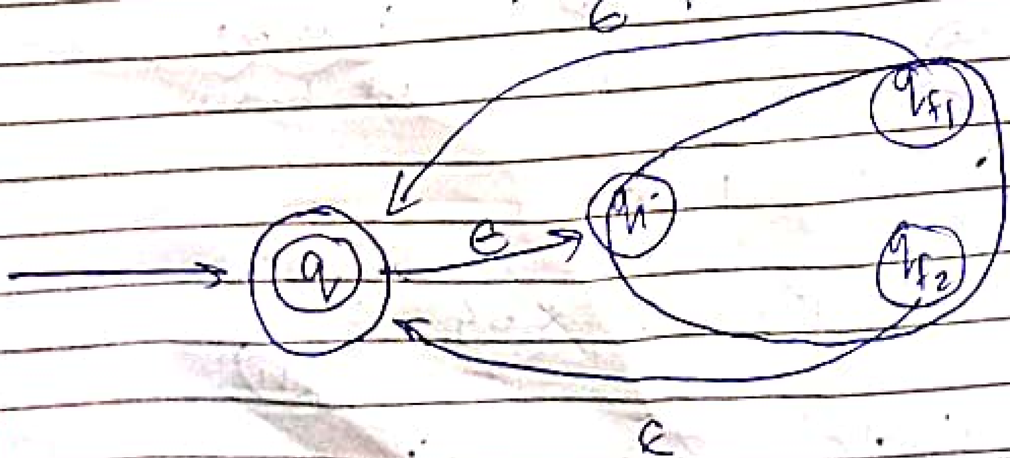
final state $\xrightarrow{\epsilon}$ initi
(repetition)

initial state $\xrightarrow{\epsilon}$ final st
null string
(ε ~~state~~)



$(a+b)^* ab$

L^* !

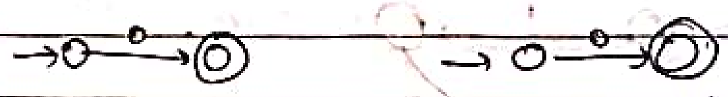


classmate

Date 15/11/2024
Page 17

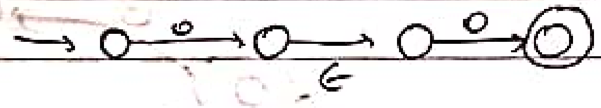
RE to ε NFA

$r = \underbrace{(00+1)^*}_{r_1} \cdot \underbrace{(10)^*}_{r_2}$ $\quad r_1 = (00+1)^* = (r_{11})^*$ $\quad r_{11} = 00 + 1 \quad r'_{11} \; r''_{11}$



0// $\quad$ concatenate $\quad$ 0//

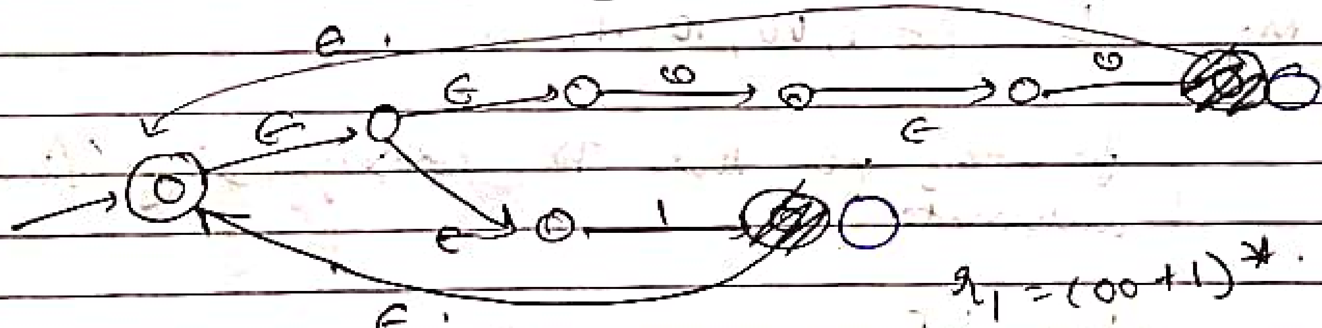
from final of one to starting of another. add ε connection & remove finalness of middle one.



00// $\quad$ 1// $\quad$ union

add new state epsilon connections

$r_{11} = 00 + 1$

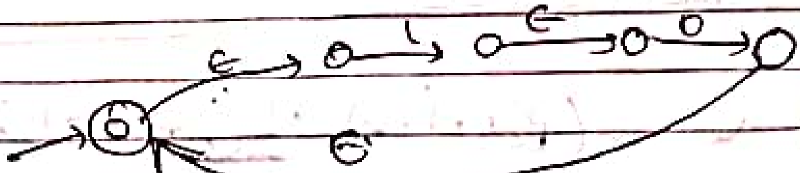


$r_1 = (00+1)^*$

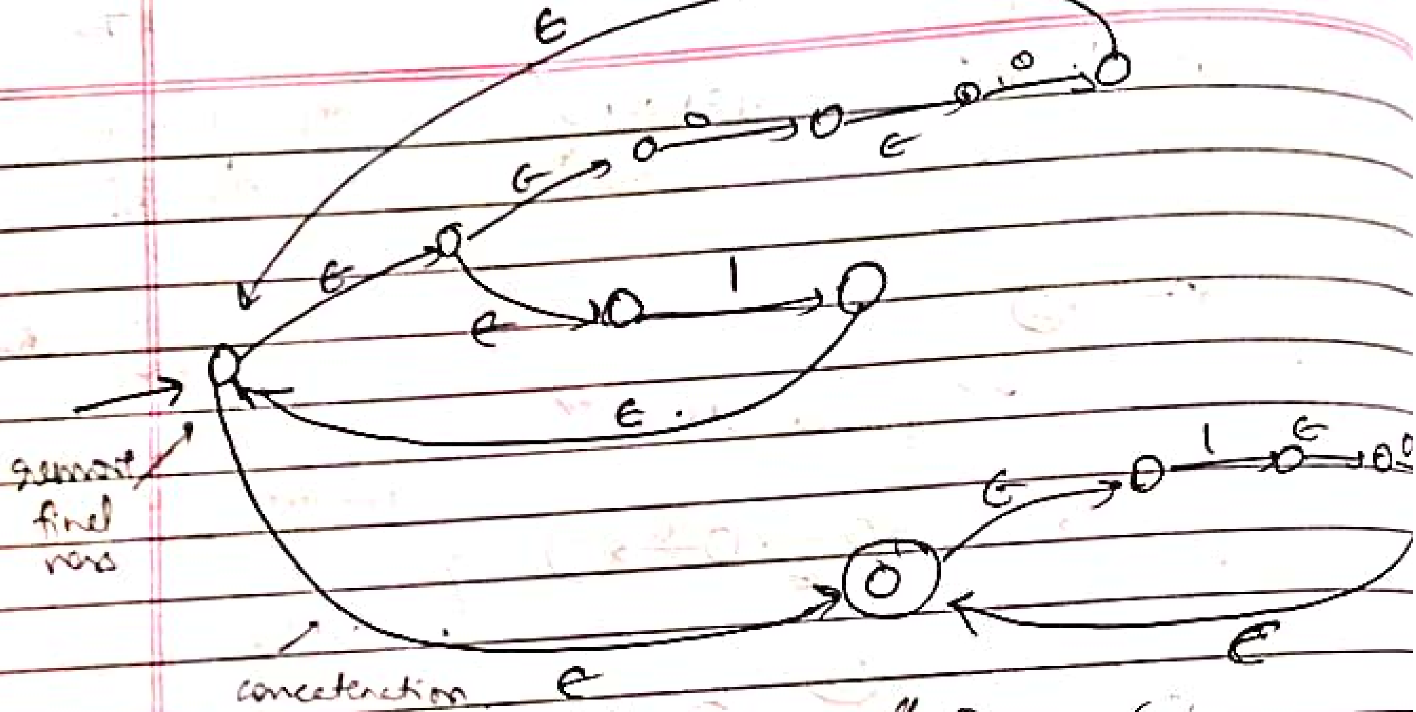
add new state epsilon connections. come back to this state with epsilon connections. remove theese' finalness. Make it final.

concatenate

10//



$r_2 = (10)^*$



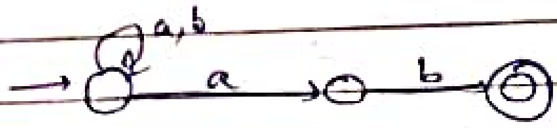
connection final of one to initial of another

$r = (00+1)^* \cdot (10)^*$

M: $L = \{\epsilon, 1, 00, 10, 11, \ldots\}$.

for every RE there is a corresponding finite automata.

$\boxed{M \Rightarrow RE}$

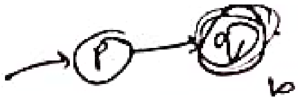


$(a+b)^*(ab)$.

$L(p, q, k+1)$.

$\boxed{= L(\cancel{p,q,k}) \cup L(p, \cancel{k+1, k}) \cdot \cancel{L(k+1, q, k}) \cdot \cancel{L(k+1, k+1, k)}^*}$

$L(p, q, k+1) = L(p,q,k) \cup L(p,k+1,k) \cdot L(k+1,k+1,k)^* \cdot L(k+1,q,k)$

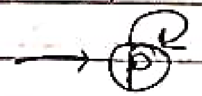


recursion base class: $r(p,q,0) = a$ if $\delta(p,a)=q$ (direct transition) $(p \neq q)$.

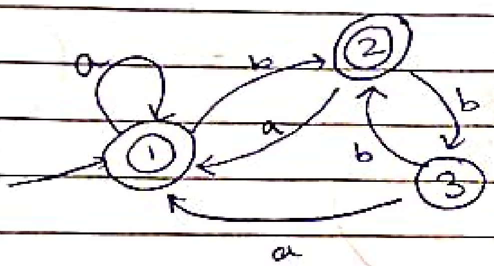
$r(1,2,3) = r(1,2,2) + r(1,3,2) \cdot r(3,3,2)^* \cdot r(3,2,2)$

terminate when $k=0$ or no intermediate states.

if $p=q$, $r(p,q,0) = \epsilon + a$



$\therefore r(p,q,0) = \begin{cases} a & p \neq q \quad \delta(p,a)=q \\ \epsilon + a & p = q \end{cases}$



1,2 are final states

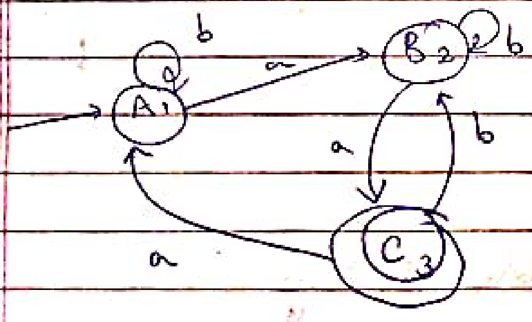
$r.e = \underset{1 = \text{starting state}}{r(1,1,3)} + r(1,2,3)$ → OR (union)

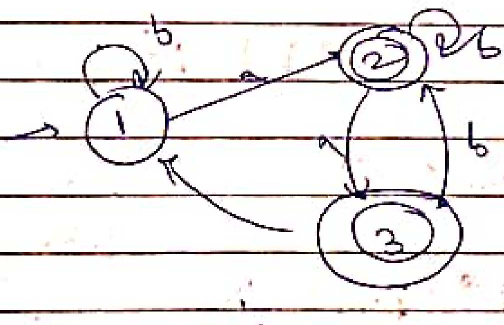
$r(1,1,3) = \underline{r(1,1,2)} + r(1,3,2) \cdot r(3,3,2)^* \, r(3,1,2)$

$r(1,1,2) = \underbrace{r(1,1,1)}_{a^*} + r(1,2,1) \, r(2,2,1)^* \, r(2,1,1)$

$a^* + a^*b \cdot a\,a^*b \quad aa^*$

↳ going from 1 to 2 using 1 as intermediate.

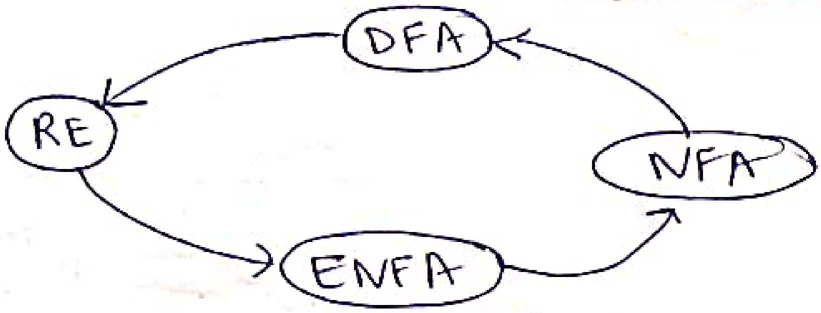




$r(1,3,3) + r(1,2,3)$ } use + to find expⁿ.

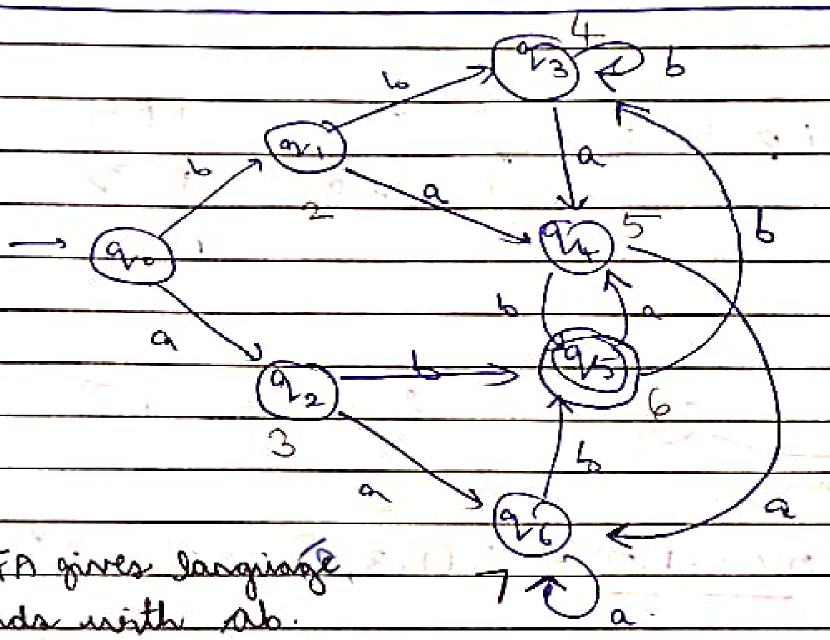
classmate
Date 20/11/2024
Page 20

8#

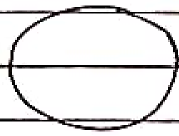


~~A: r(1,1,3) + r(1,2,3)~~

Q)



⇒

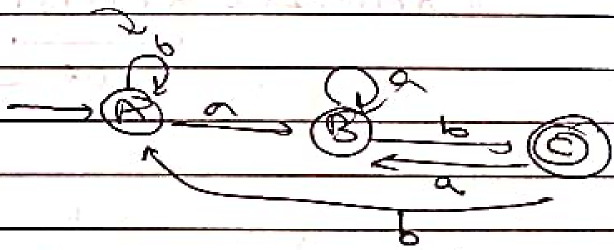


~~state minimiz^n~~

DFA gives language ends with ab.

~~minimiz^n alg.~~

$r(1, 6, 7)$
p q k+1



(p, q) $p \neq q$.

$w \in \Sigma^*$ retain distinguishable states.

$\delta^*(p, w) = r$
$\delta^*(q, w) = r$.

Date: 20/11/2024
Page: 21

$q_0 \equiv q_1$ $\qquad$ ~~$q_2 \equiv q_0$~~ $\qquad$ $q_2 \equiv q_4 \equiv q_6$:

$q_1 \equiv q_3$

$\Rightarrow q_0 \equiv q_1 \equiv q_3$

~~$(q_0\ q_1)$~~	~~$(q_1\ q_2)$~~	~~$(q_2\ q_3)$~~	~~$(q_3\ q_4)$~~	~~$(q_4\ q_5)$~~	~~$(q_5\ q_6)$~~
~~$(q_0\ q_2)$~~	~~$(q_1\ q_3)$~~	~~$(q_2\ q_4)$~~	~~$(q_3\ q_5)$~~	~~$(q_4\ q_6)$~~	
~~$(q_0\ q_3)$~~	~~$(q_1\ q_4)$~~	$(q_2\ q_5)$	$(q_3\ q_6)$		
~~$(q_0\ q_4)$~~	$(q_1\ q_5)$	$(q_2\ q_6)$			
$(q_0\ q_5)$	$(q_1\ q_6)$				
$(q_0\ q_6)$					

If final state is present in the pair, they are distinguishable (distinguishability test not required)

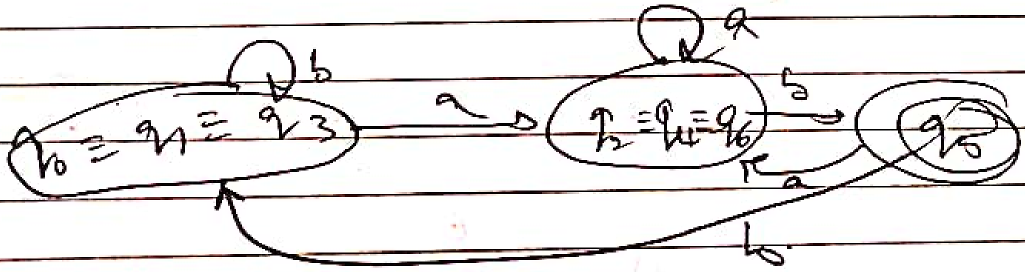
$p\ q \qquad p \not\equiv q$

(r, s)

$\delta^*(p, w) = r$
$\delta^*(q, w) = s$

(q_1, q_2) distinguishable.
$q_1, b = q_3$
$q_1, b = q_5 \rightarrow$ final state $\qquad q_3\ q_5$ distinguishable.

refer the qn for transitions.



Date 20/11/2024
Page 22

✓ = distinguishable.

q_1						
q_2	✓	✓				
q_3			✓			
q_4	✓	✓		✓		
q_5	✓	✓	✓	✓	✓	
q_6	✓	✓		✓		✓
	q_0	q_1	q_2	q_3	q_4	q_5

21/11/2024

Closure properties of RL

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

$L_1 \quad L_2$

proof ✱

$L_3 = L_1 \cdot L_2$

$L_4 = L_1^*$

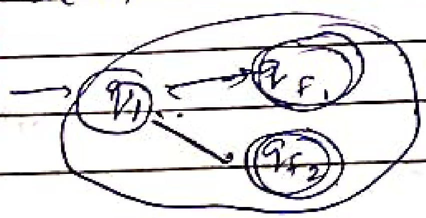
$L_5 = L_1 \cup L_2$

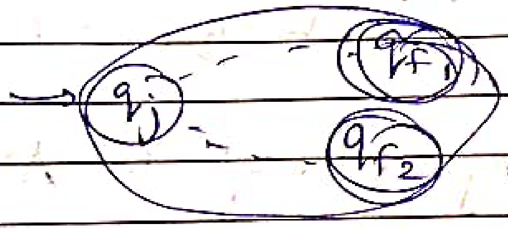
$L_6 = L_1 \cap L_2$

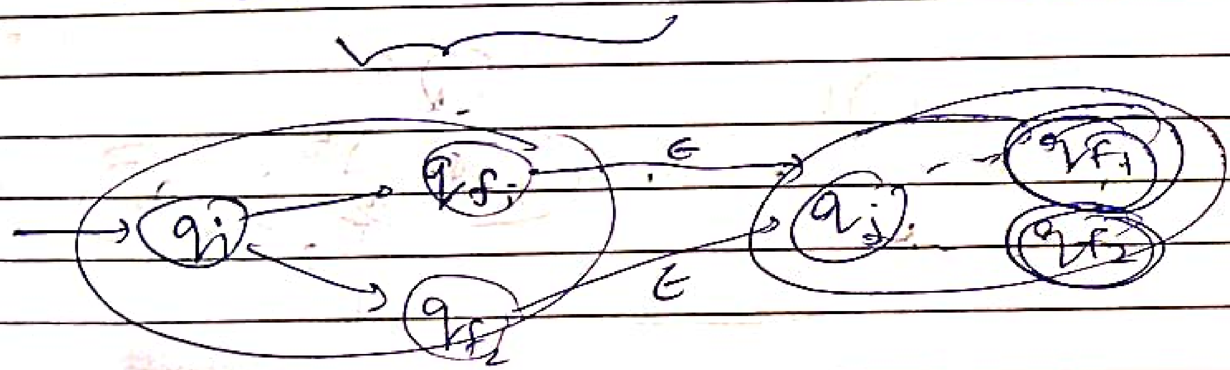
$L_7 = \bar{L_1}$

$L_8 = L_1 - L_2$

$L(M_1) = L_1 \quad , L(M_2) = L_2$





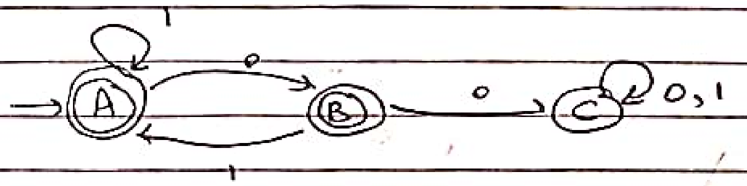


Compelment : $L_7 = \overline{L_1}$ (Invert all final and non final states - make final states non final and non final as final.)

Σ^* = universal set.

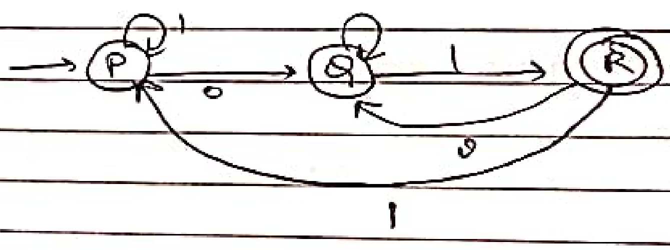
$\Sigma^* - L = \overline{L}$

Q) M_1:



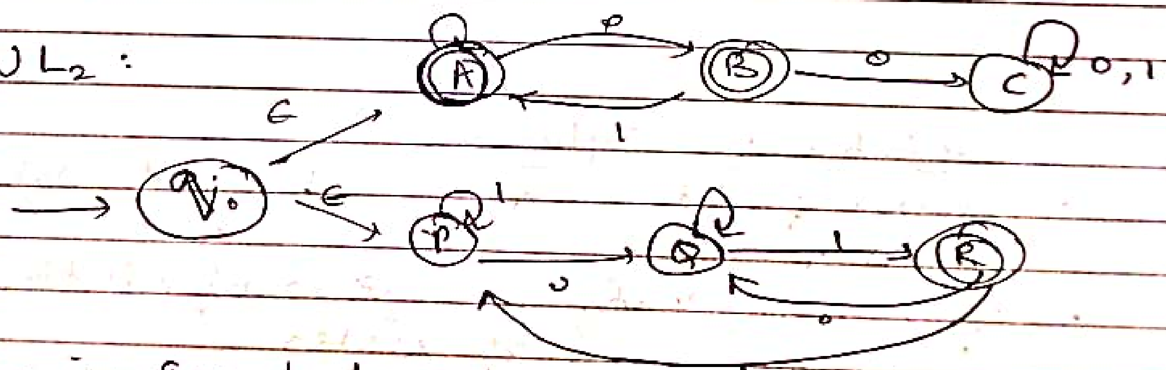
dfa

M_2 :



dfa

$L_1 \cup L_2$:



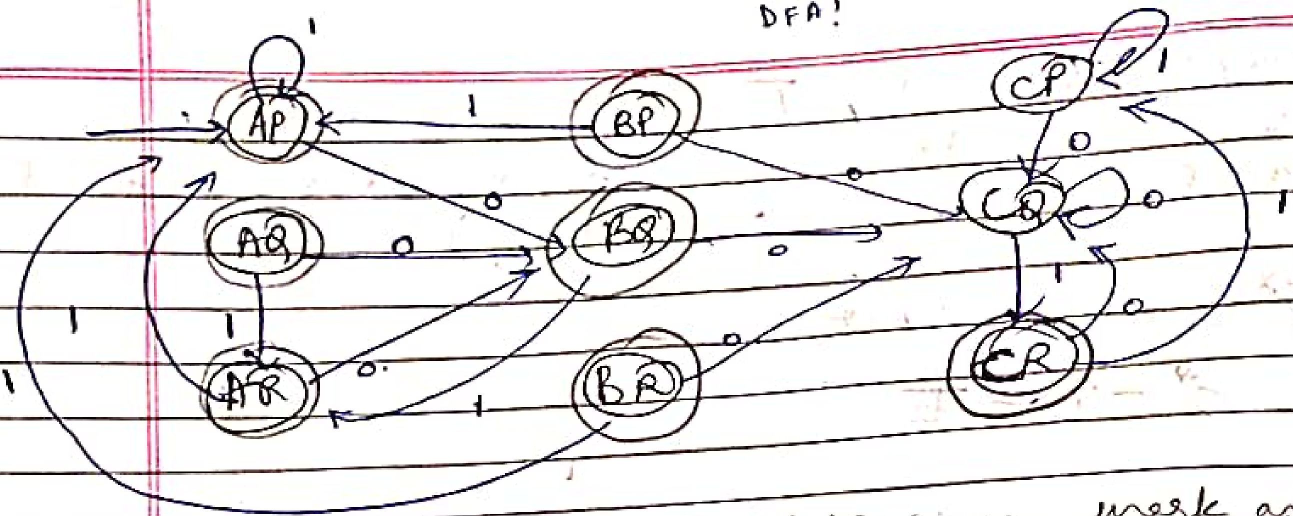
this is fine but it is εNFA not DFA.

$M_1 = (Q_1, \Sigma_1, \delta_1, q_1, F_1)$
$M_2 = (Q_2, \Sigma_2, \delta_2, p_1, F_2)$
$M_3 = (Q_1 \times Q_2, \Sigma_1, \delta_3, (q_1, p_1), F)$

A, P start states

Date 21/11/2024
Page 24

DFA!



A, B, R are final states so mark as shown for ∪ operation. [if any one is final].

for ∩ operation, both should be final in a pair : AR, BR.

for − operation $L_1 - L_2$
double circle that state with only final state of L_1 : AP, AQ, BP, BQ.

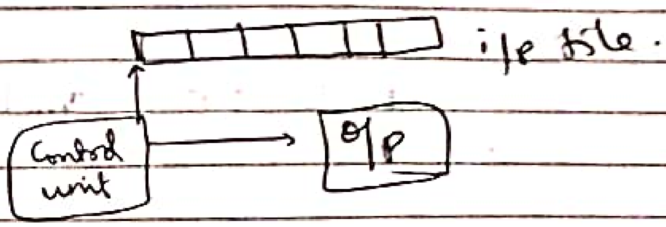
$L_2 - L_1$: only CR is final state.

Finite Automata Applications

- Software for designing & checking the behavior of digital circuits.
- Software for verifying systems of all types that have a finite no. of states (ex: stock market transaction, commⁿ/n/w protocol).
- Software for scanning large bodies of text (ex: web pages) for pattern finding.
- Lexical analyzer of a typical compiler.

Finite automata : mathematical model for study of abstract machine.

- i/p file
- control unit
- o/p : accept or reject.



3 types of Finite Automata.

1. Deterministic Finite Automata - (DFA)
2. Non Deterministic Finite Automata - (NFA)
3. Non Epsilon NFA - (ENFA)

* Language = set of strings from which statements can be obtained. L is said to be a language over Σ only if $L \subseteq \Sigma^*$
* Alphabet = finite, non-empty set of symbols
* Symbol Σ is used to denote alphabet.
* String = finite sequence of symbols chosen from Σ
*

ϵ = epsilon is empty string.

$|w|$ = length of string w - no. of non-ϵ char.

xy is a concatenation of 2 strings x & y.

Let Σ be an alphabet

- Σ^k = set of all strings of length k.
- $\Sigma^* = \Sigma^0 \cup \Sigma^1 \cup \Sigma^2 \cup \cdots$ Kleen closure
- $\Sigma^+ = \Sigma^1 \cup \Sigma^2 \cup \Sigma^3 \cup \cdots$ Kleen positive.

$\Sigma = \{0, 1\}$. $\Sigma^0 = \{\epsilon\}$

$\Sigma^1 = \{0, 1\}$ $\Sigma^2 = \{00, 01, 10, 11\}$, $\Sigma^3 = \{000, 001, \cdots 111\}$

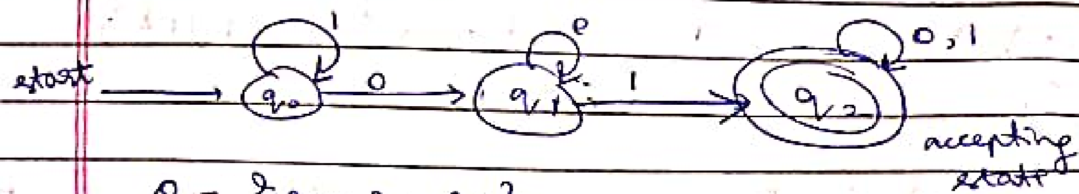
Σ^* = set of all strings (of all possible length including 0) over the given alphabet Σ.

$\Rightarrow$ $L = \{ \}$ $\hookrightarrow$ empty

$L = \{\epsilon\}$ $\hookrightarrow$ string of length 0

Let $L = \{\epsilon\}$ then $L \neq \phi$

Pictorial represent^n of FA : transition graph
tabular represent^n of FA : transition table



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

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

$\delta(V_i, a) = V_j$ → next state — if there is a transition from V_i to V_j on i/p symbol a

current state, i/p symbol

start state : q_0

final state, $F = \{q_2\}$

$\delta(q_0, 0) = q_1$
$\delta(q_0, 1) = q_0$
$\delta(q_1, 0) = q_1$
$\delta(q_1, 1) = q_2$
$\delta(q_2, 0) = q_2$
$\delta(q_2, 1) = q_2$

Transition table

symbols

δ	0	1
→ q_0 (start)	q_1	q_0
q_1 (intermediate)	q_1	q_2
$*q_2$ (final)	q_2	q_2

states

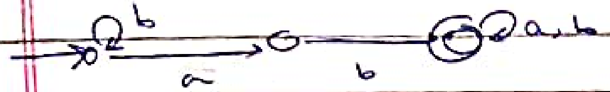
Deterministic Finite Automata consists of 5 Tuple:

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

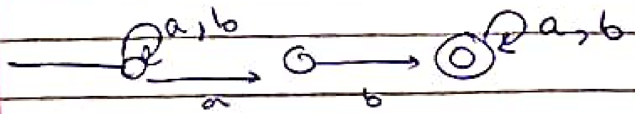
- Q → finite set of states
- Σ → finite set of i/p symbols
- q_0 → start state
- F → set of accepting states
- δ → transition f^n mapping b/w $Q \times \Sigma \Rightarrow Q$

$L_1 = \{w \mid w \in \{a,b\}^*$ where w has a substring $ab\}$

$(a+b)^* \; ab \; (a+b)^*$ RE.



DFA.



NFA.

$L_2 = \{w \mid w \in \{a,b\}^*$ where w is a palindrome$\}$

$L_3 = \{\epsilon, ab, aabb, aaabbb, \ldots\}$ → a comes before b.

↳ $a^n b^n \mid n \geq 0$

if the lang is not ~~auto~~ regular, these FA can't be used, we need different finite automatas.

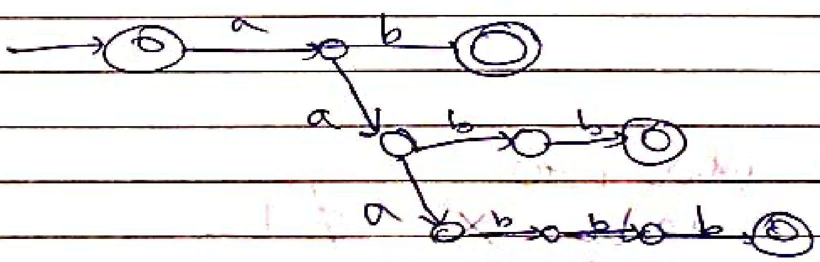
- Read only memory.
- Read only from LHS to RHS.

$L_3 = \{\epsilon, ab, aabb, aaabbb\}$.

$a^n b^n \mid n \geq 0$.

↳ infinite lang (not regular).

$L = \{\epsilon, ab, aabb, aaabbb\}$

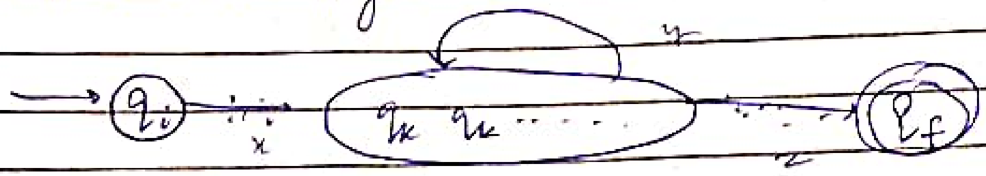


* no cycles

Prove that

Q) $L = \{a^n b^n \mid n \geq 0\}$ is not regular.

A: Assume L is regular. Model = M.



$w \in L$ x, y, z - substrings

$w = x|y|z$

pumping Lemma of Reg Lang [used to prove that given lang is not regular]

$|w| \geq n = xyz$

$\boxed{① |xy| \leq n \quad \cancel{|y| \geq 0} \; ② |y| \geq 1}$

for ex: $\overset{x}{aaa}|\overset{y}{b}|\overset{z}{bb}$

if y = 0, pumping null string ϵ won't make any difference.

$\boxed{x y^i z}$ for $i \geq 0$

must be a ~~[illegible]~~ member of L

$\overset{x}{aaa}|\overset{y}{b}|\overset{z}{bb}$ $\overset{x}{aa}|\overset{y}{a\,b}|\overset{z}{bb}$

$\overset{x}{aaa}|\overset{y}{bbb}|\overset{z}{\epsilon}$

partition is not unique but make sure to follow $|xy| \leq n$, $|y| \geq 1$

xy^iz

$(aaa)(b)^i(bb)$ where $i \geq 0$

$i = 0 \rightarrow aaabb \times \notin L$

$i = 2 \rightarrow aaabbbb \times \notin L$

By contradiction to the assumption that the lang is regular, the language is thus not regular.

Hence proved.

Context Free Grammar

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

- $S \in V$: Start Variable.
- P : Finite set of Productions. $V \to (V \cup T)^*$ — only one var → var + i/p
- T : finite set of i/p symbols (Terminals)
- V : Finite set of variables.

minimum one var is req.

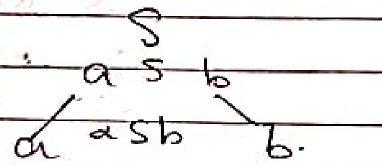
$L = \{a^n b^n \mid n \geq 0\}$. $= \{\epsilon, ab, aabb, \ldots\}$.

$V = \{S\}$

$T = \{a, b\}$

$P = \{S \to aSb$

$S \to \epsilon\}$



$aa\epsilon bb = aabb$.

$S \Rightarrow \epsilon$

$S \Rightarrow ab$; $S \to aSb$.

$\Rightarrow aaSbb$; $S \to aSb$

$\Rightarrow aaaSbbb$; $S \to aSb$;

$\Rightarrow aaabbb$ $S \to \epsilon$

$L(G) = \{w \mid w \in T^*$ such that $S \overset{*}{\underset{G}{\Rightarrow}} w\}$

Sentential form

Q) $L = \{w \mid w \in \{a,b\}^*$ & w is a palindrome$\}$.

A: $V = \{S\}$

$T = \{a, b\}$

$P = \{ S \to \epsilon$

$S \to a$

$S \to b$

$S \to aSa$

$S \to bSb \}$.

or : $P = \{S \to \epsilon \mid a \mid b \mid aSa \mid bSb\}$

$S \Rightarrow aSa$: $S \to aSa$

$\Rightarrow aaSaa$: $S \to aSa$.

$\Rightarrow aabSbaa$: $S \to bSb$.

$\Rightarrow aabaSabaa$: $S \to aSa$.

$\Rightarrow aabaaabaa$: $S \to \epsilon$.

Q) $L = \{w \mid w \in \{a,b\}^*$ such that $N_a(w) = N_b(w)\}$

$V = \{S\}$

$T = \{a, b\}$

$P = \{S \to \epsilon \mid aSb \mid bSa\}$.

$S \Rightarrow SS$; $S \to SS$.

$\Rightarrow bSaS$: $S \to bSa$

$\Rightarrow baS$; $S \to \epsilon$

$\Rightarrow baaSb$; $S \to aSb$

$\Rightarrow baab$; $S \to \epsilon$.

so baab can also be derived.

$L(G) = \{w \mid w \in \{a,b\}$ $N_a(w) = N_b(w)\}$

Q) $L = \{ a^i b^j c^k \mid i = j \}$.

$[a^i b^j \mid i = j \Rightarrow a^n b^n]$

$\therefore S \rightarrow A B$; $A \rightarrow a^i b^j$, $B \rightarrow c^k$.

$A \rightarrow aAb \mid \epsilon$.

$B \rightarrow cB \mid \epsilon$

Q) $L = \{ a^i b^j c^k \mid i = j + k \}$.

$a \quad a \, b \quad c$

$\underbrace{a}_{\text{equal}} \; \underbrace{a \; b}_{\text{equal}} \; \underbrace{c}_{\text{equal}}$ 1) 2)

$aSc \mid aSb \mid \epsilon$.

↳ accepts $a a c b$ also but order: $a^i b^j c^k$

$S \rightarrow aSc \mid A$

$A \rightarrow aAb \mid \epsilon$

2 variables $V: \{S, A\}$ ..

ex. $\Rightarrow S$

aSc ; $S \rightarrow aSc$

$aaScc$; $S \rightarrow aSc$

$aaAcc$; $S \rightarrow A$.

$aa\,aAbcc$; $S \rightarrow aAb$.

$a\,a\,a\,bcc$; $S \rightarrow \epsilon$.

G:

Q) $S \to aB \mid bA$
$A \to a \mid aS \mid bAA$
$B \to b \mid bS \mid aBB$

{ $S \to aSb \mid bSa$ } with only one var

V: ~~{S, A, B}~~ {S, A, B}
T: {a, b}
L(G) = ?

A:

$S \Rightarrow aB$	$S \Rightarrow aB$	$S \Rightarrow aB$
$\Rightarrow ab$	$\Rightarrow abS$	$\Rightarrow aaBB$
(smallest)	$\Rightarrow abbA$	$\Rightarrow aabB$
	$\Rightarrow abba$	$\Rightarrow aabb$

$L(G) = \{ w \mid w \in \{a,b\}^* \; N_a(w) = N(b) \; ; |w| \geq 2 \}$ (except ϵ)
equal no. of a, b

ambiguous:

w_1: aababb

$S \Rightarrow aB$
$\Rightarrow abS$
$\Rightarrow abbA$
$\Rightarrow abbbAA$
$\Rightarrow abbbaA$ ×
not derivable

w_2: aababb

$S \Rightarrow aB$

①	②
aaBB	
aabB	aabSB
aabaBB	aabaBB
aababB	aababB
aababb	✓ aababb

} LMD

Right to left.

③

$S \Rightarrow aB$	aB
aaBB	aaBB
aaBb	aaBaBB
aaBb	aaBaBb
aabSb	aaBabb
aabaBb	aababb
aababb	

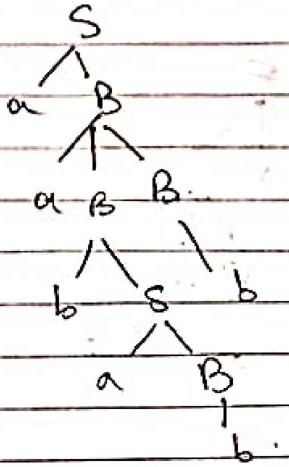
} RMD

* Left most deriv^n (LMD)
* Right most deriv^n (RMD)

* Any grammar is said to be ambiguous grammar if there exists atleast one str^{W} that corresponds to more than one LMD or RMD constructions.

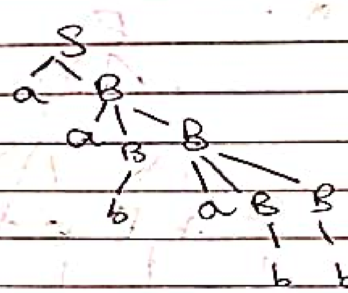
parsing | deriv^{n} tree.

2

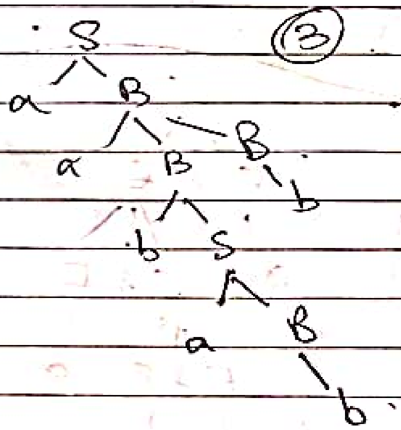


S → aB
aaBB
aabB
aabaBB
aababB
aababb

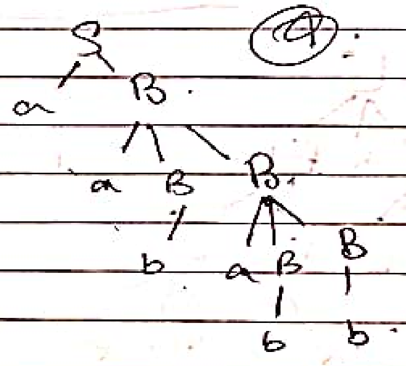
1



Q) 3



Q) 4

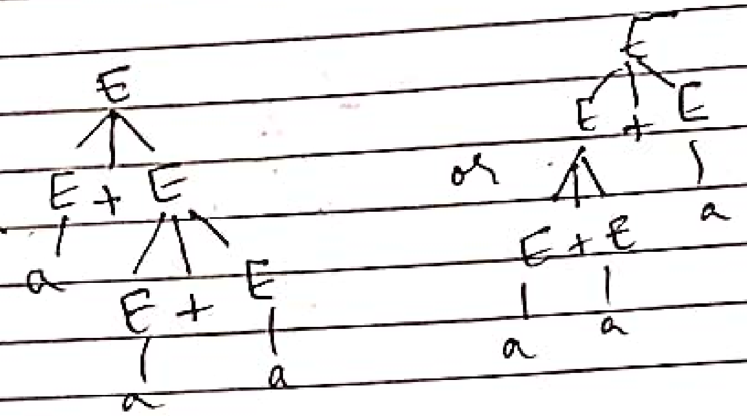


if there is an LMD, there is also an RMD for it.

Construct CFG to generate valid arithmetic expn.

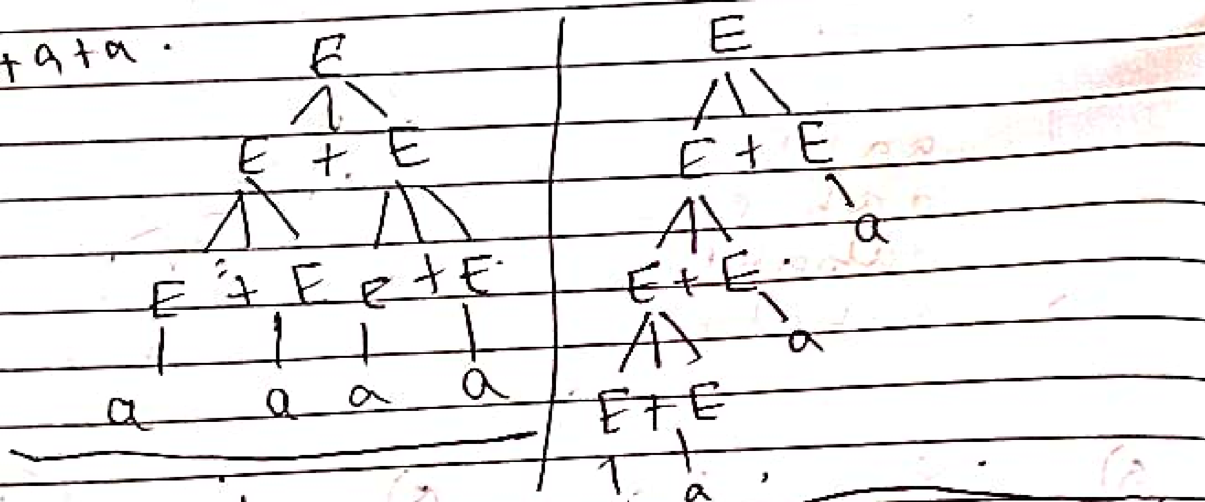
$E \rightarrow E+E \mid E-E \mid E*E \mid E/E \mid (E) \mid a \mid b \mid c.$

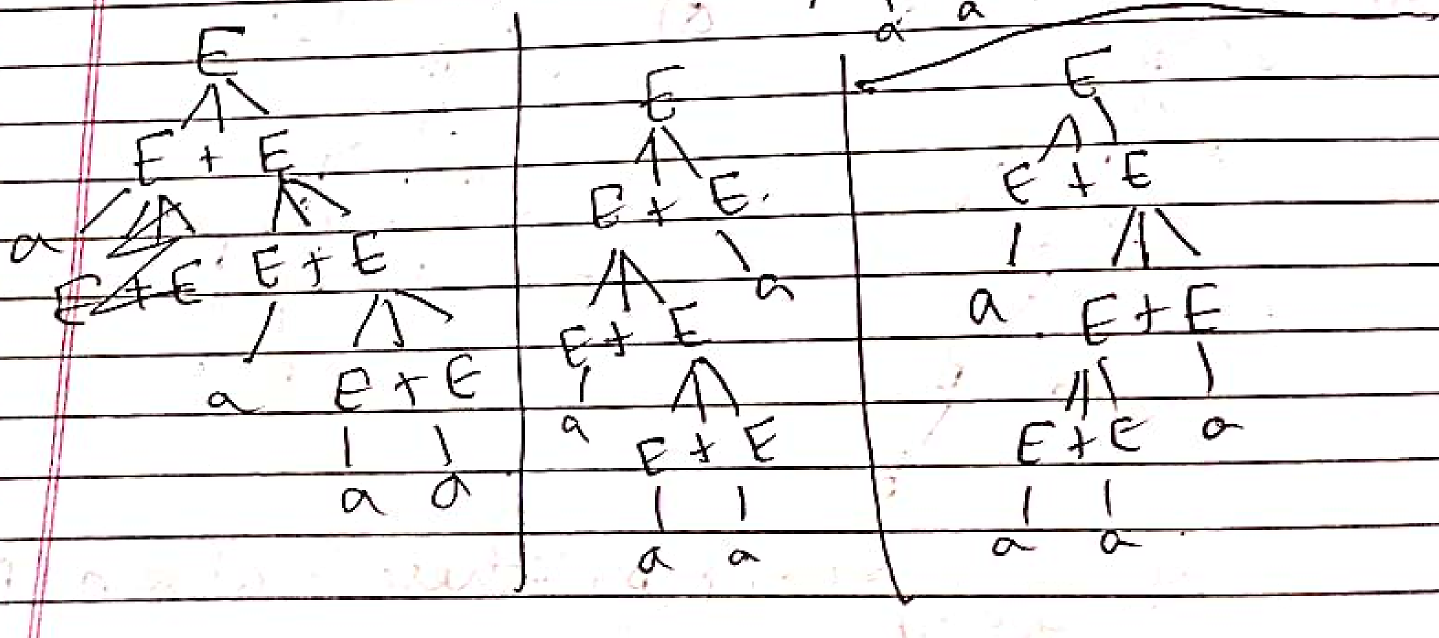
a+a+a



a+a+a+a.

No. of distinct parsing trees is 5.





Date 5/12/2024
Page 35

6 variables, 9 productions

Q) $S \to aA \mid bB$
$A \to aA \mid a$
$B \to bB$
$D \to ab \mid Ea$
$E \to ac \mid d$

$S \Rightarrow aA$
$\Rightarrow aa$
$\therefore aa \in L(G)$

$S \Rightarrow aA$
$\Rightarrow aaA$
$\Rightarrow aaa$
$\therefore aaa \in L(G)$

$L(G) = \{w \mid w \in \{a\}^*$ & $|w| \geq 1\}$

$S \to aA$
$A \to a \mid aA$
2 variables 3 prod"s.

Simplification of CFG

① * Identify useful variables.

Any var $A \in V$ is useful if $A \overset{*}{\Rightarrow} w$ $w \in T^*$
it is possible to derive a string of input symbols from this variable (terminal)

S, A, D, E ✓
B, C X terminal strings are not derivable

② * identify unreachable variables.

S, A ✓

B, C X D, E X

$\boxed{\begin{array}{l} S \to aA \\ A \to aA \mid a \end{array}}$ SIMPLIFIED.

8) $S \to aA \mid a \mid Bb \mid cC$.
$A \to aB$.
$B \to a \mid Aa$
$C \to cCD$
$D \to dd$

step1 : useful $\to$ S, B, D, A.
useless $\to$ ~~A~~, C.

step2 : reachable $\to$ S, A, B
unreachable $\to$ D, C
is useless.

D is reachable through C but C is useless

$\therefore$ usefull : S, A, B
useless : D, C.

$S \to aA \mid a \mid Bb$.
$A \to aB$
$B \to a \mid Aa$.

Q) $S \to ABCa \mid bD$
$A \to BC \mid b$
$B \to b \mid \epsilon$
$C \to c \mid \epsilon$
$D \to d$.

step1: S, A, B, C, D ✓
step2. S, A, B, C, D ✓

$L(G) = \{a, ba, bd, \ldots\}$
ϵ, b, c, d ✗

we need to simplify & eliminate ϵ or null prodns.

☆ Method of eliminating null or ϵ production.

remove B, C ϵ productions

1) nullable: B, C, A

$B \Rightarrow \epsilon$, $C \Rightarrow \epsilon$.
$A \Rightarrow BC$.
$\Rightarrow \epsilon C$.
$\Rightarrow \epsilon$

we can derive ϵ from the variables A, B, C.
$\therefore$ in this grammer, A, B, C are nullable variables.

include $B \to \epsilon$, $C \to \epsilon$ since we removed it.

2)
$S \to ABCa \mid bD \mid ACa \mid ABa \mid Aa$.
$A \to BC \mid b \mid C \mid B \mid \epsilon$
$B \to b$
$C \to c$
$D \to d$.

3) remove $A \to \epsilon$ connection and add all possibilities.

$S \to ABCa \mid bD \mid ACa \mid ABa \mid Aa \mid a \mid Ba \mid Ca \mid BCa$
$A \to BC \mid b \mid C \mid B$
$B \to b$
$C \to c$
$D \to d$

$L(G') = L(G) - \{\epsilon\}$.

$S \Rightarrow AB$
$\Rightarrow aB$
$\Rightarrow aC$
$\Rightarrow aD$
$\Rightarrow aE$
$\Rightarrow ad$

$S \Rightarrow AB$
$\Rightarrow aB$
$\Rightarrow ad$

* unit prodn

5)

$S \to AB$	$S \to AB$.	$S \to AB$	$S \to AB$
$A \to a$	$A \to a$.	$A \to a$	$A \to a$
$B \to c \mid b$	$B \to C \mid b$	$B \to C \mid b$	$B \to d \mid Ab \mid bC \mid b$
$C \to D$	$C \to D$.	$C \to d \mid Ab \mid bC$	
$D \to E \mid bC$	$D \to d \mid Ab \mid bC$		
$E \to d \mid Ab$			

unit prod's are modified.
$X \to Y$ $Y \to \{b\}$ $X \to b$.

ϵ prod^n ex:

Q) $S \to ABCa \mid bD$
$A \to BC \mid b$
$B \to b \mid \epsilon$
$C \to c \mid \epsilon$
$D \to d$.

① no useless Var.
② no unit prod^n
③ ϵ prod^n:
$S \to ABCa \mid bD \mid BCa \mid ACa \mid ABa \mid Ca \mid Aa \mid Ba \mid a$.
$A \to BC \mid b \mid C \mid B$.
$B \to b$
$C \to c$
$D \to d$.

unit prod^n.

$S \to ABCa \mid bD \mid BCa \mid ACa \mid ABa \mid Ca \mid Aa \mid Ba \mid a$.
$A \to BC \mid b \mid c \mid b$.
$C \to c$
$D \to d$

$\therefore$ it's better to use ϵ prod^n before unit prod^n.

Q) $S \to Aa \mid B \mid Ca$
$B \to aB \mid b$.
$C \to Db \mid D$.
$D \to E \mid d$
$E \to ab$.

① useless: A.
$S \to B \mid Ca$.
$B \to aB \mid b$
$C \to Db \mid D$.
$D \to E \mid d$
$E \to ab$.
② no ϵ prod^n
③ unit prod^n.
$S \to aB \mid b \mid Ca$.
$B \to aB \mid b$.
$C \to Db \mid ab \mid d$
$D \to ab \mid d$
$E \to ab$

④ $S \to aB \mid b \mid Ca$
$B \to aB \mid b$
$C \to Db \mid ab \mid d$
$D \to ab \mid d$

but E is now useless.
so it's better to use unit prod^n before useless var.

~~doing unit prodⁿ first might.~~

side effect of ϵ prod^n $\rightarrow$ unit prod^{n}s introd^n.

side effect of unit prod^n $\rightarrow$ introd^n of useless var.

$\therefore$ ORDER : 1) ϵ prod^n
2) unit prod^n
3) useless variables.

Normal form.

1) Chomsky Normal form. (CNF)
2) Gribech Normal form. (GNF).

CNF: for $A \in V$,

$\left\{ \begin{array}{l} A \rightarrow BC. \\ \text{or} \\ A \rightarrow a \end{array} \right.$

RHS? 2 var or single terminal.

Q) $S \rightarrow aB \mid bA$
$A \rightarrow a \mid aS \mid bAA$.
$B \rightarrow b \mid bS \mid aBB$.

} 3 var, 8 prod^{n}s

$S \rightarrow aB \} \rightarrow \left\{ \begin{array}{l} S \rightarrow XB \\ X \rightarrow a. \end{array} \right.$

$S \rightarrow bA \} \rightarrow \left\{ \begin{array}{l} S \rightarrow YA \\ Y \rightarrow b. \end{array} \right.$

~~$A \rightarrow \underbrace{b}_{Y}\underbrace{AA}_{P}$~~

$A \rightarrow a \mid XS \mid YP$
$P \rightarrow AA$.
$B \rightarrow b \mid YS \mid XQ$.
$Q \rightarrow BB$.

7 variables, 12 prod^{n}s $\longrightarrow$ normal form.

Language is still the same.

G :

Q) $S \to AACD$
$A \to aAb \mid \epsilon$
$C \to aC \mid a$
$D \to aDa \mid bDb \mid \epsilon$

HW

* ϵ prod^n elimination
* unit prod^n elimination
* normalisation.

G' : CNF = ?

ϵ prod^n elimination

A : remove $A \to \epsilon$, $D \to \epsilon$.

$S \to AACD \mid C \mid CD \mid ACD \mid AC \mid AAC$
$A \to aAb \mid ab$
$C \to aC \mid a$
$D \to aDa \mid bDb \mid bb \mid aa$

unit prod^n elimination

$S \to C$
$S \to AACD \mid aC \mid a \mid CD \mid AC \mid ACD \mid AAC$

take $P \to AA$, $Q \to CD$, $X \to a$, $Y \to b$,
$R \to XA$, $T \to DX$, $U \to DY$.

$S \to PQ \mid XC \mid a \mid CD \mid AC \mid AQ \mid PC$
$A \to RY \mid XY$
$C \to a \mid XC$
$D \to XT \mid YU \mid XX \mid YY$

Date 19|12|2024
Page 41

GNF

↳ $A \to aBCD\ldots$

↳ $A \to a$

ex: $S \to aB \mid bA$.

$A \to a \mid aS \mid bAA$.

$B \to b \mid bS \mid aBB$.

$A \to BC$

$B \to CA \mid b$

$C \to AB \mid a$.

take $A \Rightarrow A_1$, $B \Rightarrow A_2$, $C \Rightarrow A_3$

$A_1 \to A_2A_3$ $\quad$ $h \to l$

$A_2 \to A_3A_1 \mid b$. $\quad$ $h \to l$

$A_3 \to \underline{A_1}A_2 \mid a$. $\quad$ $l \to h$.

$A_1 \to A_2A_3$

$A_2 \to A_3A_1 \mid b$ $\qquad$ $(A_1 \to A_2A_3)$ $\quad$ $A_3A_1 \mid b$.

$A_3 \Rightarrow bA_3A_2 \mid a \mid A_3A_1A_3A_2$

take $X \to A_1A_3A_2X \mid \epsilon$.

- $\therefore A_3 \to bA_3A_2X \mid aX \mid bA_3A_2 \mid a$.

put A_3 in A_2

- $A_2 \to bA_3A_2XA_1 \mid aXA_1 \mid bA_3A_2A_1 \mid aA_1 \mid b$.

put A_2 in A_1

- $A_1 \to bA_3A_2XA_1A_3 \mid aXA_1A_3 \mid bA_3A_2A_1A_3 \mid aA_1A_3 \mid bA_3$

Δ. $X \to \ldots\ldots$

Grammar: for all regex, lang has a context free grammar
(inverse is not true!)

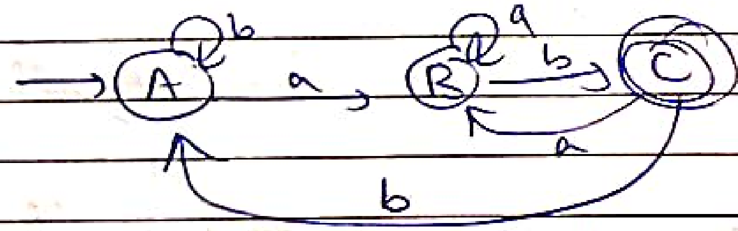
ex: $(a+b)^* ab$ (ending with ab)

$S \to aS \mid bS \mid ab$

All reg languages form a linear grammar
i.e.) for ex: ∴ $A \to BCD$ × never occurs in regex
anything like $A \to C \mid Bb \mid Cb \mid Cab$ holds
(no Branching)

$M = (Q, \Sigma, \delta, q_0, F)$ given
CFG $\to$ $G = (V, T, P, S)$ find
i) $\delta(q, a) = p$
$q \to ap$
or if $\delta(A, b) = A$ $\delta(A, a) = B$
$A \to bA \mid aB$

ex:



right:
$A \to aB \mid bA$
$B \to aB \mid bC$
$C \to aB \mid bA \mid \epsilon$
from $\delta(A, a) = B$, $\delta(A, b) = A$).

left: invert dfa then reverse
$C \to bB$
$B \to aB \mid aC \mid aA$
$A \to bA \mid bC \mid \epsilon$

$C \to Bb$
$B \to Ba \mid Ca \mid Aa$
$A \to Ab \mid Cb \mid \epsilon$

for every reg lang,
we can construct finite automata.

Regular lang → ① RG → LLG : Left Linear Grammar
RLG : Right Linear Grammar.

② FA
③ RE.

CFL ⟶ CFG

Push down Automata (PDA).

for every CFL, we can construct a PDA.

definition of PDA

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

Q : Finite set of states.
Σ : Set of i/p symbols.
T : Set of stack symbols.
δ : Transition function : $Q \times \Sigma \cup \{\epsilon\} \times T \rightarrow Q \times T^*$

$\delta(q_0, a, A) = (P, \epsilon)$ pop

$\delta(q_0, a, A) = (P, BC)$

B
C
Z

state, i/p symbol, what is on top of stack, next state, replace top symbol, push such that left most symbol is on top.

$q_0 \in Q$ start state.
$Z \in T$ Bottom of stack.

$F \subseteq Q$ final state.

classmate

Date 19/12/2024

Page 44

initially (q_0, aW, Z) $\quad$ (p, w, AZ)

read A

replace top symbol Z

by AZ

left most - A on top

(q_f, ϵ, T^*)

Accepted

else

rejected.

$$(q_0, aW, Z) \vdash^* (q_f, \epsilon, T^*)$$

acceptance by final state.

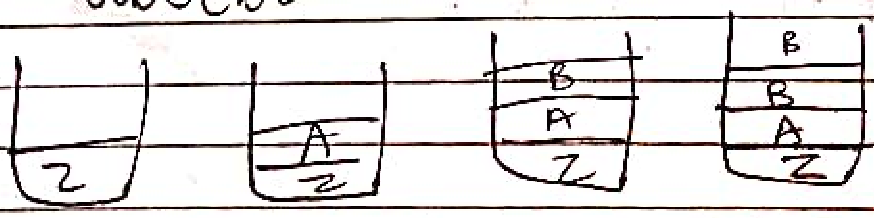
$$(q_0, aW, Z) \vdash^* (P, \epsilon, \epsilon)$$

acceptance by empty stack.

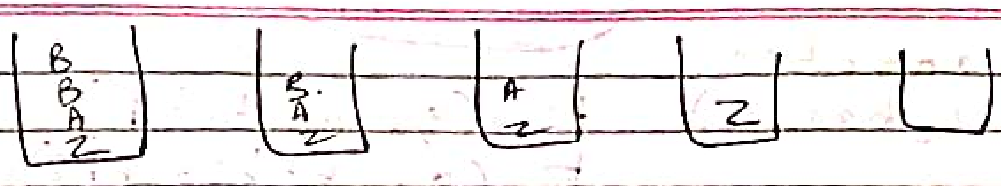
Q) $\Sigma = \{a, b, c\}$

strings of type $L = \{W c W^R \mid W \in \{a, b\}^*\}$

abbcbba.



keep pushing until C. Keep stack as it is and later push to pop state.



$\underbrace{abb}_{q_0}\, c\, \underbrace{bba}_{q_1} \mid \underbrace{\epsilon}_{q_2}$

$Q = \{q_0, q_1, q_2\}$
$\Sigma = \{a, b, c\}$
$T = \{A, B, Z\}$

q_0 start
Z is on top of stack
$F = \{q_2\}$

q_0 - push
q_1 - Pop
q_2 - final.

δ: $\delta(q_0, a, Z) = (q_0, AZ)$
$\delta(q_0, b, Z) = (q_0, BZ)$
$\delta(q_0, a, A) = (q_0, AA)$
$\delta(q_0, a, B) = (q_0, AB)$
$\delta(q_0, b, A) = (q_0, BA)$
$\delta(q_0, b, B) = (q_0, BB)$
} push

$\delta(q_0, c, Z) = (q_1, Z)$
$\delta(q_0, c, A) = (q_1, A)$
$\delta(q_0, c, B) = (q_1, B)$
} i/p symbol c changes state from push to pop

$\delta(q_1, a, A) = (q_1, \epsilon)$
$\delta(q_1, b, B) = (q_1, \epsilon)$
} pop

i/p symbol, top of stack should match.

$\delta(q_1, \epsilon, Z) = (q_2, \epsilon, Z)$
final state.

tracing

$x =$ aabcbaa

$(q_0, aabcbaa, Z) \vdash (q_0, abcbaa, AZ)$

$\vdash (q_0, bcbaa, AAZ)$

~~$(q_0, abcbaa, AZ)$~~ $\vdash (q_0, cbaa, BAAZ)$

$\vdash (q_1, baa, BAAZ)$

$\vdash (q_1, aa, AAZ)$

$\vdash (q_1, a, AZ)$

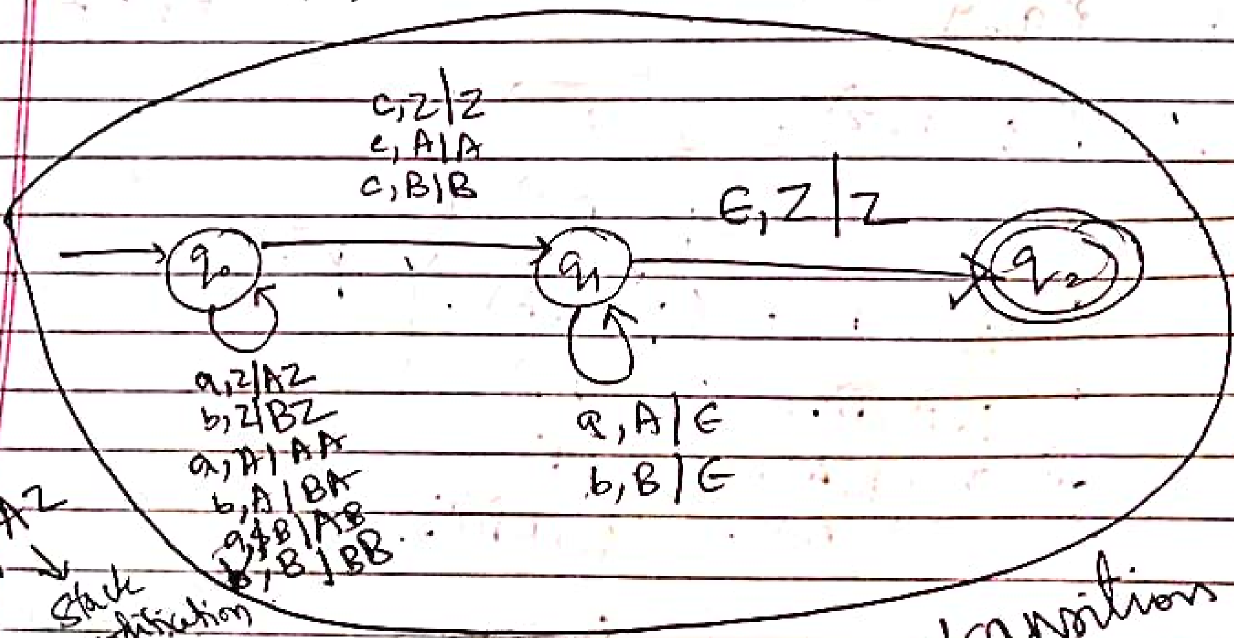
$\vdash (q_1, \epsilon, Z)$

$\vdash (q_2, \epsilon, Z)$

when c comes only change state from q_0 to q_1, stack stays the same.

accepted by final state.

stack is empty, string is processed, final state reached.



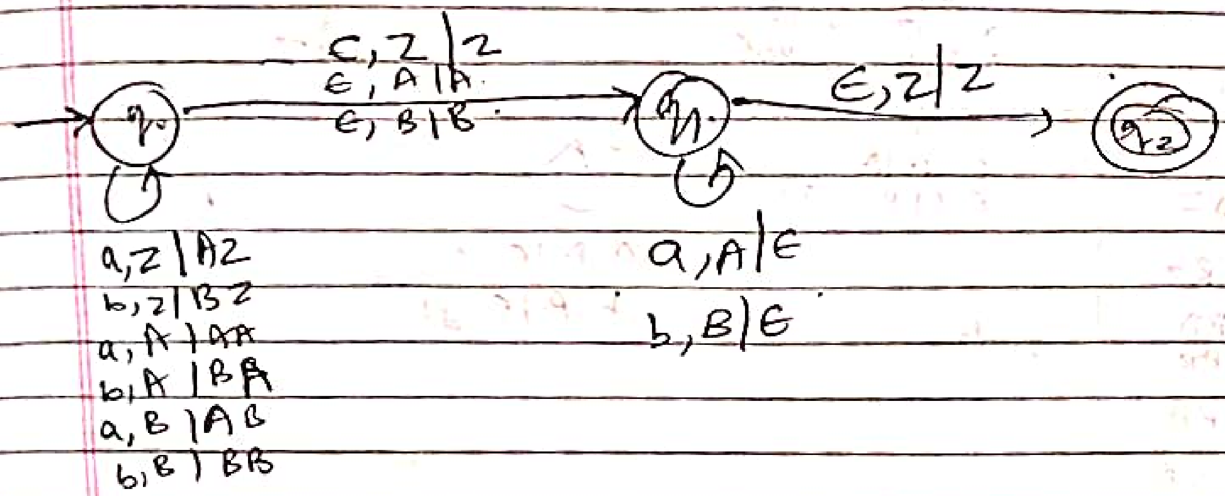
a, Z|AZ
↓ I/P symbol
↓ top of stack
↓ stack modification

transition diagram

(Palindrome of even length)

Date 19|12|2024
47

Q) $L = \{ w w^R \mid w \in \{a, b\}^* \}$



$(q_0, abba, Z)$

$\rightarrow (q_0, bba, AZ)$; $\rightarrow (q_1, abba, Z)$

$(q_1, abba, Z) \rightarrow (q_2, abba, Z)$ ✗ string is not processed

$(q_0, bba, AZ) \rightarrow (q_0, ba, BAZ)$; $\rightarrow (q_1, bba, AZ)$ ✗

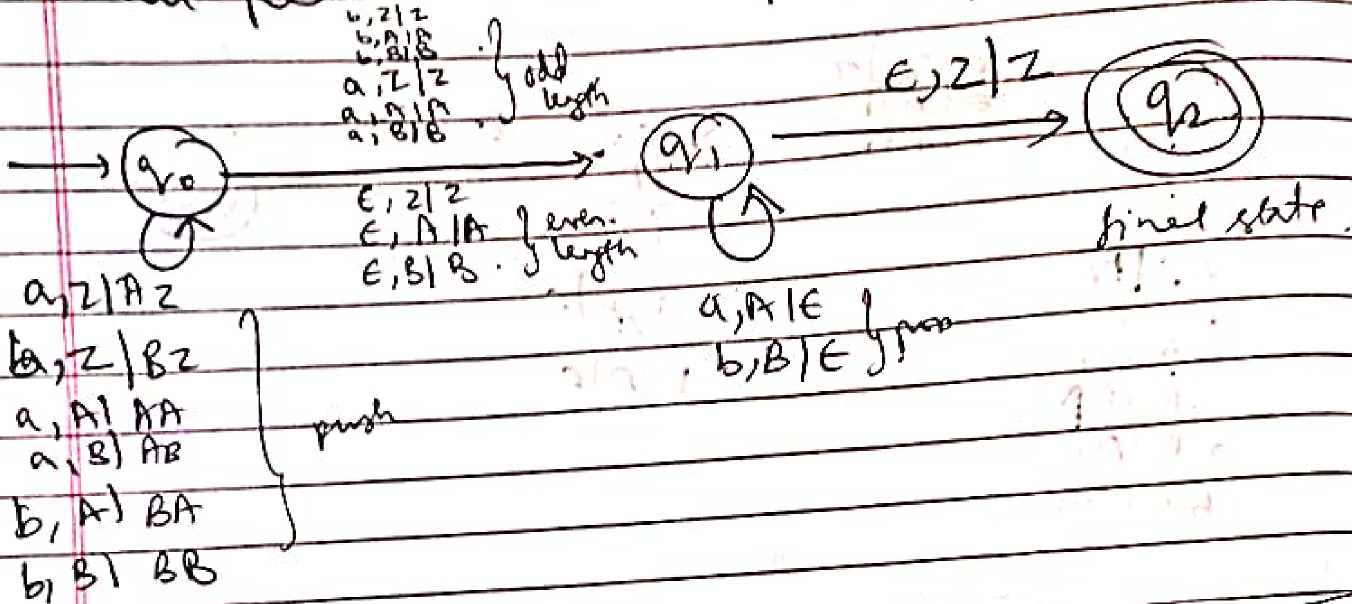
$(q_0, ba, BAZ) \rightarrow (q_0, a, BBAZ)$; $\rightarrow (q_1, ba, BAZ)$

$(q_0, a, BBAZ) \rightarrow (q_0, \epsilon, ABBAZ)$; $\rightarrow (q_1, a, BBAZ)$ ✗

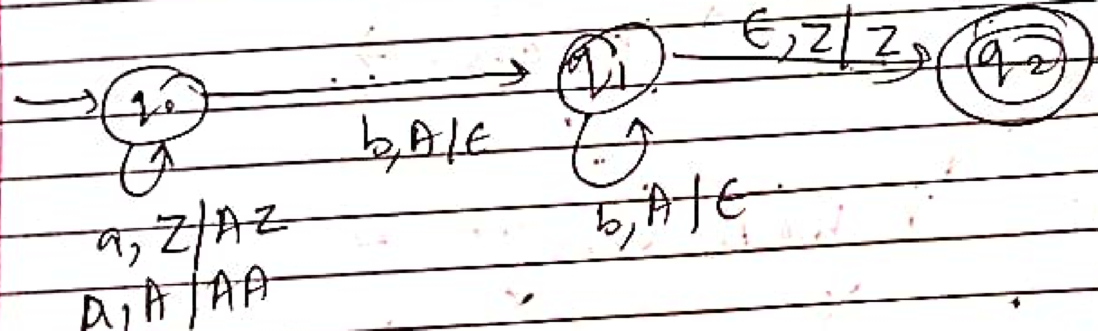
$(q_0, \epsilon, ABBAZ) \rightarrow q_1, \epsilon, ABBAZ$ ✗

$(q_1, ba, BAZ) \rightarrow (q_1, a, AZ) \rightarrow (q_1, \epsilon, Z) \rightarrow (q_2, \epsilon, Z)$ ✓

Construct PDA to recognise all palindromes irrespective of length.

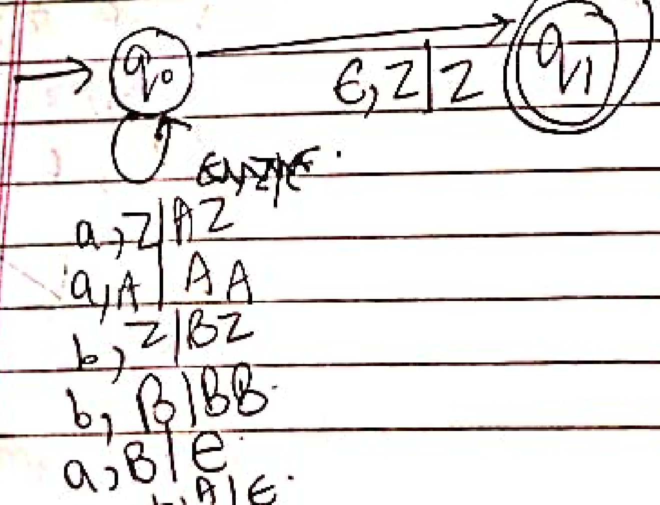


Q)

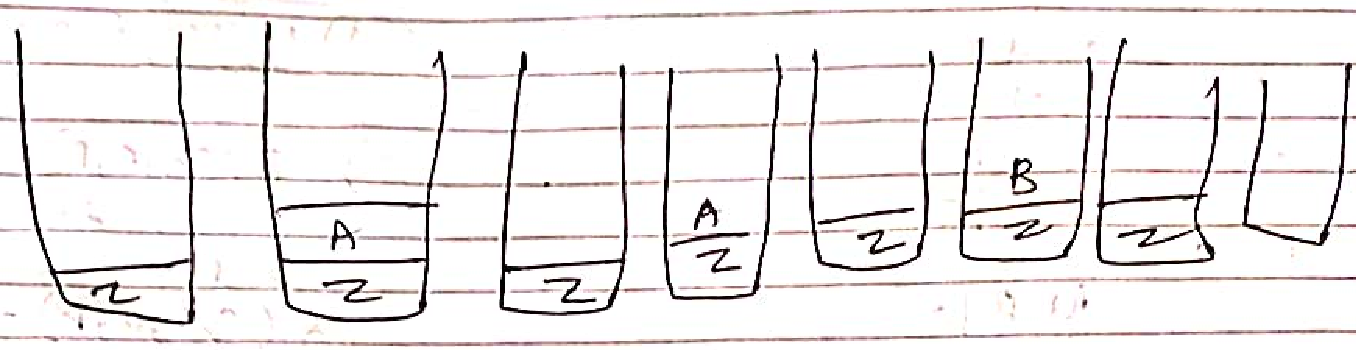


↳ PDA for $a^n b^n$

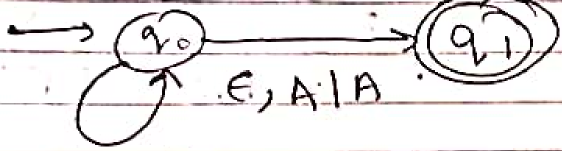
Q) equal no. of a,b



ababba.



Q) no. of a's is more than b. $N_a(w) > N_b(w)$.



a, Z | AZ
a, A | AA.
b, Z | BZ
b, B | BB
a, B | ε.
b, A | ε.

20/12/2024

PDA (non deterministic)

for deterministic, it will be mentioned explicitly

DPDA

1). $\delta(q, a, A) = (\quad)$

$q \in Q$ only one answer not a set. $\{ \}$ ×

$a \in \Sigma$

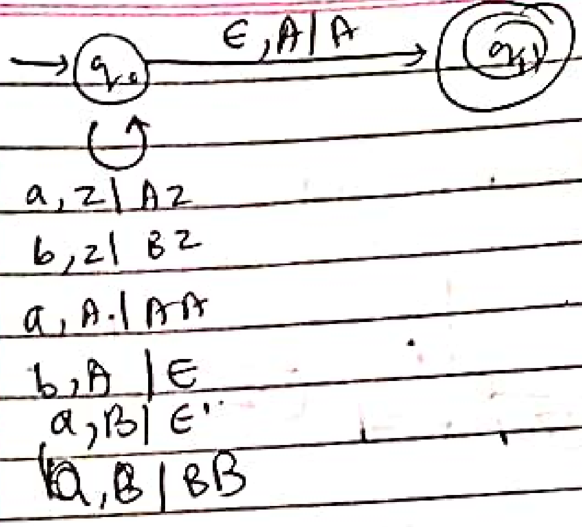
$A \in \Gamma$

Need not define δ for every i/p symbol or stack symbol in a PDA

2). if $\delta(q, \epsilon, z)$ is defined then $\delta(q, a, z)$ should not be defined.

$N_a(w) > N_b(w)$.

*



1) every δ gives single result.

2) $\delta(q_0, \epsilon, A) = (q_1, A)$ defined.

but
$\delta(q_0, a, A) = (q_0, AA)$
$\delta(q_0, b, A) = (q_0, BA)$.
Since both are defined 2nd cond^n is violated.
∴ NOT DPDA

* wcw^R.
① every δ gives single result.
② $\delta(q, \epsilon, z)$ not defined
but $\delta(q, a, z)$ is defined.

∴ DPDA.

* ww^R
① every δ gives single result
② $\delta(q, \epsilon, z)$ and $\delta(q, a, z)$ are both defined

∴ not DPDA.

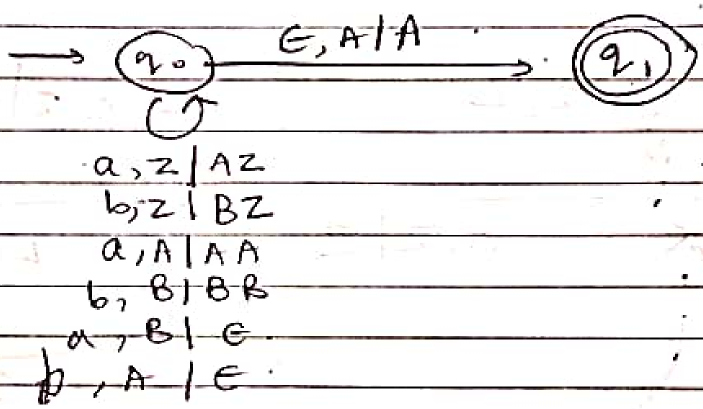
* Palindromes

not DPDA.

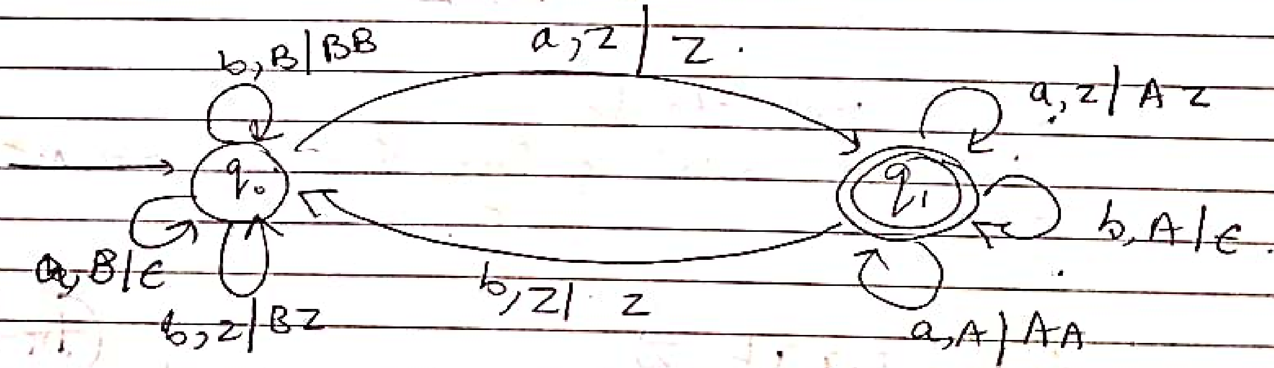
- There is no such equivalence to convert NPDA to DPDA
- Palindromes lang - NO DPDA.
- Only few languages can have both versions.

Q) $N_a(w) > N_b(w)$.

NPDA



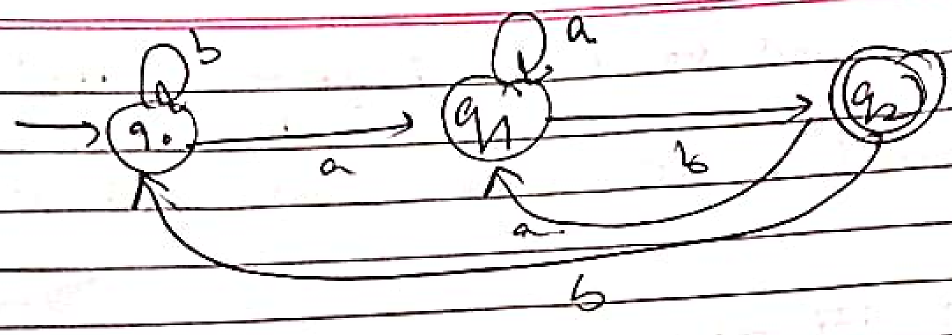
DPDA



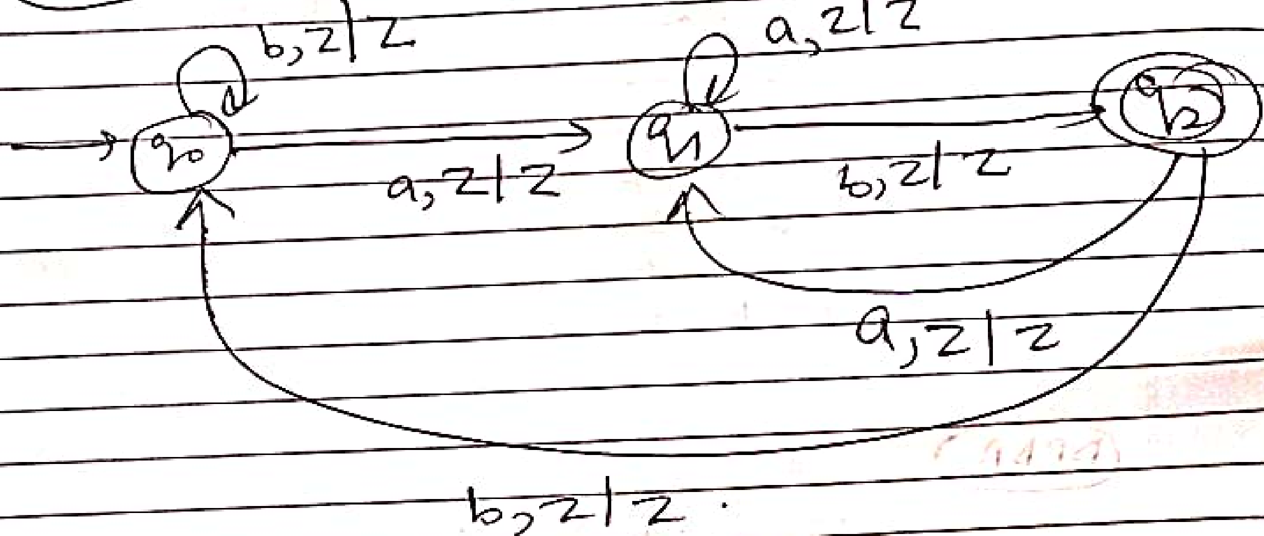
every reg lang = context free

↳ PDA possible

Q) $(a+b)^*ab$ (ends with ab)



(PDA)



don't use stack.
Process string by read move write
just scan & move.

(DPDA)

$\delta(q_0, aaab, z)$.
$\delta(q_1, aab, z)$
$\delta(q_1, ab, z)$
$\delta(q_1, b, z)$
$\delta(q_2, \epsilon, z)$ // final

every reg lang has DPDA! as it is from DFA.

$$\begin{array}{ccc} & CFL & \\ \diagup & & \diagdown \\ CFG & & PDA \end{array} \qquad \begin{array}{ccc} & RL & \\ \diagup & & \diagdown \\ RG & \longleftrightarrow & FA \end{array}$$

$$\begin{array}{ccc} & RG & \\ \diagup & & \diagdown \\ LLG & & RLG \end{array}$$

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

Req final: $M = (Q, \Sigma, \delta, T, q_0, Z, F)$.

1) Convert G to GNF.

2) $\delta(q_0, \epsilon, Z) = (q_1, SZ)$.

without processing any symbol, push S to stack.

$$\begin{array}{|c|}\hline S \\ \hline Z \\ \hline \end{array}$$

prodⁿ:
GNF: terminal followed be as many variables.
$A \to aBCD\ldots$

ex:

3) $\delta(q, a, A) = (q, BCD)$

$$\begin{array}{|c|}\hline B \\ \hline C \\ \hline D \\ \hline Z \\ \hline \end{array}$$

↳ $A \to a\alpha$:
(α=seq of var)

$\delta(q, a, A) = (q, \alpha)$ do this for all productions

4) after processing the entire string,
$\delta(q_1, \epsilon, Z) = (q_2, Z)$

δ = ② + all of ③ + ④.

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

$\Sigma = T$.

$\Gamma = V \cup \{Z\}$

Q) GNF

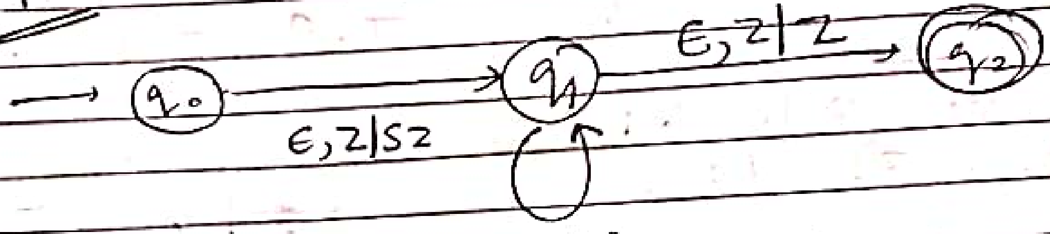
$S \to aB \mid bA$

$A \to a \mid aS \mid bAA$

$B \to b \mid bS \mid aBB$.

equivalent PDA

PDA



$a, S | B$

$b, S | A$

$a, A | \epsilon$

$a, A | S$

$b, A | AA$.

$b, B | \epsilon$

$b, B | S$

final state PDA

(stack modific^n = α) $a, B | BB$

total $1 + 8 + 1 = 10$ productions.

$\delta(q_0, \epsilon, Z) = (q_1, SZ)$

$\delta(q_1, \epsilon, A) = (q_1, \alpha)$

$A \to \alpha$
aBB.
aBa
whatever.

$\delta(q_1, a, a) = (q_1, \epsilon)$

$\delta(q_1, \epsilon, Z) = (q_2, Z)$

ε, S | aB

ε, S | bA

ε, S | a

ε, S | aS

ε, S | bAA

ε, B | b

ε, B | bS

ε, B | aBB

every CFG has equivalent PDA

ε baba

(q₀, baba, z)

(q₁, ε baba, Sz)

(q₁, baba, aBz)   (q₁, baba, bAz)

×

(q₁, aba, Az)

(q₁, aba, az)   (q₁, aba, aSz)   (q₁, aba, bAAz) classmate Date 26/12/2024 Page 56

PDA ⇒ CFG

empty stack PDA not final state PDA

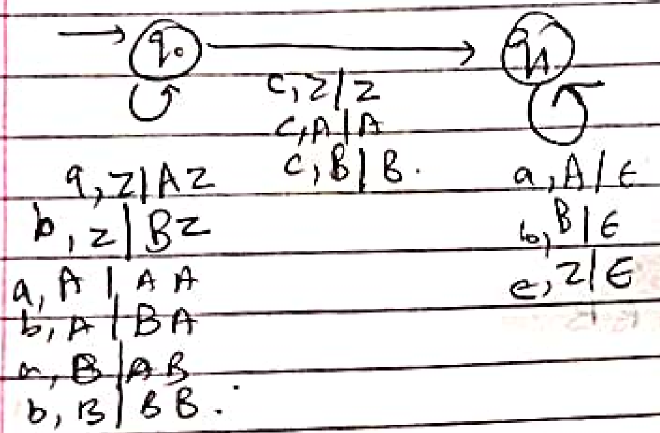
$M = (Q, \Sigma, \Gamma, \delta, q_0, Z, \phi)$

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

$T = \Sigma$

$Z \in \Gamma;\ p, q \in Q.$

V $Q : [q, Z, p] \cup \{S\}$ Z : all possible stack symbols
q, p : all possible states.



$S \rightarrow [q_0, Z, q_0] \mid [q_0, Z, q_1]$

$\delta(q_i, a, Z) = (q_j, AZ).$

$[q_i\ Z\ q_k] \rightarrow a[q_j\ A\ q_l][q_l\ Z\ q_k]$

$\delta(q_0, a, Z) = (q_0\ AZ)$

$l=0,\ k=0$: $(q_0\ Z\ q_0) \rightarrow a[q_0\ A\ q_0][q_0\ Z\ q_0]$

$l=0,\ k=1$: $[q_0\ Z\ q_1] \rightarrow a[q_0\ A\ q_0][q_0\ Z\ q_1]$

$l=1,\ k=0$: $[q_0\ Z\ q_0] \rightarrow a[q_0\ A\ q_1][q_1\ Z\ q_0]$

$l=1,\ k=1$: $[q_0\ Z\ q_1] \rightarrow a[q_0\ A\ q_1][q_1\ Z\ q_1]$

4 for each δ.

so $6 \times 4 = 24$ on the left.

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

$[q_0 z q_0] \to c\ [q_1 z q_0]$ }
$[q_0 z q_1] \to c\ [q_1 z q_1]$ }

$\delta(q_1, a, A) = (q_1, \epsilon)$

$[q_1 A q_1] \to a$

$[q_1 B q_1] \to b$.

$[q_1 z q_1] \to \epsilon$.

$\underset{(S)}{2} + 24 + 6 + 3 = 35$ productions.

simplify & remove useless productions/ variables.

$S \to [q_0, z, q_0]$ is useless as empty stack is not possible in q_0.

$S \to [q_0, z, q_1]$ is useful.

3 var, 2 states } $6 + 6 = 12$ var + S (starting).

$V: [q, z, p] \cup \{S\}$.

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

$L = \{a^n b^n c^n \mid n \geq 0\}$ not possible to write CFG or PDA.

$\therefore$ this language is not a context-free language.

$L_2 = \{w \mid w \in \{a, b, c\}^*\ N_a(w) = N_b(w) = N_c(w)\}$

not possible

concatenation

$L_3 = \{ww \mid w \in \{a,b\}^*\}$
CFG or PDA not possible.

Pumping Lemma
assume that the given lang is CFG ∴ G exists.
$z \in L$ $|z| = n$
$z = u\underline{vwx}y$

$|vx| \geq 1$
$|vwx| \leq n$ } conditions

uv^iwx^iy $\underbrace{i \geq 0}_{\text{pumping factor}}$
$\in L$.
$\notin L$: not CFL

1. Assume that it is CFG
2. Make it CNF. (Normalisation)
3. write parser tree - binary tree.
   leaf nodes - input symbols
   intermed nodes - variables.
   root - starting var.

30/12/2024
$z = uvwxy$.
* $|vx| \geq 1$
* $|vwx| \leq n$.

$|z| \geq n$
$uv^iwx^iy \in L$ $i \geq 0$.
if $\notin L \rightarrow$ not CFL.

Q) $L_1 = \{a^n b^n c^n \mid n \geq 0\}$

Assume L_1 is CFL.

$z = aabbcc$. [Take a str that belongs to L].

$|z| = n = 6$.

$u = a$, $v = a$, ~~bb~~ $w = bb$, $x = c$, $y = c$.

$\underline{a}\,\underline{a}\,\underline{bb}\,\underline{c}\,\underline{c}$
$u\ v\ w\ x\ y$

2 Conditions

$|vx| \geq 1 \rightarrow 2 > 1$. ✓

$|vwx| \leq n \rightarrow 4 < 6$ ✓

according to Lemma,

$uv^iwx^iy \in L$, $i \geq 0$..

$a(a)^i bb (c)^i c$

$i = 0$ $abbc \notin L$

$i = 2$ $aaabbccc \notin L$

Hence, the given language is not a context free language.

L_1 & L_2 are CFLs. $G_1 : L(G_1) = L_1$
$G_2 : L(G_2) = L_2$.

$G_1 = (V_1, T_1, P_1, S_1)$

$G_2 = (V_2, T_2, P_2, S_2)$

~~$L_1 \cup L$~~. $\boxed{L_1 \cdot L_2}$: $G = (V, T, P, S)$

↳ $V = V_1 \cup V_2 \cup \{S\}$

↳ $T = T_1 \cup T_2$

↳ $P = P_1 \cup P_2 \cup \{S \rightarrow S_1 S_2\}$

$S \Rightarrow S_1 S_2$ (derivⁿ) $\therefore L_1 \cdot L_2$ is also CFL

~~$S \Rightarrow S_1 S_2$~~

L_1^*

L_1 : CFL.

$G_1 : L(G_1) = L_1$

$G_1 = (V_1, T_1, P_1, S_1)$.

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

$\hookrightarrow V = V_1 \cup \{S\}$

$T = T_1$

$P = P_1 \cup \{S \rightarrow SS_1 \mid \epsilon\}$.

$\therefore L_1^*$ is also CFL.

$L_1 \cup L_2$

L_1 & L_2 are CFLs.

$G_1 \quad L(G_1) = G_1 \qquad (V_1, T_1, P_1, S_1)$

$G_2 \quad L(G_2) = G_2 . \qquad (V_2, T_2, P_2, S_2)$.

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

$\hookrightarrow V = V_1 \cup V_2 \cup \{S\}$

$T = T_1 \cup T_2$

$P = P_1 \cup P_2 \cup \{S \rightarrow S_1 \mid S_2\}$.

$\therefore L_1 \cup L_2$ is also CFL.

Hence CFLs are closed under $\cdot, *, \cup$.

Q) $L_1 = \{a^n b^n c^m \mid n, m \geq 0\}$

$S_1 \rightarrow AB$.

$A \rightarrow aAb \mid \epsilon$

$B \rightarrow cB \mid \epsilon$

$L_2 = \{a^n b^m c^m \mid n, m \geq 0\}$

$S_2 \rightarrow CD$

$C \rightarrow aC \mid \epsilon$

$D \rightarrow bDc \mid \epsilon$.

$L_1 \cap L_2 = \{a^i b^i c^i \mid i \geq 0\}$. But this is not CFL.

whenever $m = n$, intersection ~~exists~~.

CFLs are not closed under intersection.

$\overline{L_1 \cap L_2} = \overline{L_1} \cap \overline{L_2}$ DeMorgan's

↳ intersection will hold only if complementⁿ holds.

∴ CFLs are not closed under intersection, complementⁿ, difference.

* Decision trees
* Closure
* Pumping lemma.
* Write, simplify.
* PDA, DPDA.
* $G \to PDA$, $PDA \to G$.
* Show that it is ambiguous.

restricted turing machine

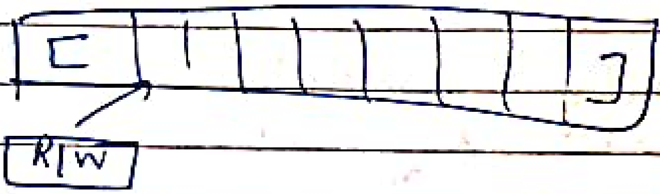
Linear Bounded Automata [LBA]

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

L, R – position of read/write tape end

$\delta : Q \times \Gamma \rightarrow Q \times \Gamma \times \{L, R\}$

$\delta(q_0, a) = (q_1, A, L)$



~~M = (Q,~~ $\delta(q_0, a) = (q_1, A, L)$
$($ $R)$

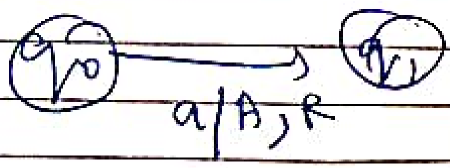
$A = \{a^n b^n \mid n \geq 0\}$

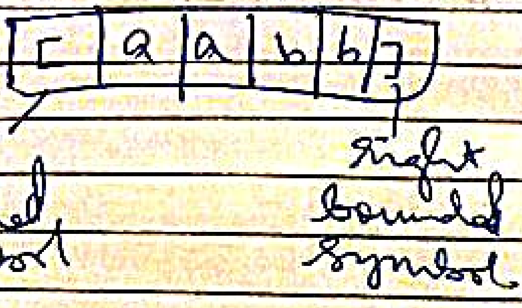
$(q_0, [w]) \vdash (q_1, A, R)$

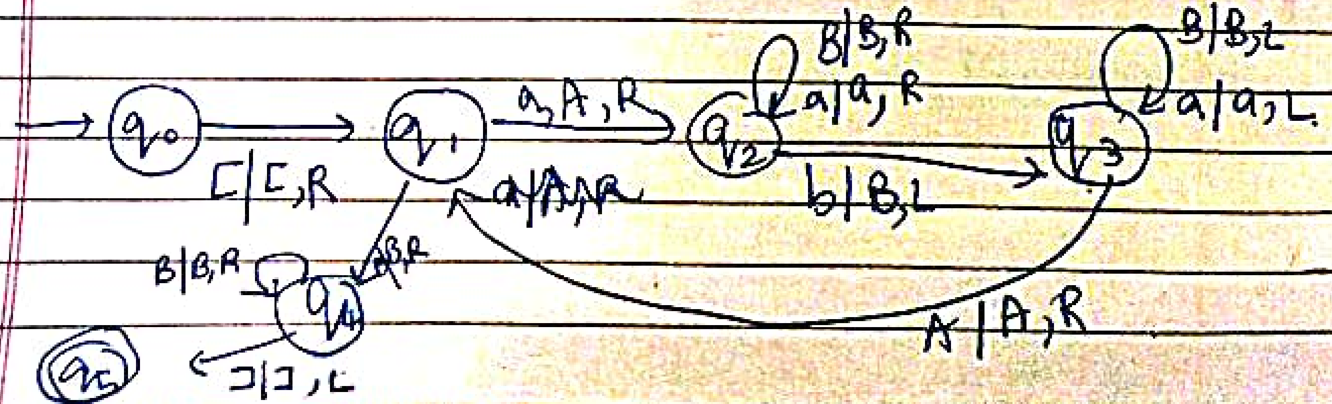
$(q_0, w) \vdash^* (q_f, [AABBCC])$ — [aabbcc], [Aabbcc]

$L(M) = \{ w \mid w \in \Sigma^* (q_0)[w]] \vdash (q_f, F^* \}$

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







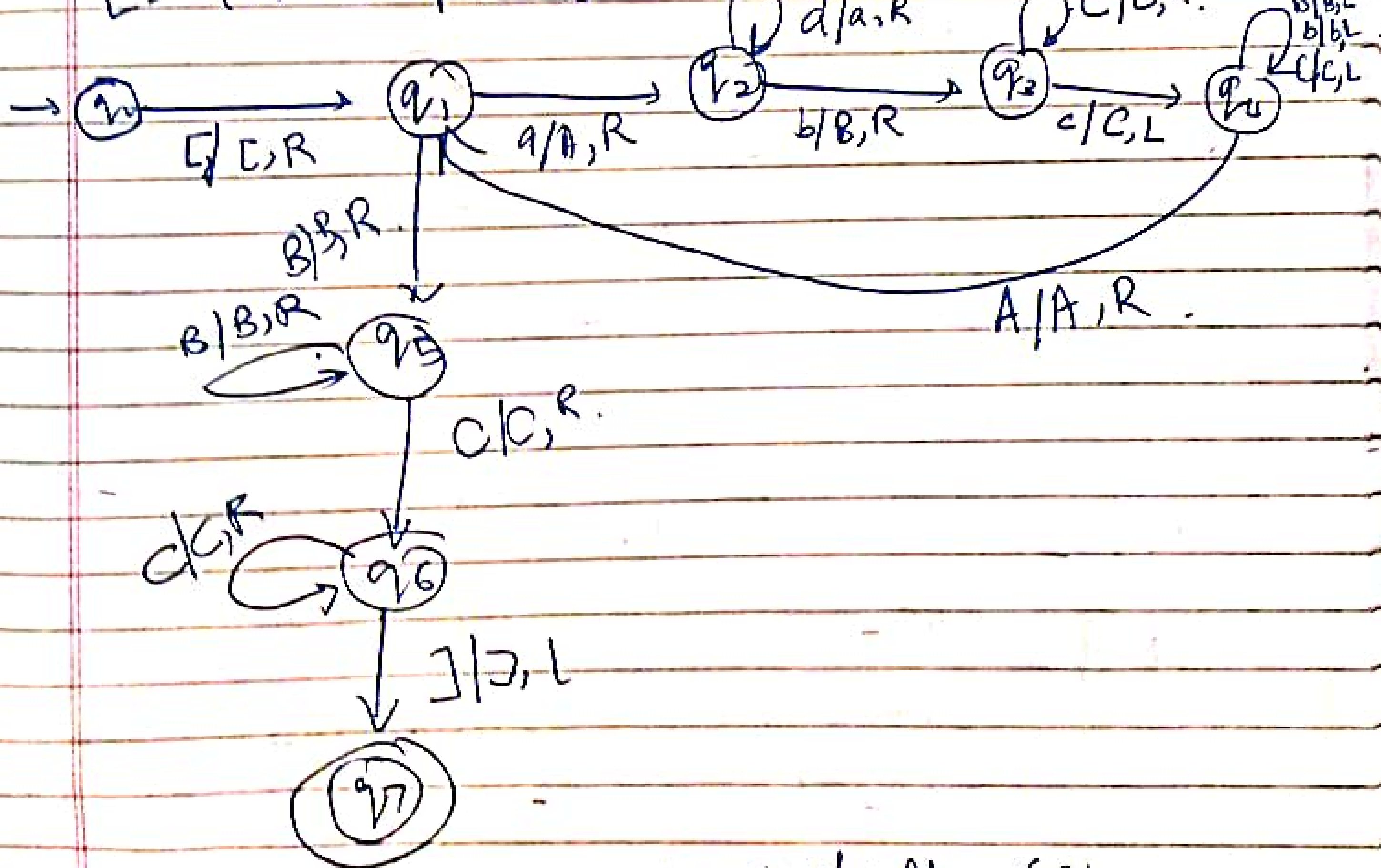
use underscore to mark R|W pointer
move L or R accordingly

start id.

$(q_0, \underline{[}aabb]) \vdash (q_1, [\underline{a}abb]) \vdash (q_2, [A\underline{a}bb])$
$\vdash (q_2, [Aa\underline{b}b]) \vdash (q_3, [A\underline{a}Bb]) \vdash (q_3, [\underline{A}aBb])$
$\vdash (q_1, [A\underline{a}Bb]) \vdash (q_2, [AA\underline{B}b]) \vdash (q_2, [AAB\underline{b}])$
$\vdash (q_3, [AA\underline{B}B]) \vdash (q_3, [A\underline{A}BB]) \vdash (q_1, [AA\underline{B}B])$
$\vdash (q_4, [AABB\underline{]}) \vdash (q_4, [AABB\underline{]}) \vdash (q_5, [AAB\underline{B}])$

last id.

$L = \{a^n b^n c^n \mid n \geq 0\}$.



Lang of LBA = CSL
Context Sensitive Lang.

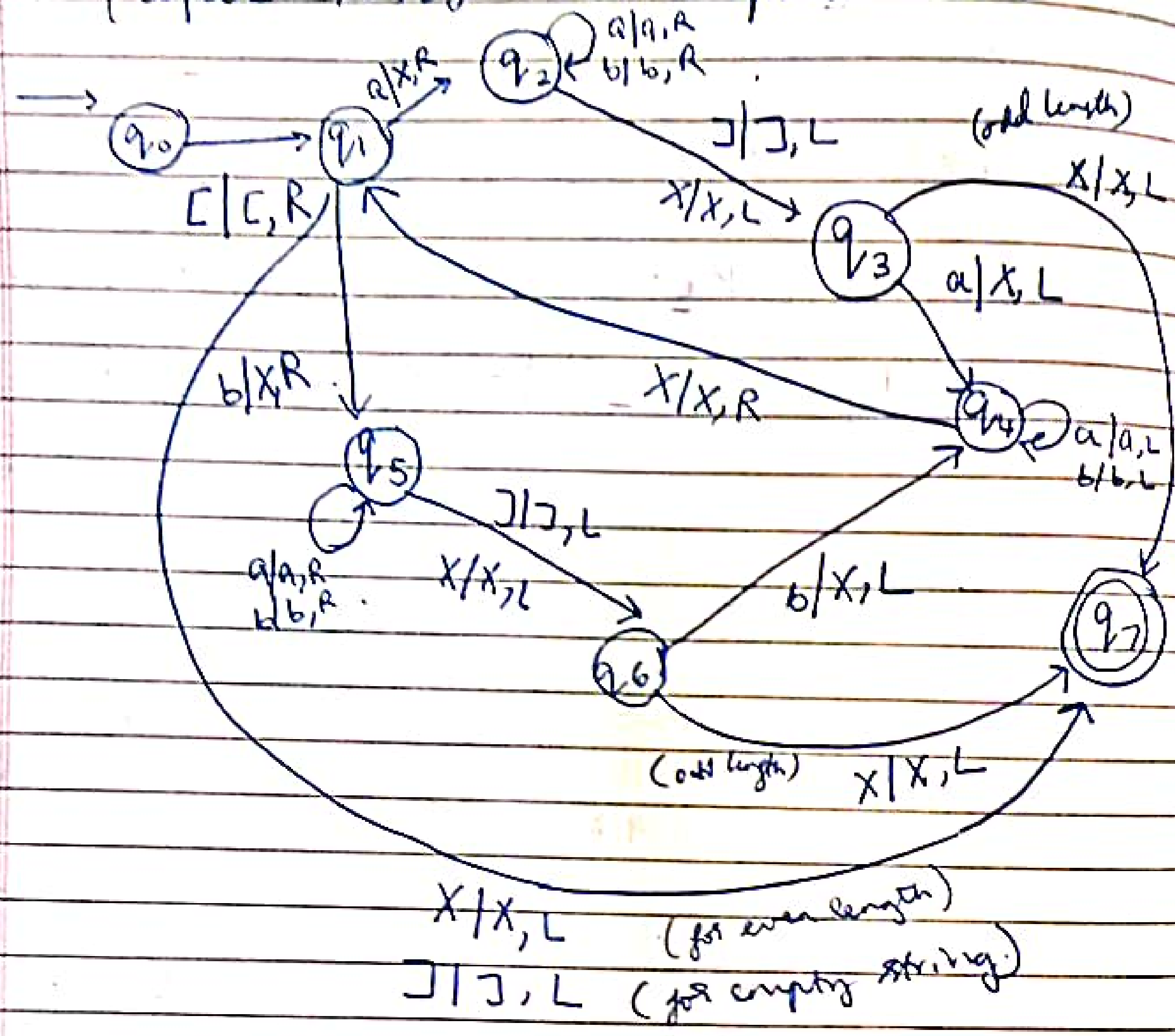
Grammar = CSG
Context Sensitive Grammar

$a^n b^n c^n$ CFG or PDA is not possible

[ababa]   XXaXX
[Xbab(a)]   XXXXX
(X)bab X

classmate Date 3/1/2023 Page 64

8) $L = \{w \mid w \in \{a,b\}^*$ w is a palindrome$\}$



W = ababa

$(q_0, [ababa]) \vdash (q_1, [\underline{a}baba]) \vdash (q_2, [X\underline{b}aba])$

$\vdash (q_2, [Xb\underline{a}ba]) \vdash (q_2, [Xba\underline{b}a]) \vdash (q_2, [Xbab\underline{a}])$

$\vdash (q_2, [Xbaba\underline{]}) \vdash (q_3, [Xbab\underline{a}]) \vdash (q_4, [Xba\underline{b}X])$

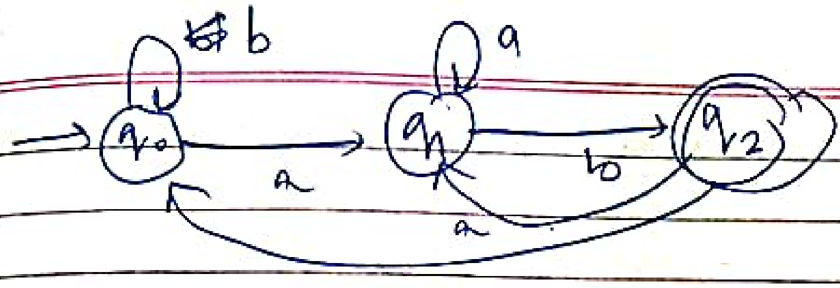
$\vdash (q_4, [Xb\underline{a}bX]) \vdash (q_4, [X\underline{b}abX]) \vdash (q_4, [\underline{X}babX])$

$\vdash (q_1, [X\underline{b}abX]) \vdash (q_5, [XX\underline{a}bX]) \vdash (q_5, [XXa\underline{b}X])$

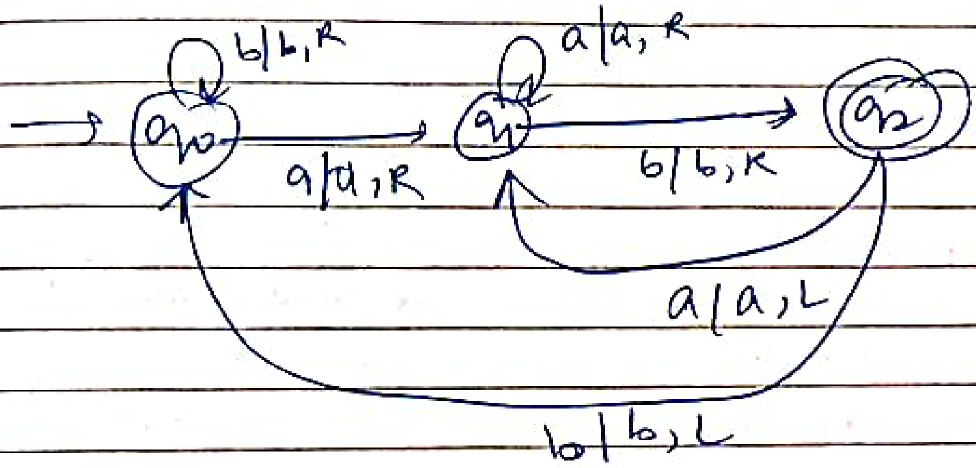
$\vdash (q_5, [XXabX]) \vdash (q_6, [XXab\underline{X}]) \vdash (q_4, [XXaXX])$

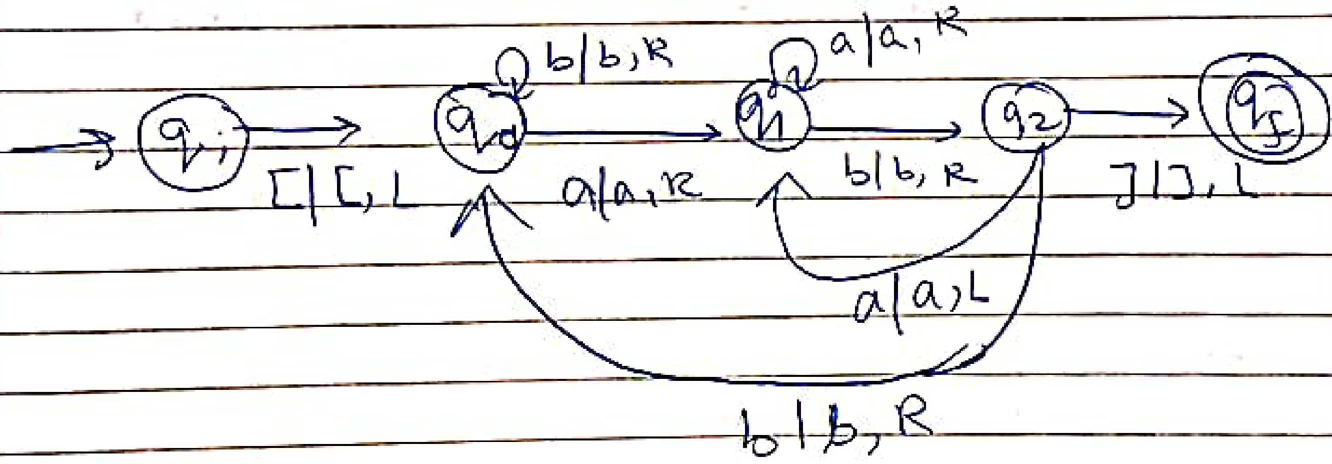
$\vdash (q_4, [X\underline{X}aXX]) \vdash (q_1, [XX\underline{a}XX]) \vdash (q_2, [XXX\underline{X}X])$

$\vdash (q_3, [XX\underline{X}XX]) \vdash (q_7, [X\underline{X}XXX])$



str ending with ab.





↳ every reg lang → context sensitive

but every CS ✗→ reg lang

ex: palindromes

no DFA & PDA

but LBA exists.

RL < FA, RG

CFL < PDA, CFG

CSL < LBA, CSG

* every Reg grammar (left linear or right linear) is CFG

* every CFG is not reg grammer because reg grammer RHS can have max 1 variable

Context Sensitive Grammar

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

$P: \alpha \to \beta \quad ; \quad \alpha, \beta \in (V \cup T)^*, \quad |\alpha| \le |\beta|$

α must contain atleast one var.

valid: $aA \to aa$, $aBb \to aba$, $AB \to BA$

not valid: $aBb \to ab$ $(|\alpha| > |\beta|)$, $ab \to ba$

every CFG can be represented as CSG but every CSG cannot be represented as CFG.

Q) $L = \{a^n b^n c^n \mid n \ge 0\}$

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

$V = \{S, A, B, C\}$

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

S: start symbol.

~~$S \to ABCS \mid ABC$~~
~~$BA \to AB$~~
~~$CB \to BC$~~
~~$CA \to AC$~~
~~$A \to a$~~
~~$aA \to AA$~~
~~$bB \to bb$~~
~~$cC \to cc$~~

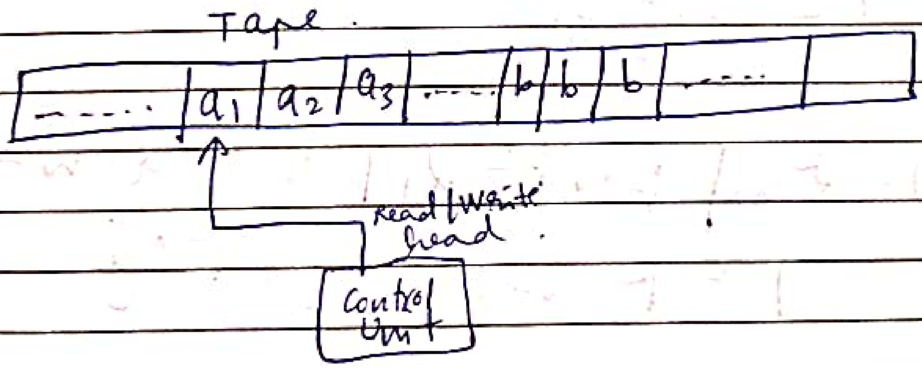
$P = \{$
$S \to ABCS \mid ABC$
$BA \to AB$
$CB \to BC$
$CA \to AC$
$A \to a$
$aA \to aa$
$aB \to ab$
$bB \to bb$
$bC \to bc$
$cC \to cc \}$

$S \Rightarrow ABCS$
$\Rightarrow ABCABC$
$\Rightarrow ABACBC$
$\Rightarrow ABABCC$
$\Rightarrow AABBCC.$
$\Rightarrow aABBCC$
$\Rightarrow aaBBCC$
$\Rightarrow aabBCC$
$\Rightarrow aabbCC$
$\Rightarrow aabbcC$
$\Rightarrow \boxed{aabbcc}$

10/1/2025 Turing Machines

TM – modified version of PDA.
instead of stack like in PDA, use tape.

TM can ~~detect~~ recognize : 1) Reg Lang (Reg Grammar)
2) CF L (CFG)
3) CSL (CSG)
4) Unrestricted grammar lang



String to be scanned should end with blanks.
Various actions performed by machine:
i) change state from one state to another.
ii) Symbol pointing to by R/W head can be replaced by another symbol
iii) The R/W head may move either towards left or right.

Δ = Blank

If there is no entry in table for current comb^n of symbol & state, machine will halt.

$M = (Q, \Sigma, \Gamma, \delta, q_0, \cancel{B}, F)$

- Q: set of finite states
- Σ: set of i/p alphabets
- Γ: set of tape symbols
- q_0: start state
- $F \subseteq Q$ is a set of final states.

δ is transition f^n from $Q \times \Gamma \rightarrow Q \times \Gamma \times \{L, R, S\}$

S = Stay
L = left
R = Right

~~TM can do one of the following:~~

1. Halt and accept by entering into final state
2. Halt and reject. This is possible if the transition is not defined
3. ~~TM will never halt and enters into an infinitely loop.~~

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

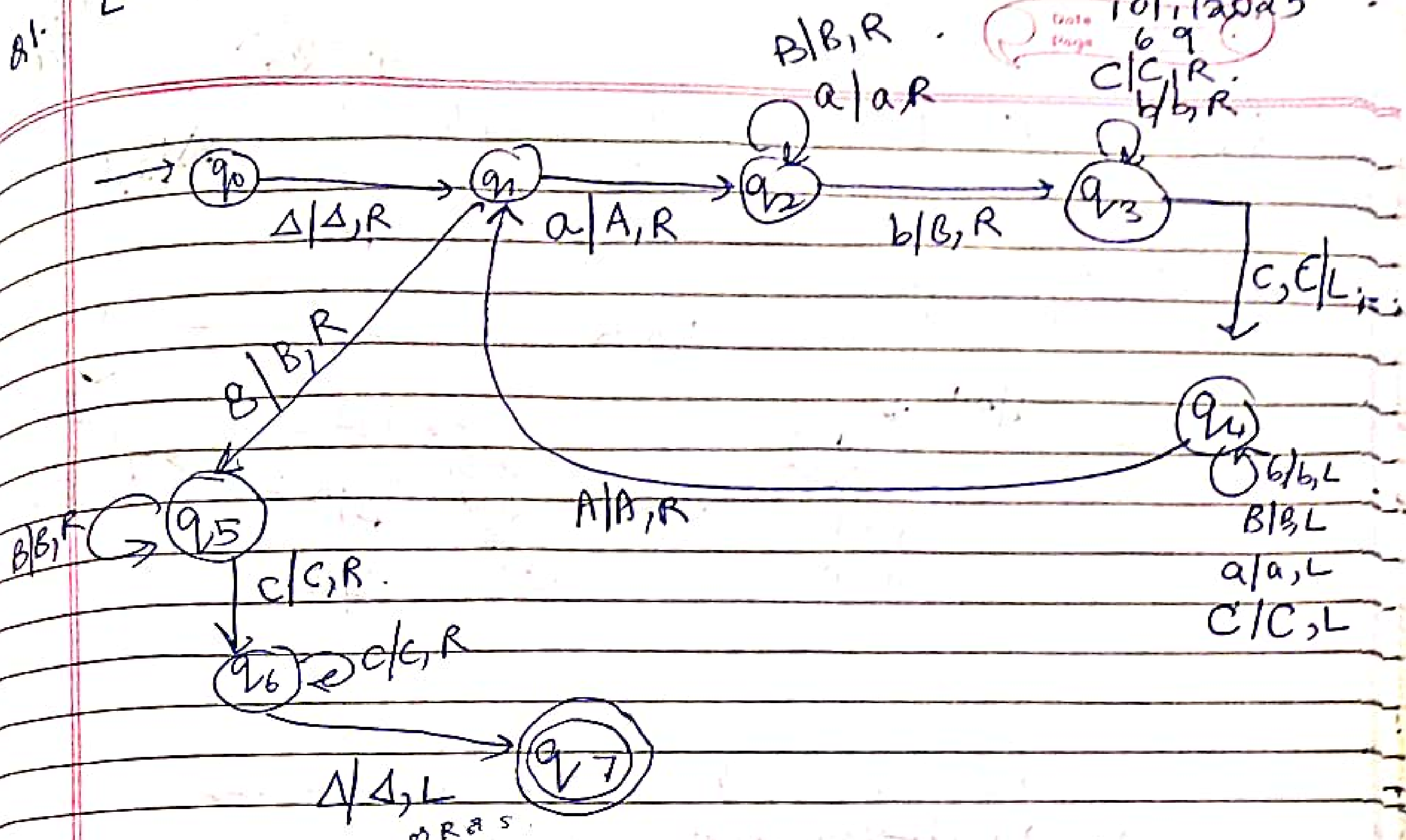
L(M) = Recursively enumerable Language or RE language

$L(M) = \{ w \mid q_0 w \vdash^* \alpha_1 p \alpha_2$ where $w \in \Sigma^*$, $p \in F$ and $\alpha_1, \alpha_2 \in \Gamma^* \}$

Q1. $L = \{a^n b^n c^n \mid n \geq 0\}$

10/1/2025

69



W = ΔaabbccΔ.

$(q_0, \underline{\Delta}aabbcc\Delta) \vdash (q_1, \Delta\underline{a}abbcc) \vdash$ - - - - -

~~⊢ (q7, ΔAABBCCΔ)~~

$\vdash (q_7, \Delta AABBC\underline{C}\Delta)$

ΔaabbccΔ

ΔAabbccΔ

ΔAaBbccΔ

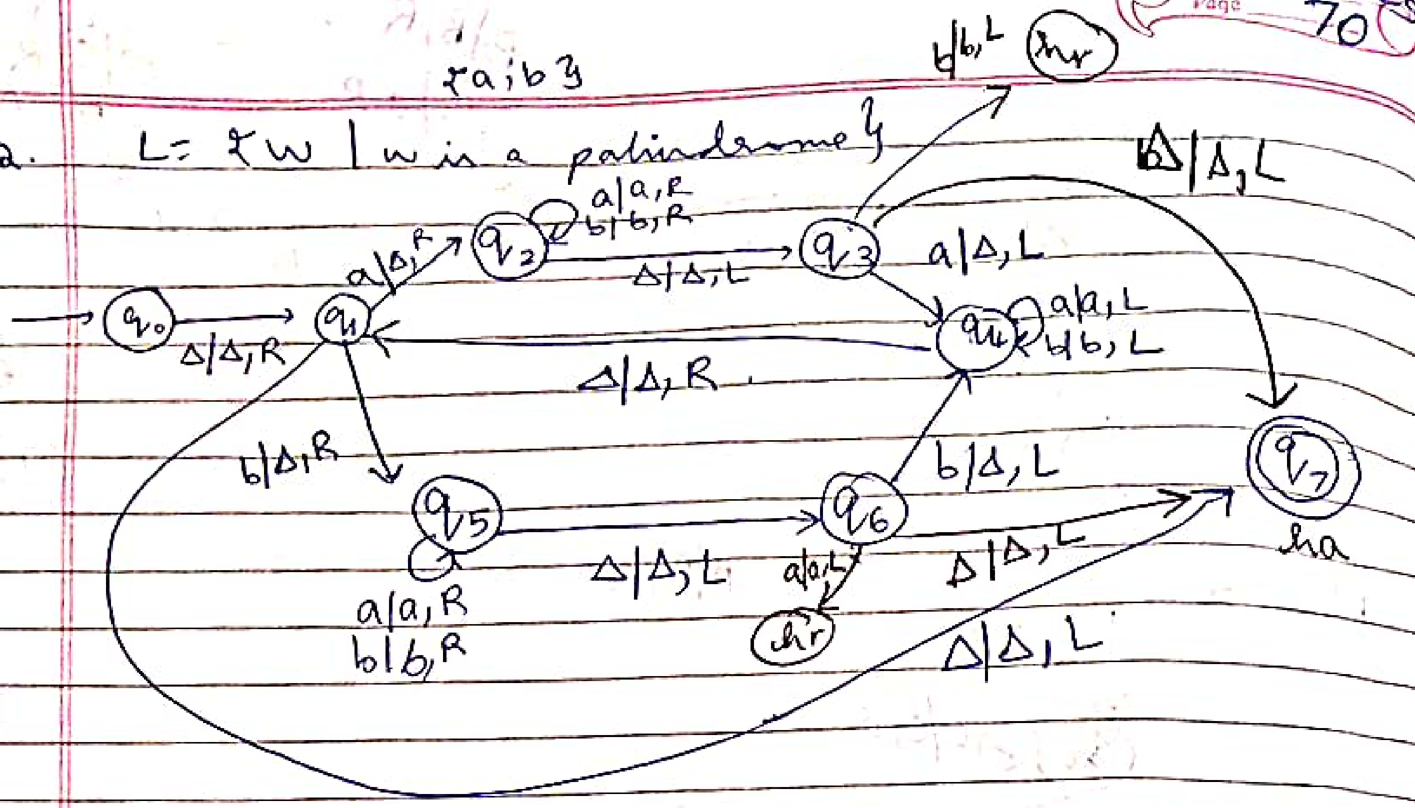
ΔAaBbCcΔ

ΔAABbCcΔ

⋮

ΔAABBCCΔ

Q2. $L = \{w \mid w \text{ is a palindrome}\}$ $\{a, b\}$



$q_3 \rightarrow q_7$, $q_6 \rightarrow q_7$ } odd length palindromes
$q_1 \rightarrow q_7$ } even length palindromes

final state: (q_7, Δ)

$(q_0, \underline{\Delta}abba\Delta) \vdash (q_1, \Delta\underline{a}bba\Delta) \vdash (q_2, \Delta bba\Delta)$
$\vdash (q_2, \Delta b\underline{b}a\Delta) \vdash (q_2, \Delta bb\underline{a}\Delta) \vdash (q_2, \Delta bba\underline{\Delta})$
$\vdash (q_3, \Delta bb\underline{a}\Delta) \vdash (q_4, \Delta b\underline{b}\Delta) \vdash (q_4, \Delta \underline{b}b\Delta)$
$\vdash (q_4, \underline{\Delta}bb\Delta) \vdash (q_1, \Delta\underline{b}b\Delta) \vdash (q_5, \Delta\underline{b}\Delta)$
$\vdash (q_5, \Delta b\underline{\Delta}) \vdash (q_6, \Delta\underline{b}\Delta) \vdash (q_4, \underline{\Delta}\Delta\Delta)$
$\vdash (q_1, \Delta\underline{\Delta}\Delta) \vdash (q_7, \Delta)$

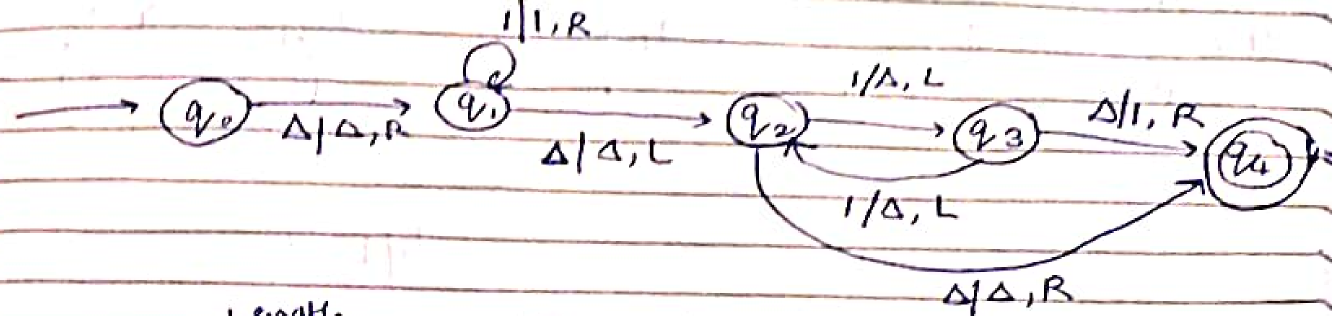
ha = halt and accept
hr = halt and reject

w: 1111   4 % 2 = 0
w = 111   3 % 2 = 1

tells whether input string is odd or even

Date 13|1|2025
Page 71

Q) a) $w \in \{1\}^*$

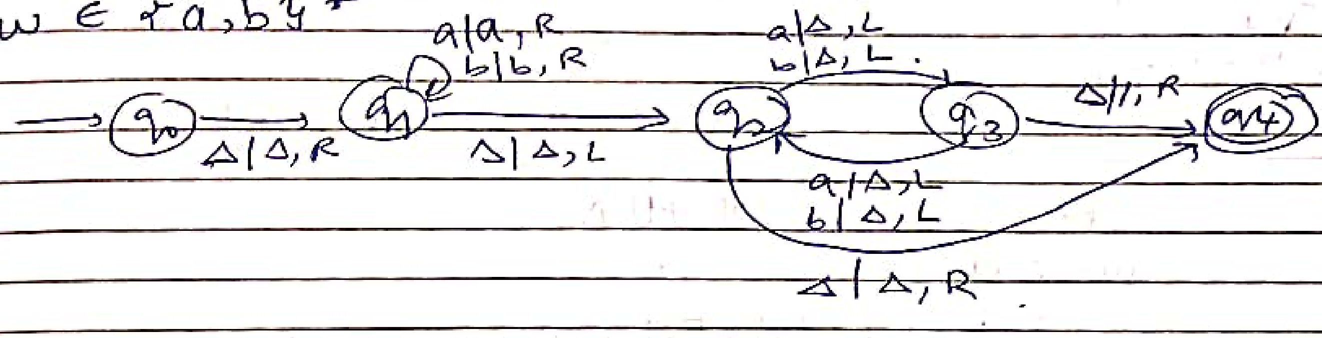


Length

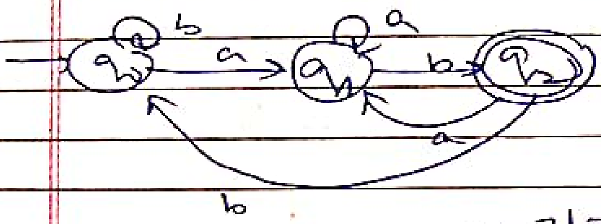
Result: if odd → on the tape - only 1 cell has 1, other rest is blank
if even → on the tape, only blanks.

The string w has only one input symbol = $\{1\}^*$ - Unary

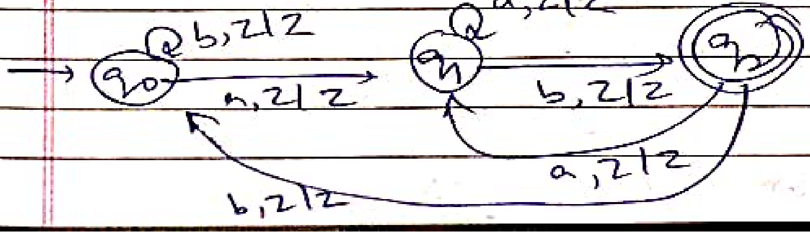
b) $w \in \{a, b\}^*$



Ends with ab



DFA   Reg lang
every reg lang has finite automata

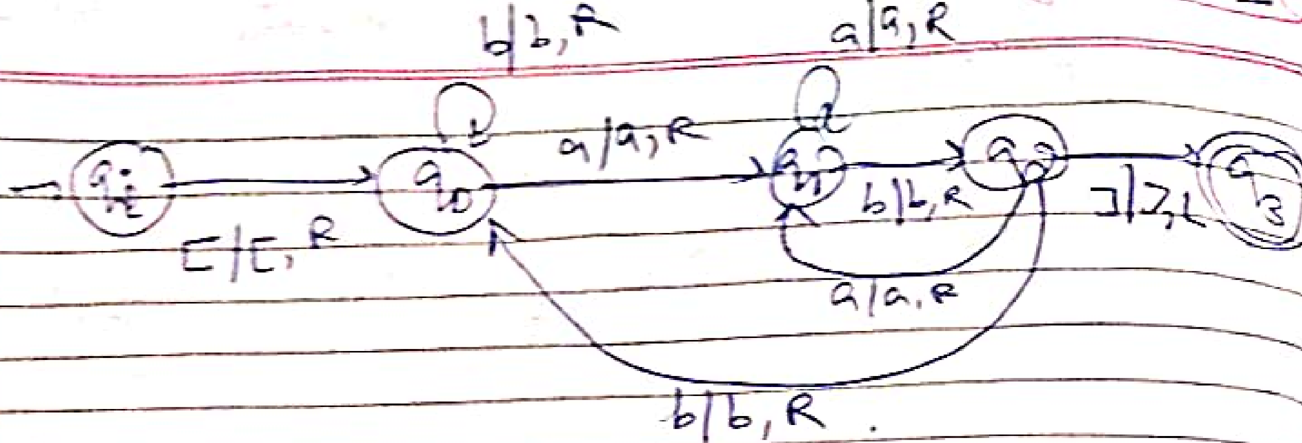


PDA   CFG
every reg lang has CFG

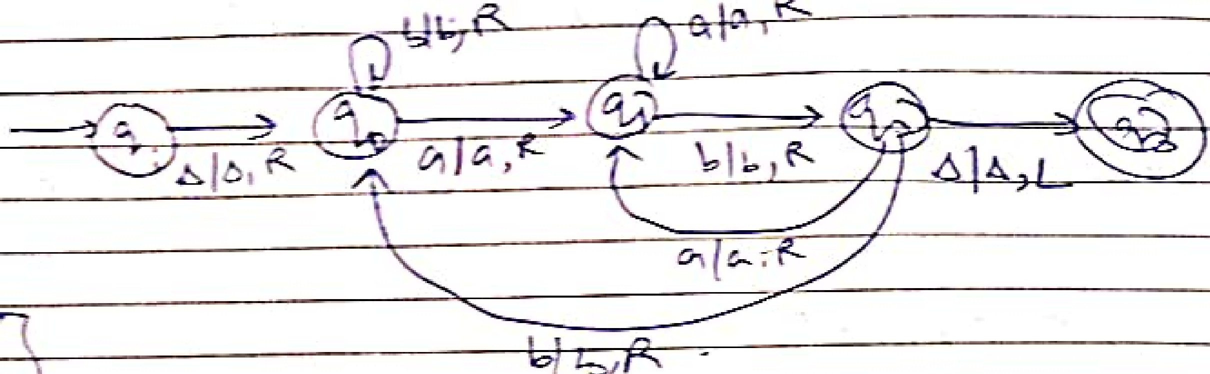
CSL

~~se~~

LBA



every reg lang has CSL



REN

Turing Machine

every reg lang has REN

COPY STRING OPERATION

w = {aa, bb, ---- }

Δ	a	b	a	b	b	Δ	a	b	a	b	b	Δ

M1 read first a → A then process till Δ next also will be Δ replace by a then come back left. when you get A go right. b → B go right till we get a or b after Δ then replace Δ by b And so on.

after processing fully, on left journey if you get A & B, copy can be stopped and replace A by a, B by b.

↳ copy is on the right of w

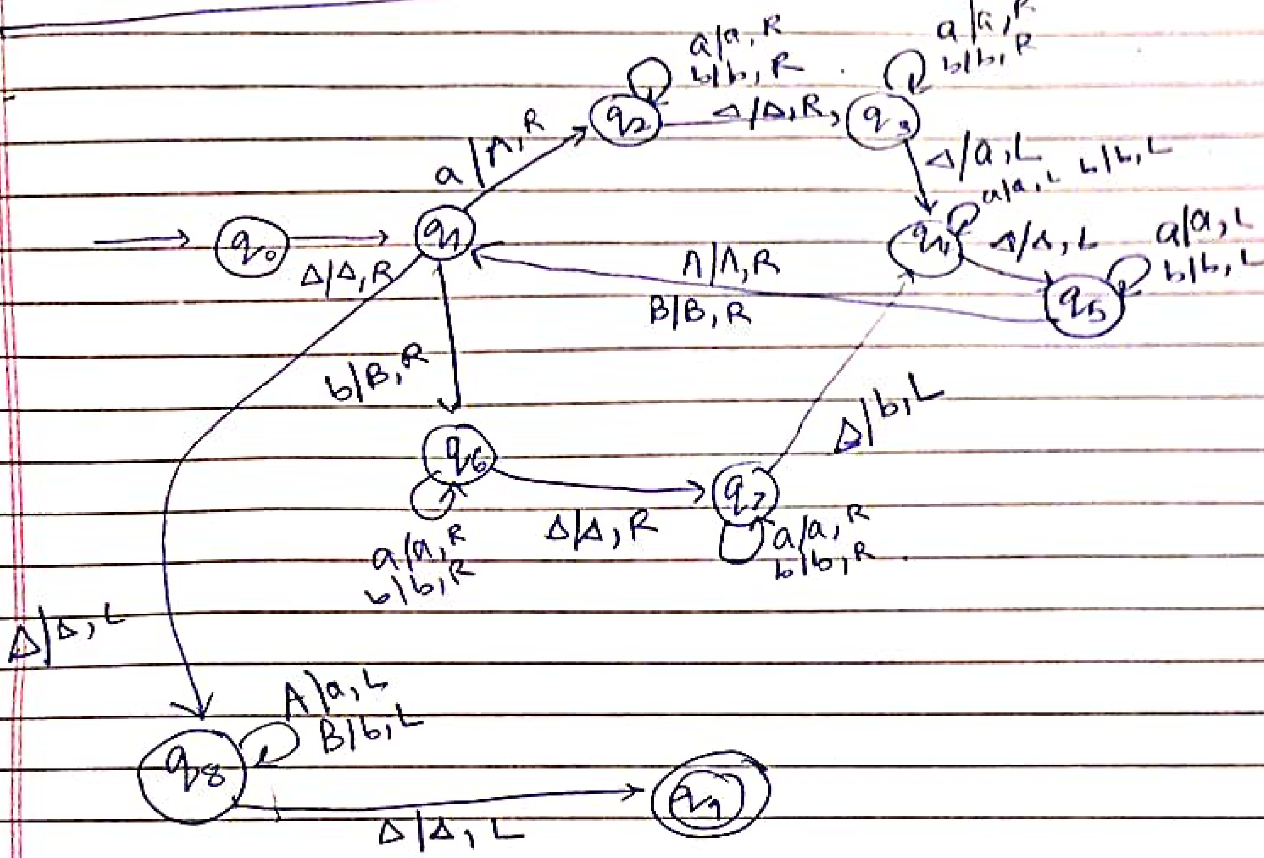
(M2) COPY can also be on the left of w

Δ a a b b a Δ

start from Right end

then copy

(M1)



if $w \in \{a,b,c\}^*$,
add another branch from q_1
for c
and add transitions for c along
with a, b everywhere.

~~ww^R~~ $\boxed{ww^R}$

Δ a b a b Δ b a b a Δ.

copying reverse of w to its right.

$\Delta W_1 \# W_2 \Delta \longrightarrow \Delta W_1 W_2 \# \Delta$ Concatenation

traverse to # i.e., finish W_1 traversal then read what is after #. if a (replace as #) then go left replace # as a and then continue

20/1/2025

Type 3	RL	FA	RG
Type 2	CFL	PDA	CFG
Type 1	CSL	LBA	CSG
Type 0	REL	TM	URG

✲ Write a note on Chomsky Hierarchy.

RG: $V \rightarrow (V \cup T)^*$, but max 1 var on right hand side

only 1 var

CFG: $V \rightarrow (V \cup T)^*$.

CSG: $\alpha \rightarrow \beta$ $\quad \alpha, \beta \in (V \cup T)^*, \ |\alpha| \leq |\beta|$

URG: $\alpha \rightarrow \beta$ $\quad \alpha, \beta \in (V \cup T)^*$

CSG $L = \{a^n b^n c^n \mid n \geq 0\}$ $\quad G = (V, T, P, S)$ — Start symbol

$\{S, A, B, C\}$ $\quad \{a, b, c\}$

$P = \{ S \rightarrow ABCS \mid ABC$

$BA \rightarrow AB$ $\quad CB \rightarrow BC$ $\quad \cancel{AC \rightarrow CA}$ $\quad CA \rightarrow AC$

$A \rightarrow a$ $\quad aA \rightarrow aa$ $\quad aB \rightarrow ab$ $\quad bB \rightarrow bb$ $\quad bC \rightarrow bC$

$cC \rightarrow cc \}$

UAG

$S' \rightarrow FS$

$S \rightarrow ABCS \mid ABC$

$BA \rightarrow AB$, $CA \rightarrow AC$, $CB \rightarrow BC$

$\boxed{FA \rightarrow a}$ $(|\alpha| > |\beta|)$

$aA \rightarrow aa$

$aB \rightarrow ab$

$bB \rightarrow bb$

$bC \rightarrow bc$

$cC \rightarrow cc$

$S' \Rightarrow F\underline{S}$
$\Rightarrow FABCS$
$\Rightarrow FABCABCS$
$\Rightarrow FAB\underline{A}CBC$
$\Rightarrow FAABC\underline{CB}C$
$\Rightarrow \underline{FA}ABBCC$
$\Rightarrow \underline{aA}BBCC$
$\Rightarrow aa\underline{B}BCC$
$\Rightarrow aab\underline{B}CC$
$\Rightarrow aabbcC$
$\Rightarrow aabbc\underline{C}$
$\Rightarrow \boxed{aabbcc}$

Q) $N_a(w) = N_b(w) = N_c(w)$

$S' \rightarrow FS$

$S \rightarrow ABCS \mid ABC$

$BA \rightarrow AB$, $CA \rightarrow AC$, $CB \rightarrow BC$ $\quad FB \rightarrow b$, $FC \rightarrow c$

$\boxed{AB \rightarrow BA,\ AC \rightarrow CA,\ BC \rightarrow CB}$

$FA \rightarrow a$

$aA \rightarrow aa$

$aB \rightarrow ab$

$bB \rightarrow bb$

$bC \rightarrow bc$

$cC \rightarrow cc$

$bA \rightarrow ba$

$cA \rightarrow ca$

$aC \rightarrow ac$

REL: $\cup$, $\cap$, complement, difference

Q) Construct URG for given lang.
Q) Identify type of grammar
Q) Which language is the given grammar for?

Undecidable Problems

Post Correspondence Problem [PCP]

2 lists. $|A| = |B|$
A, B. (both have same no. of strings)

A : $x_1, x_2, \ldots, x_n$
B : $y_1, y_2, \ldots, y_n$.

$x, y \in \Sigma^*$ $\qquad \Sigma = \{0, 1\}$

There exists a Post Correspondence Solution of pair (A,B) if there is a non empty sequence of integers $i, j, \ldots, k$, such that.

can have repetitions but use every string atleast once

$x_i x_j \ldots x_k = y_i y_j \ldots y_k$.
(concatenation)

PCP: Devise an algorithm that will tell us, for any (A,B) whether or not there exists a PCP solution.

Q)

	List 1	List 2
1	$x_1 = 1$	$y_1 = 111$
2	$x_2 = 10111$	$y_2 = 10$
3	$x_3 = 10$	$y_3 = 0$

Trial & Error

Matching options: ①, ② not ③

start at the bottom $x_3 = 10$, $y_3 = 0$.
no match. so ~~pattern~~ don't start with 3.
$x_2 = 10111$, $y_2 = 10$ ✓

classmate
Date 23 | 2025
Page 77

List 1 → 10111
List 2 → 10.

Now we need to start with 1 : 1 or 3.
but we need 1 after this 1. So chose 1.

List 1 → 101111 .
List 2 → 101111 .
we need 1 : 1 or 2 : Choose 1 .

List 1 → 1011111
List 2 → 10111111
We need 1 : choose 3 .

List 1 → 10111110 .
List 2 → 10111110 .

2 1 1 3 .

Q) CSG
Q) URG
Q) PCP .
Q) Chomsky Hierarchy . short note
Q) Prop of REL .
Q) LBA Design
Q) TM Design
Q) TM Lang accept , given fn