design → automata

~~comp~~ implement → ~~ten~~ compiler

DATE: 3/1/17
PAGE NO.:

Book:

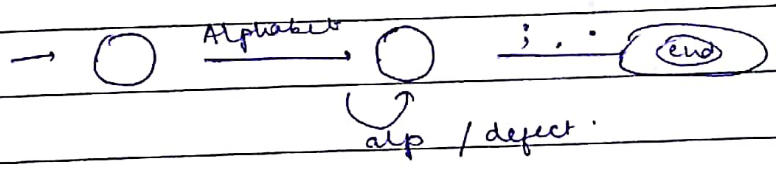
Formal language and Automata Theory — Peter Linz

Theory of Computation

→ System, while converting HLL to binary, first checks the rules & then converts.

→ Theoretical model of how compiler converts ~~to~~ HLL to LLL is done by Automata

→ Representing in form of diagram is a type of automata



Automata : tool which fiinds whether given i/p is correct or not

(1) Finite Automata : have finite no. of states (have no memory, processing directly)

(2) Push Down Automata (FA + stack) → memory in form of stack

(3) Linear Bounded Automata (FA + Finite tape) → can process any element

(4) Turing Machine (FA + Infinite tape)
↓
can process any type of data.

→ One ~~not p~~ which can't be processed by Turing Machine:

Uncomputable / Undecidable.

→ State remembers on which step I am

DATE: / /
PAGE NO.:

Finite Automata

It is defined as 5 tuple $Q, \Sigma, \delta, q_0, F$

Q → set of states
q_0 → initial state
F → set of final states
δ → Transition func^n
Σ → Input symbol

Finite Automata (FA)

Deterministic (DFA) | Non-Deterministic (NFA)

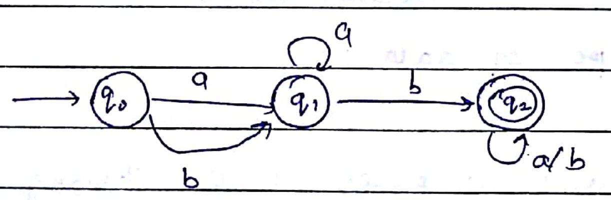
→ are the automata which have move defined for every symbol.
→ has only 1 move / symbol
→ moves not defined are considered as dead moves

DFA

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

see state. see func^n → generate new state

Transition diagram



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

Transition table

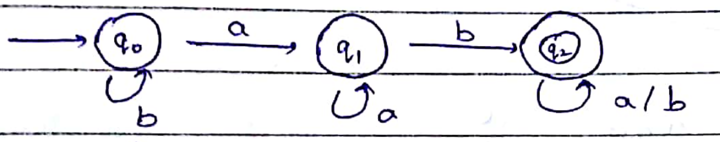
	a	b
I.S. → q_0	q_1	q_1
q_1	q_1	q_2
q_2 f.s.	q_2	q_2

Teacher's Signature

Transition func^n :

$\delta(q_0, a) \longrightarrow q_1$
$\delta(q_0, b) \longrightarrow q_1$
$\delta(q_1, a) \longmapsto q_1$
$\delta(q_1, b) \longmapsto q_2$
$\delta(q_2, a) \longrightarrow q_2$
$\delta(q_2, b) \longrightarrow q_2$

- Design a finite automata over $\{a, b\}$ that accept string containing ab as substring.



: Tool / Flow chart

need one move for every symbol

→ Now, have to check whether correct or not

Acceptance by final state : acceptable ~~by~~ only if it reaches to Final state

↳ baa

$(q_0, baa) \vdash (q_0, aa) \vdash (q_1, a) \vdash (q_1, \wedge)$

$\downarrow$
Not on F.S.
$\Downarrow$
not acceptable

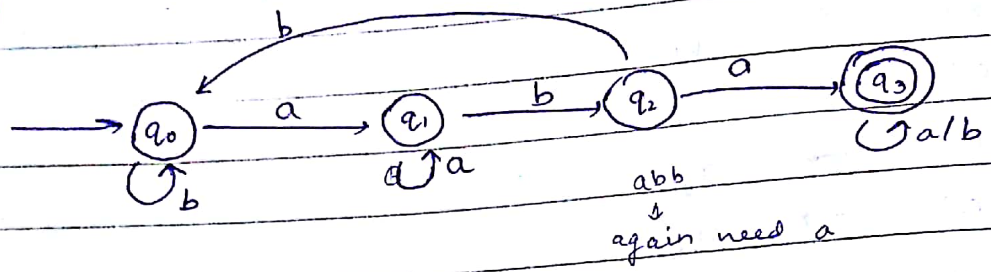
↳ ab ab a

$(q_0, ababa) \vdash (q_1, baba) \vdash (q_2, aba)$
$\vdash (q_2, ba) \vdash (q_2, a) \vdash (q_2, \wedge)$

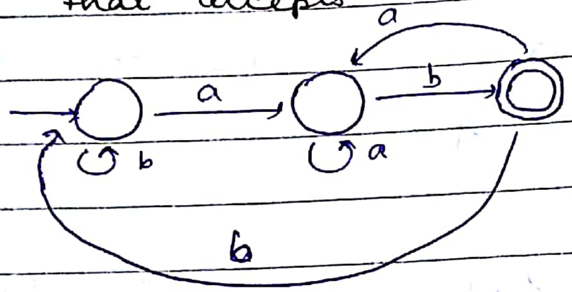
$\Downarrow$
on F.S. ⇒ acceptable

In DFA : only 1 automata is possible

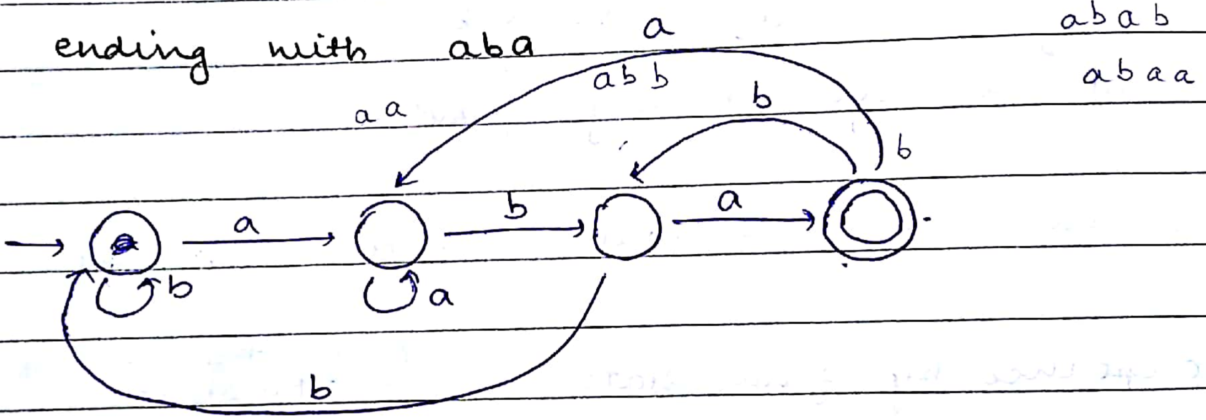
ii.) contains aba as substring.



Q i.) FA that accepts all string ending with ab.

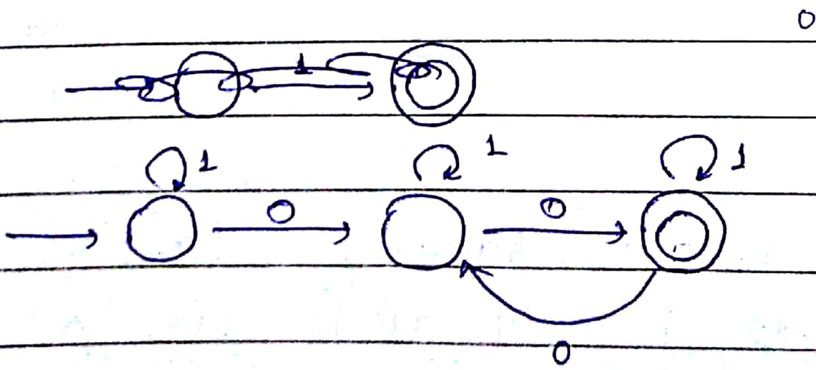


ii.) ending with aba



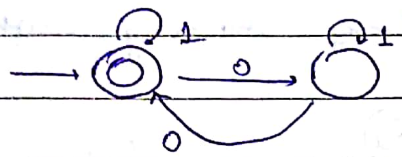
Q. $\Sigma = \{0, 1\}$ (no of $0 \geq 1$)

accept all string containing 0 in even no.



Ⓑ→ whenever accepting string of 0 length, initial state is always final state

if 101 ≥ 0



→ i/p : a , b

all strings containing

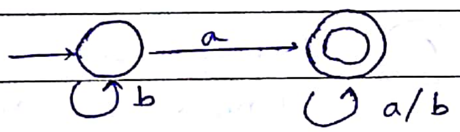
~~exactly 1~~ a

1.) atleast 1 a.

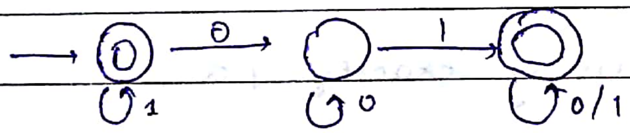
2.) 0 followed by 1.

3.) odd no. of 1's

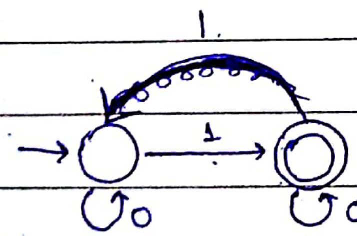
1.)



2.) 0-1



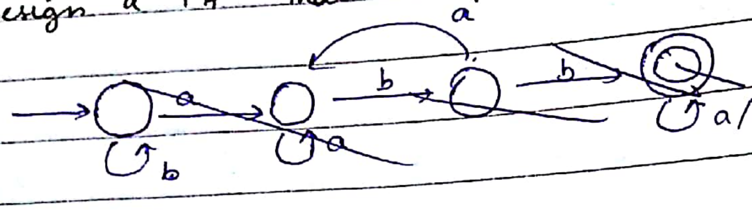
3)



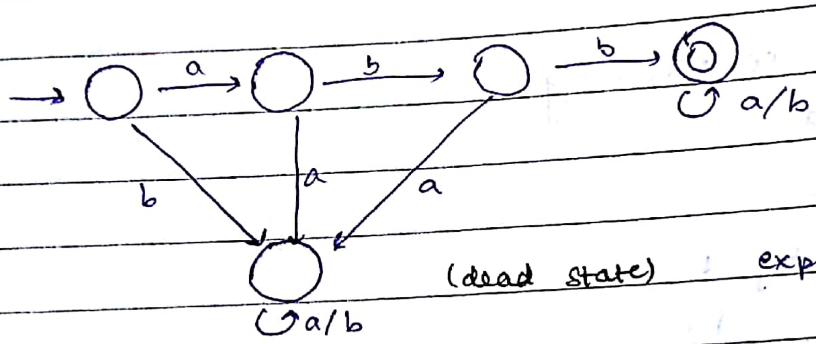
Dead State : It is a state which has all outgoing moves defined to itself, ie, $\forall\ a \in \Sigma$

$\delta(q_d, a) \rightarrow q_d$

Q Design a FA that accept ~~a~~ string starting with abb.

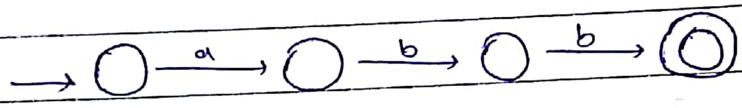


aba



(dead state) explicitely specified dead state

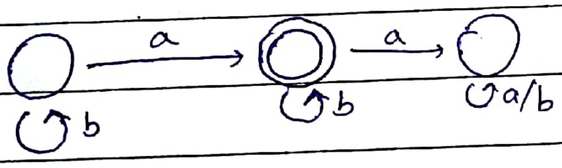
OR



implicite (moves not defined are considered as dead moves)

→ $\Sigma = \{a, b\}$

Accept all string containing exactly 1 a.

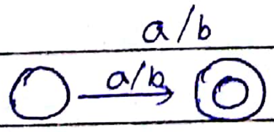


Q. What will be min no. of states required to accept string of length n over $\Sigma = \{a, b\}$

(a) n ✓(b) n+1 (c) ~~n−1~~ (d) None of these

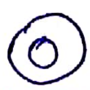
Use principle of induction

1.) n = 1



2.) n = 2 : 3 states

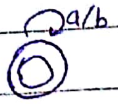
3.) n = 0



⇒ n + 1

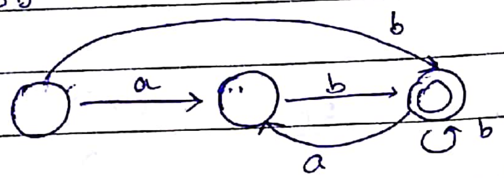
Teacher's Signature

* length : exactly n ⇒ dead move after (O) in all cases
: atleast n ⇒ we can write (O) ↺ a/b (length can be more than n)

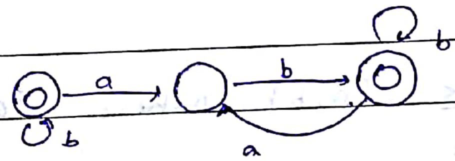


* : automata accepting ∧.
* Automata should not accept anything other than what's mentioned.

→ accept all strings having a followed by b
bb ✓ ab ✓ aba ✗
b → should accept b

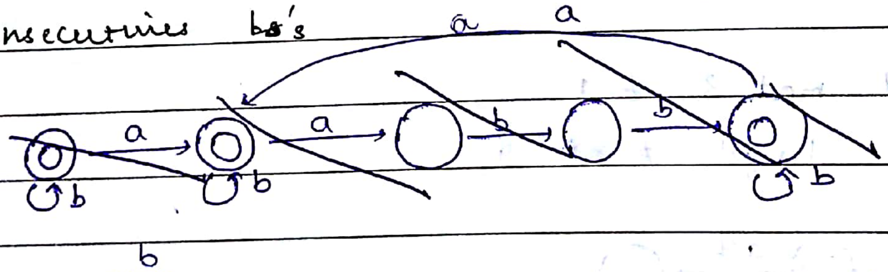


OR

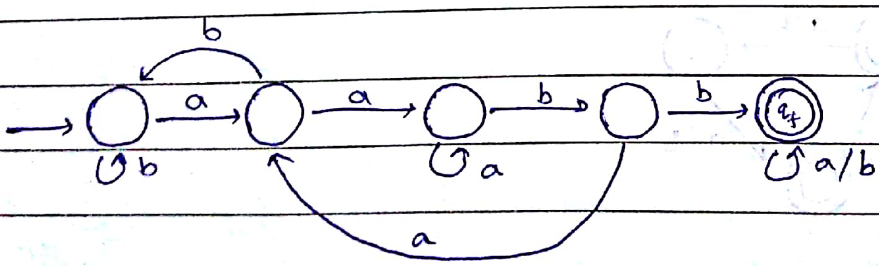


* We can've multiple Final set but only 1 initial set in this type

→ accept all strings containing ~~atleast~~ atleast 2 consecutive 'a' followed by 2 consecutive b's



ab ⇓
~~1 time rule~~ is followed, it is dones



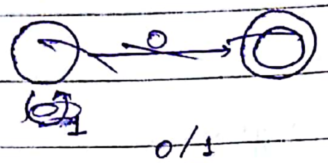
aaabb ✓
aabb ✓

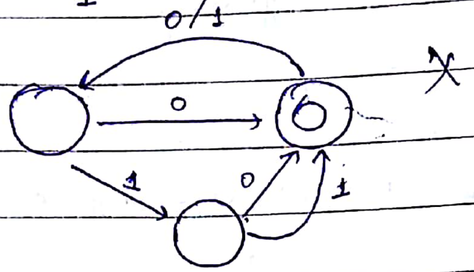
* If rule is ~~not~~ followed with no occurence also, 1st state will be final state.

Teacher's Signature

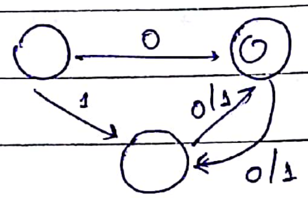
→ all string that start with 0 have odd length & start with 1 have even length

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

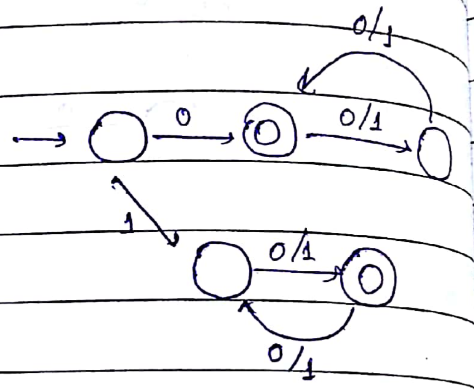




→

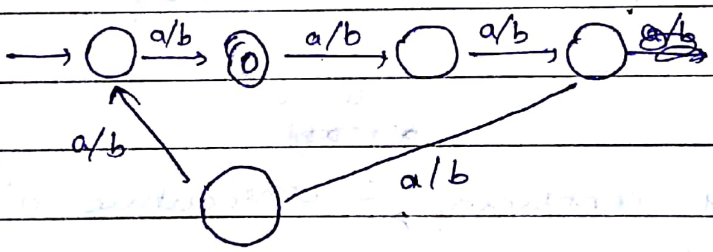


OR



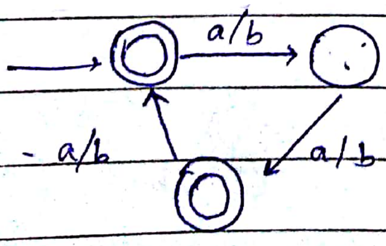
→ Draw a FA over $\Sigma = \{a,b\}$ where $|w| \bmod 5 = 1$

⇒ length : 1, 6, 11, ...

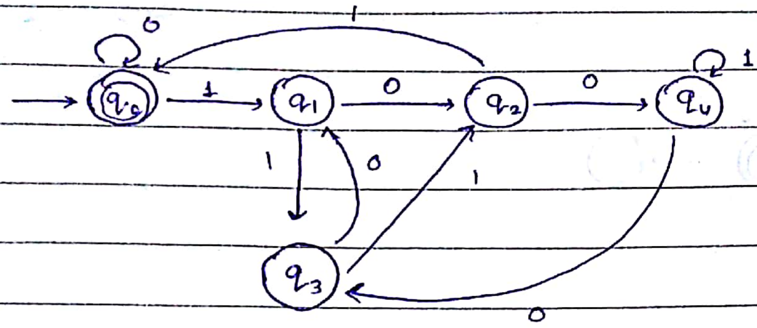


→ $|w| \bmod 3 \neq 1$

⇒ length $\neq$ 1, 4,



→ accept all binary string divisible by 5.



Rem
0
0
10
11

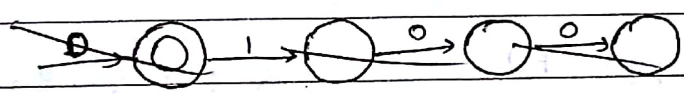
→ all string that don't contain 100 as substring

→ " " where every odd position is 1.

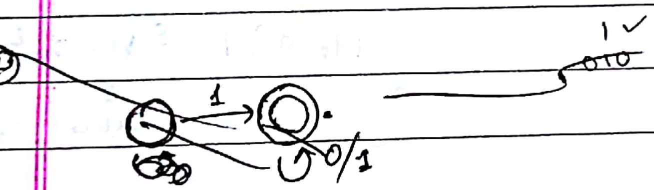
→ all string containing equal occurrence of 01 and 10.

→ " " even a's and odd b's.

①

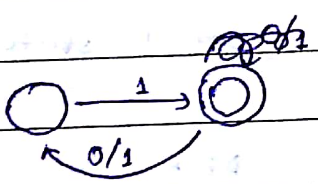


②

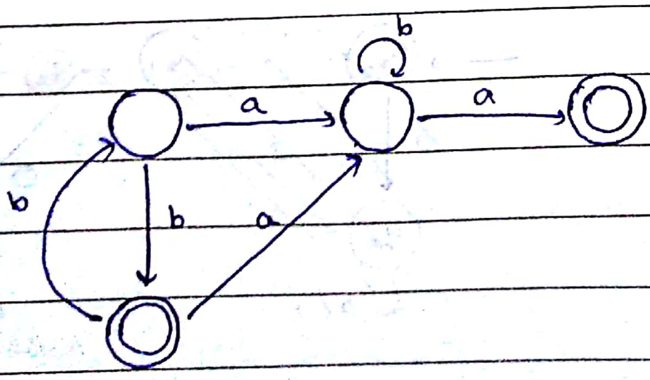


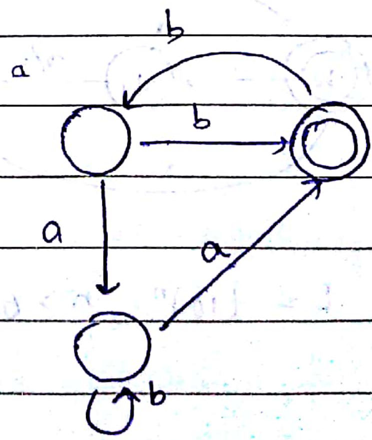
011

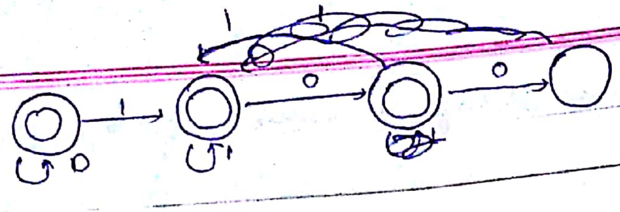
②



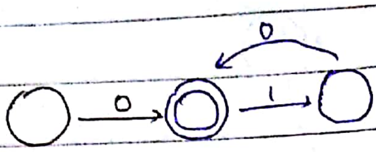
④





(1) 

010

(3) 

Non - Deterministic Finite Automata

5 tuple $(Q, \Sigma, \delta, q_0, F)$ $\qquad Q = \{q_0, q_1, q_2\}$

$\delta : Q \times (\Sigma \cup \Lambda) \longrightarrow 2^Q$, $\qquad (q_0, a) \vdash \{q_1, q_2\}$

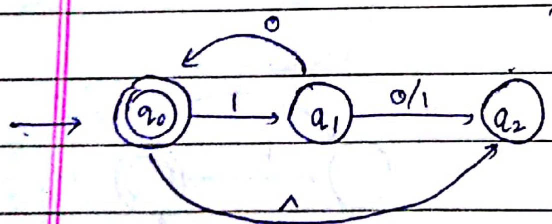
subset of Q

→ Acceptance of string :-

On null move also, we can move from 1 state to another

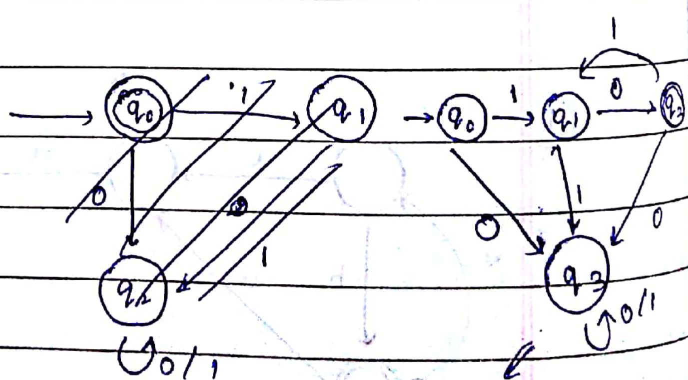
NDFA

dead state



$L = (10)^n , n > 0$

DFA



dead state

$L = (10)^n , n > 0$

not accepting null

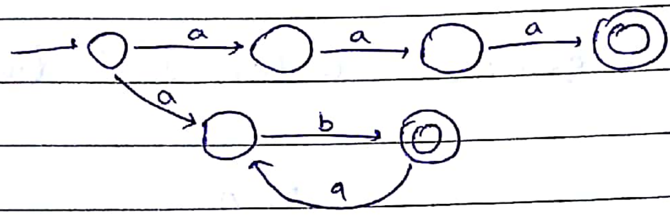
1010

→ $(q_0, 1010) \vdash (q_1, 010) \vdash (q_2, 10) \vdash (q_2, 0) \vdash (q_2, \wedge)$

OR

→ $(q_0, 1010) \vdash (q_1, 010) \vdash (q_0, 10) \vdash (q_1, 0) \vdash (q_0, \wedge)$

Eg. $L = (a^3 \cup (ab)^n \quad n > 0)$



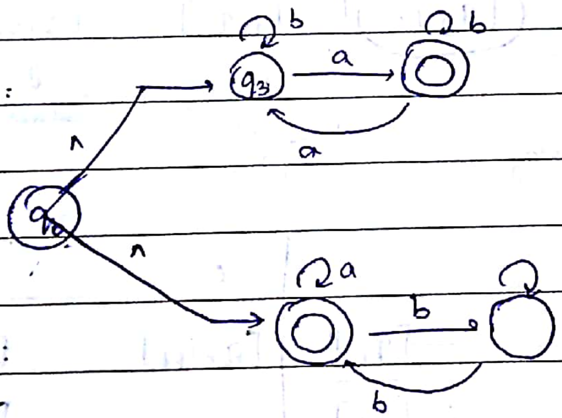
Eg. $L = L_1 \cup L_2$

L_1 : a in odd no.

L_2 : b in even no.

L_1 :

L_2 :



$(q_0, aaab) \vdash (q_3, aaab)$

⇓

State has changed but string is the same ⇒ using '$\wedge$'

** **NDFA to DFA**

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

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

$Q' = [q_0, q_1, \ldots q_i]$ where $q_0, q_1, \ldots q_i \in Q$

Eg. : NDFA : $\delta(q_0, a) \vdash q_1, q_2$

DFA : $[q_1 q_2]$

Teacher's Signature

$q_0' = [q_0]$

$\Sigma' = \Sigma$

F' = it is state that contains at least 1 final state of NDFA

$[q_0\ q_1\ \ldots\ q_k] \in F'$ if $\exists\ q_i \in F$

$\delta'([q_1\ q_2\ \ldots q_i], a) \vdash (q_1, q_2 \ldots, q_k)$

where $\delta(q_1,a) \cup \delta(q_2,a) \cup \quad \cup (q_i,a) = \{q_1, q_2, \ldots, q_k\}$

Eg

NDFA

	a	b
$\rightarrow q_0$	q_0	q_1
q_1	q_1	$q_0\ q_1$

DFA

	a	b
$\rightarrow [q_0]$	$[q_0]$	$[q_1]$
$[q_1]$	$[q_1]$	$[q_0 q_1]$
$[q_0 q_1]$	$[q_0 q_1]$	$[q_1 q_0]$

final state all containing q_0.

union of $([q_0],a)$ & $([q_1],a)$

same as $[q_0 q_1]$

Eg.

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

NDFA

	a	b
$\rightarrow [q_0]$	$[q_0 q_1]$	$[q_2]$
$[q_2]$		$[q_0 q_1]$
$[q_0 q_1]$	$[q_0 q_1]$	$[q_2 q_1]$
$[q_2 q_1]$	$[q_0]$	$[q_0\ q_1]$

DFA

[Don't see which states are defined in NDFA] eg.

	0	1	2
$\rightarrow q_0$	$q_0 q_4$	q_4	$q_2 q_3$
q_1		q_4	
q_2	q_2		q_3
q_3		$q_4 q_1$	
q_4	q_1	q_2	$q_1 q_2$

DFA:

	0	1	2
$\rightarrow [q_0]$	$[q_0 q_4]$	$[q_4]$	$[q_2 q_3]$
$[q_0 q_4]$	$[q_0 q_4 q_1]$	$[q_4 q_2]$	$[q_1 q_2 q_3]$
$[q_4]$	$[q_1]$	$[q_2]$	$[q_1 q_2]$
$[q_2 q_3]$	$[q_2]$	$[q_4 q_1]$	$[q_3]$
$[q_0 q_4 q_1]$	$[q_0 q_4 q_1]$	$[q_2 q_4]$	$[q_1 q_2 q_3]$
$[q_2 q_4]$	$[q_1 q_2]$	$[q_2]$	$[q_1 q_2 q_3]$
$[q_1 q_2 q_3]$	$[q_2]$	$[q_1 q_4]$	$[q_3]$
$[q_1]$		$[q_4]$	
$[q_2]$	$[q_2]$		
$[q_1 q_2]$	$[q_2]$	$[q_4]$	
$[q_1 q_4]$	$[q_1]$	$[q_2 q_4]$	$[q_1 q_2]$
$[q_3]$		$[q_1 q_4]$	

Eg. NDFA

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

DFA :

	a	b
$\to [q_0]$	$[q_1, q_2]$	$[q_2]$
$[q_1, q_2]$	$[q_0, q_2]$	$[q_0, q_1, q_3]$
$[q_2]$	$[q_0]$	$[q_1, q_0]$
$[q_0, q_2]$	$[q_0, q_1, q_2]$	$[q_0, q_1, q_2]$
$[q_0, q_1, q_3]$	$[q_1, q_2]$	$[q_1, q_2, q_3]$
$[q_1, q_0]$	$[q_1, q_2]$	$[q_1, q_2, q_3]$
$[q_0, q_1, q_2]$	$[q_0, q_1, q_2]$	$[q_0, q_1, q_2, q_3]$
$[q_1, q_2, q_3]$	$[q_0, q_2]$	$[q_0, q_1, q_2, q_3]$
$[q_0, q_1, q_2, q_3]$	$[q_0, q_1, q_2]$	$[q_0, q_1, q_2, q_3]$

17-1-18

NFA with Λ transition

NFA to DFA

1. Λ - moves (remove)
2. NFA to DFA
3. minimize it
4. Implement it

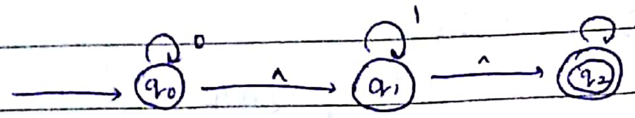
(jaha bhi hum q_2 se ja skte h , waha hum q_1 se bhi ja [sakte] h)

⇒ ③ $q_1 \xrightarrow{\wedge} q_2$

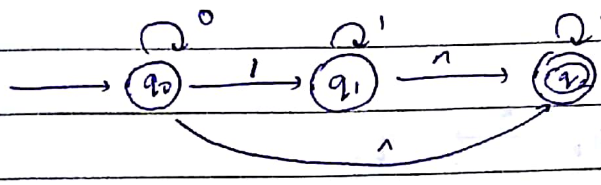
① Duplicate all moves from q_2 to q_1 itself

② If q_2 is FS , then make q_1 also as final state

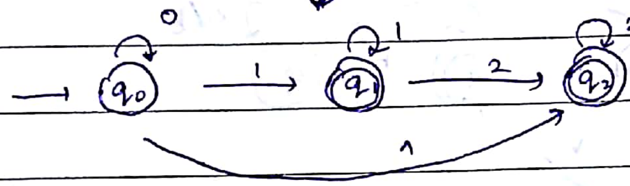
Eg.



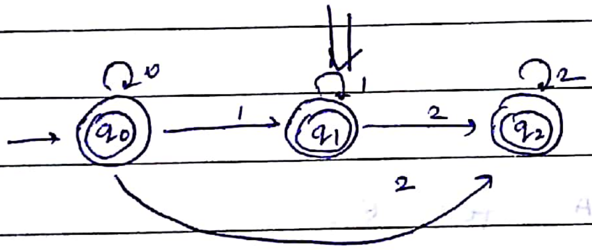
⇓ Remove $q_0 \xrightarrow{\wedge} q_1$



⇓ Remove $q_1 \xrightarrow{\wedge} q_2$



⇓



OR

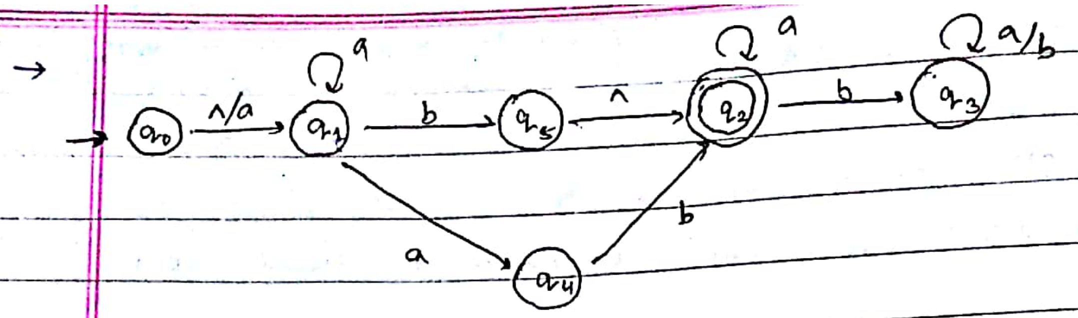
	0	1	2	$\wedge$
→ q_0	q_0	q_1		~~q_1~~ q_2
q_1		q_1		q_2
q_2			q_2	

Final table

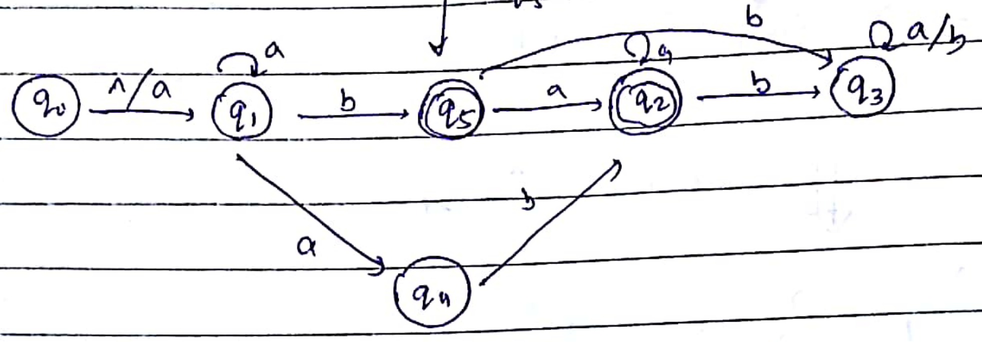
	0	1	2	$\wedge$
→ q_0	q_0	q_1	q_2	
q_1		q_1	q_2	
q_2			q_2	

Remove all

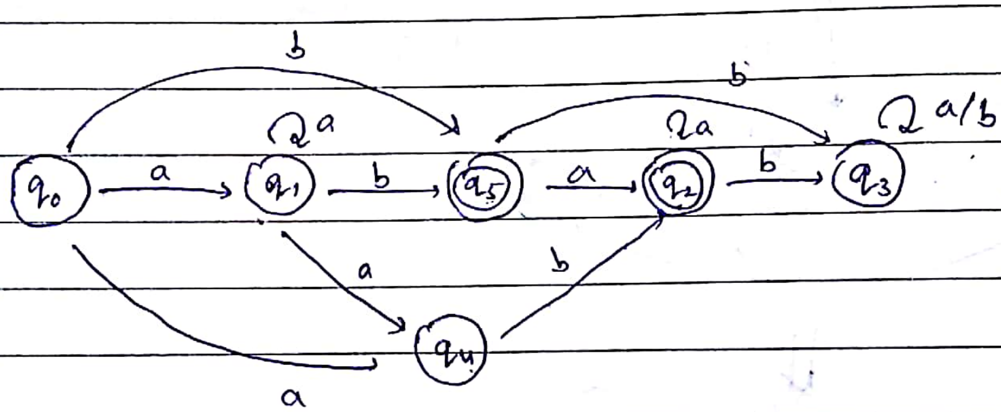
Teacher's Signature



$q_5 \xrightarrow{\wedge} q_2$



$q_0 \xrightarrow{\wedge} q_1$



→ How to check a NFA for ϵ.

ϵ- closure (q)

→ is the set of state possible to reach from q with $\wedge$ moves

ϵ - closure (q)

1. $q \in \epsilon$ - closure (q)
2. if $\delta(q, \wedge) \to q_i$, then $q_i \in \epsilon$-closure (q)
3. $\forall\ P \in \wedge$ - closure (q) if $\delta(p, \wedge) \to r$

then $r \in \wedge$ - closure (q)

Teacher's Signature

$\delta(B, 101)$

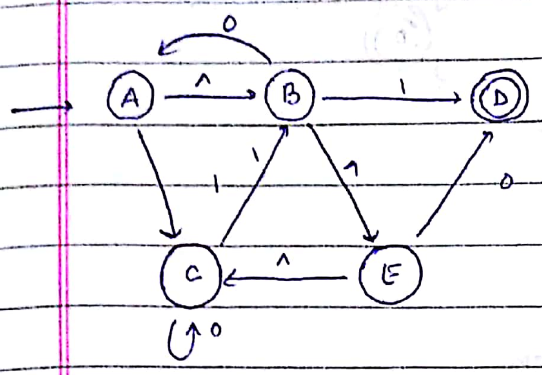
$\rightarrow (\{BEC\}, 1) \rightarrow \{DB\}$

$\rightarrow (\{DBEC\}, 0) \rightarrow \{ADC\}$

$\rightarrow (\{ABCED\}, 1) \rightarrow \{CDB\}$

$\rightarrow \{CBDE\}$

Repeat 3 till no new state is added in $\wedge$- closure (q)



ϵ- closure (A) = $\{A, B, E, C\}$

ϵ- closure (C) = $\{C\}$

$\rightarrow$ $\delta(q, w)$

$\epsilon\text{-closure}(q) \xrightarrow{\delta(\epsilon\text{-closure}(q), a)} P$

$\downarrow$

ϵ- closure (P) = R

$\downarrow \delta(R, a)$

S

$\downarrow$

ϵ- closure (S)

Eg.: $\delta(A, 01)$.

$(\{A, B, E, C\}, 01)$.

$\rightarrow \{ADC\} \rightarrow (\underbrace{\{ABECD\}}_{\text{closure } \{ADC\}}, 1) \rightarrow \{CDB\}$

$\downarrow$

$\{CDEB\}$

Eg. $\delta(B, 101)$

$\rightarrow \{B, E, C\} \rightarrow (\{DB\}, 01) \rightarrow \{DBEC\}. \rightarrow (\{ADC\}, 1)$

$\rightarrow \{CB\} \rightarrow \{BEC\}$

{BEC}, 1 → {CBDE}

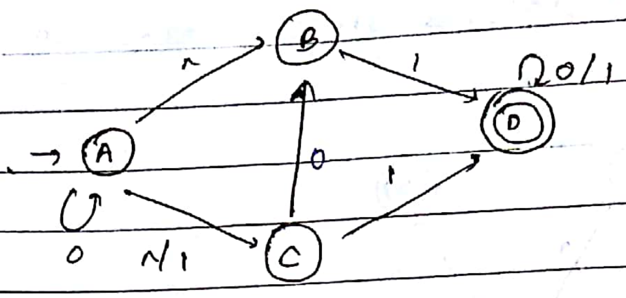
→ ({BBEC}, 0)

→ {AD ~~B~~ BE}, 1

DATE :

PAGE NO. :

eg.



Task 1 δ (A, 01)

Task 2 Remove ^ - move

Task 3 Convert NFA to DFA

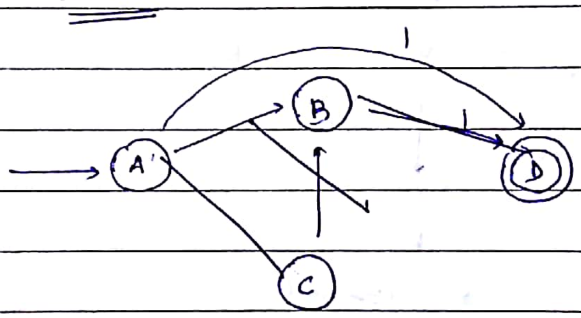
① → δ (A, 01)

→ ({A B, C}, 0) → ~~B~~ {A B}

→ ({A B C}, 1) → {C D}

→ {C D}

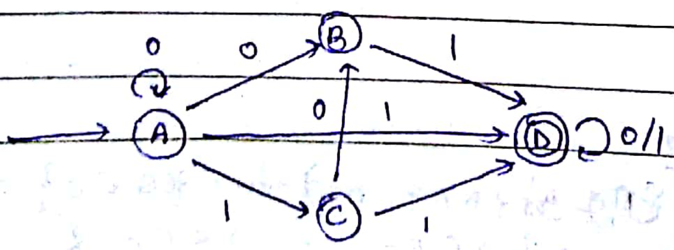
②



	0	1	^
→ A	A	C	BC
B		D	
C	B	D	
D	D	D	

→

	0	1	^
→ A	AB	CD	~~B~~ C
B		D	
C	B	D	
D	D	D	



Teacher's Signature

(3) NFA to DFA

	0	1
→ A	AB	CD
B		D
C	B	D
D	D	D

: NFA

DFA :

	0	1
→ [A]	[AB]	[CD]
[AB]	[AB]	[CD]
[CD]	[BD]	[D]
[BD]	[D]	[D]
[D]	[D]	[D]

22-01-18

Minimization of ~~Deterministic~~ Finite Automata

- State Equivalence : q_1 and q_2 are equivalent iff

1.) $(q_1, x) \rightarrow q_f$ $\quad q_f \in F$

then

$(q_2, x) \rightarrow q_f$ $\quad \forall x \in \Sigma$

2.) $(q_1, x) \vdash q_j$ $\quad q_j \notin F$

$(q_2, x) \vdash q_k$ $\quad q_k \notin F$

→

	a	b	c
→ q_1	q_1	q_2	q_4
q_2	q_1	q_3	q_4
q_3	q_4	q_2	q_4
q_4	q_1	q_1	q_4

NF NF F

⇒ q_1 & q_2 : state equivalent

don't consider for final state

Steps :

① Remove all dead and unreachable set

② Divide Q into two sets Q_1 and Q_2 containing set of NF and final state

③ 1.) check equivalence for every set in Q_1 and Q_2 if the state in set are not equivalent split the set. Repeat till all states in set are equivalent.

→

	0	1
→ q_0	q_1	q_5
q_1	q_6	q_2
q_2	q_0	q_2
~~q_3~~	~~q_2~~	~~q_6~~
q_4	q_7	q_5
q_5	q_2	q_6
q_6	q_6	q_4
q_7	q_6	q_2
~~q_8~~	~~q_8~~	~~q_8~~

Reachable state :

q_0 q_6

q_1 q_7

q_2

q_4

q_5

q_8 → dead state ⇒ remove

→ unreachable state ⇒ q_3 : Remove

Assume : Automata contain only 2 states :

N.F. Final

$P_1 (q_0, q_1, q_4, q_5, q_6, q_7)$ $P_2 (q_2)$ → single state ⇓ no split

$q_0, a \to P_1$

$q_0, b \to P_1$ $P_3 (q_0, q_4, q_6)$ $P_4 (q_1, q_5, q_7)$

$q_1, a \to P_1$

$q_1, b \to P_2$

$P_5 (q_0, q_4)$ $P_6 (q_6)$ $P_7 (q_1, q_7)$ $P_8 (P_5)$

Now, we have B_2 B_3 B_4

$q_0, a \to P_4$	$q_1, a \to P_6$	$q_0, a \to P_7$	$q_1, a \to P_6$
$q_0, b \to P_4$	$q_1, b \to P_2$	$q_0, b \to P_8$	$q_1, b \to P_2$
$q_4, a) \to P_4$	$q_5, a \to P_2$	$q_4, a \to P_7$	$q_7, a \to P_6$
$q_4, b \to P_4$	$q_5, b \to P_6$	$q_4, b \to P_8$	$q_7, b \to P_2$
$q_6, a \to P_3$	$q_7, a \to P_6$		
$q_6, b \to P_3$	$q_7, b \to P_2$		

q_0 & q_4 → equi q_1 & q_7 : equi.

	a	b
→ P_5	P_7	P_8
P_6	P_6	P_5
P_7	P_6	P_2
P_8	P_2	P_6
P_2	P_5	P_2

→ Final State isn't dead state

DATE : / /
PAGE NO. :

Ex.

	a	b
$\rightarrow q_0$	q_1	q_0
q_1	q_0	q_2
q_2	q_3	q_1
q_3 (final)	q_3	q_0
q_4	q_3	q_5
q_5	q_6	q_4
q_6	q_5	q_6
~~q_7~~	~~q_6~~	~~q_3~~

q_0
q_1
q_2
q_3
q_4
q_5
q_6

unreachable : ~~q_7~~

Q

- $P_1 (q_0, q_1, q_2, q_4, q_5, q_6)$
  - $P_3 (q_0, q_1, q_5, q_6)$
    - P_5
    - P_6
  - $P_4 (q_2, q_4)$
- $P_2 (q_3)$

q_0 — P_1, P_1

q_1 — P_1, P_1

q_2 — P_2, P_3

q_4 — P_2, P_3

q_2 — P_2, P_6

q_0 — P_3, P_3

q_1 — P_3, P_4

q_5 — P_3, P_4

q_6 — P_3, P_3

$P_5 (q_0, q_6)$ $\quad$ $P_6 (q_1, q_5)$

q_0 — P_6, P_5

q_6 — P_6, P_5

q_1 — P_5, P_4

q_5 — P_5, P_1

Teacher's Signature

Scanned by CamScanner

	a	b
$\rightarrow P_5$	P_6	P_5
P_6	P_5	P_4
P_4	P_2	P_6
P_2	P_2	P_5

$\rightarrow q_0$	q_1	q_0
q_1	q_0	q_2
q_2	q_3	q_1
q_3	q_3	q_0

Ans.

Que

	a	b
$\rightarrow q_0$	q_0	$q_0 q_2$
q_1	q_3	$q_2 q_3$
q_2	q_4	–
q_3	q_4	–
q_4	–	–

(NFA $\longrightarrow$ DFA first)

		a	b
A	$\rightarrow [q_0]$	$[q_0]$	$[q_0 q_2]$
B B_0	$[q_0 q_2]$	$[q_0 q_4]$	$[q_0 q_2]$
C B_0	$[q_0 q_4]$	$[q_0]$	$[q_0 q_2]$

Unreachable :

Q

	a	b
$\rightarrow A$	A	B
B	C	B
C	A	B

Q

$Q_1 \{A \quad B\}$ $\qquad$ $Q_2 \{C\}$

$Q_2 \{A\}$ $\qquad$ $Q_3 \{B\}$

A < A, B

B < C, B

Teacher's Signature

	a	b
→ A	A	B
B	C	B
C	A	B

29-01-18

Regular Expression :

→ it is an algebric notation for the language accepted by FA

→ it includes 3 symbols : $+$, $\cdot$, $*$

$+$: union

$\cdot$: concatenation

$*$: Kleene closure

↳ a^* = 0 or more occure of a

$L = a^*$

$= \{\Lambda, a, aa, aaa, \ldots\}$

↳ $(a+b)$: either a or b

↳ $a \cdot b$

① L = all string containing abb as substring

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

all strings possible from a & b

$(a+b)^* = \{\Lambda, a, b, ab, aa, bb, ba, aaa, \ldots\}$

(a+b)(a+b)

$$(a+b)^* \; abb \; (a+b)^*$$

② all string ~~ss~~ ending with abb

$$(a+b)^* \; abb$$

③ starting with abb

$$abb \; (a+b)^*$$

④ even no. of a's.

$$(b^* a b^* a b^*)^* \quad \text{or} \quad b^* (a b^* a)^* b^*$$

not interested in b

⑤ atleast 3 a's

~~aaa~~

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

⑥ exactly 3 a's

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

Q. $L_1 = a \cdot b + c$

$L_2 = (a \cdot b) + c$

NO * ⇒ finite set

L_1 : $a \cdot b + c$

$a \cdot (b+c)$ → (ab, ac)

$(ab + c)$ → (ab, c)

Teacher's Signature

* Precedence graph:
  → Star
  → Concatenation
  → Union

* L_1 : $a \cdot b + c \Rightarrow (ab) + c$
  $= (ab, c)$

Eg. $L = a^{2m} b^{2n+1} a b \quad m > 0 \quad n > 0$

→ This is not in regular expression

⇒ even no. of a, then odd no. of b followed by a b

R_1 : $(aa)^* (bb)^* b \quad a \cdot b$ ✗ [because ~~m, n > 0~~]

R_2 : $aa(aa)^* bb(bb)^* b a b$ ✓

1st string possible : $a^2 b^3 a b$
So, R_2 is correct.

* a^* : {0 or more} Kleene Closure
  aa^* : {1 or more}
  $aa^* = a^+$: Positive Kleene's Closure

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

Scanned by CamScanner

Eg. atleast 1 pair of consecutive zero

$\Sigma = 0, 1$

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

*** Eg. has no pair of consecutive zeroes ⇒ whenever 0 is occurring, it should be followed by 1.

$$(1+01)^*(0+\wedge)$$

either 1 or 01 will appear — can have single zero or not zero

Kleene Theorem

→ For every R.E., there exists a FA to accept it.

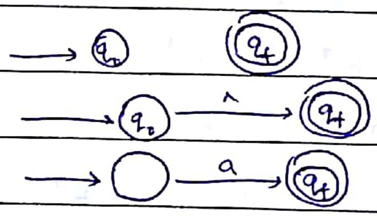
Lang
↓
R.E
↓
ε - NFA
↓
NFA
↓
DFA
↓
Minimize

Proof : Based on Induction

$R = \phi$

$R = \wedge$

$R = a$



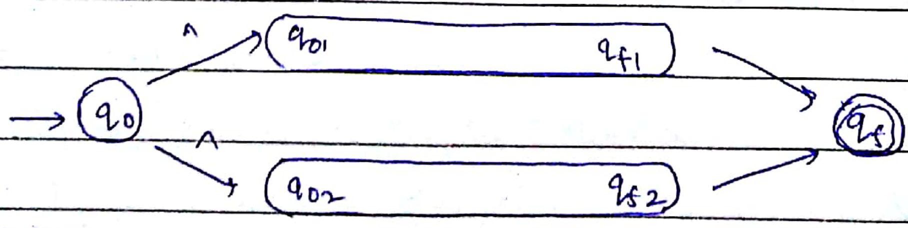
Basic

let us assume there exist FA for ~~R.E~~ R.E. R_1 and R_2

R_1 : it is $M_1(Q_1, q_{01}, \Sigma_1, \delta_1, q_{f1})$

R_2 : $M_2(Q_2, q_{02}, \Sigma_2, \delta_2, q_{f2})$

1.) $R = R_1 + R_2$



$M'(Q, \Sigma, \delta, q_0, q_f)$

$Q = Q_1 \cup Q_2 \cup \{q_0, q_f\}$

$\Sigma = \Sigma_1 \cup \Sigma_2$

$\delta = \delta_1 \cup \delta_2 \cup \; q_0, \wedge \rightarrow \{q_{01}, q_{02}\}$

$\cup \; q_{f1}, \wedge \rightarrow q_f$

$\cup \; q_{f2}, \wedge \rightarrow q_f$

30-01-18

Finite Automata to Regular Expression

Identities :

symbol
$\wedge$: Empty string
ϕ : Null (No transition)
set [Not even a single $\wedge$]

1. $\phi + R = R$
2. $\phi \cdot R = \phi$
3. $\wedge \cdot R = R$
4. $\wedge^* = \wedge$
5. $R^* R = R R^* = R^+$
6. $R^* R^* = R^*$
7. $R + R = R$
8. $\wedge + RR^* = R^*$ $\left[\begin{array}{l} RR^* = R^+ \\ R^+ + \wedge = R^* \end{array} \right]$
9. $(PQ)^* P = PQPQ \; PQP = P(QP)^*$
10. $(P+Q)^* = (P^* Q^*)^* = (P^* + Q^*)^*$
11. $(P+Q)R = PR + QR$

Arden Theorem :

$R = Q + RP$ and $P \neq \wedge$

then $R = QP^*$

8th → 17th Feb

DATE: / /
PAGE NO.:

Proof. 1.) let $R = QP^*$

$R = Q + RP$

$= Q + QP^*P$

$= Q(\Lambda + P^*P) = QP^*$ I_8

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

$= Q + QP + RP^2$

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

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

$\vdots$

$= \underbrace{(Q + QP + QP^2 + \dots + QP^i)}_{A} + \underbrace{(RP^{i+1})}_{B}$

let $w \in R$ and $|w| = i$

$\therefore R \in (Q + QP + \dots + QP^i)$

or

$R \in RP^{i+1}$

$P \not\ni \Lambda \Rightarrow |RP^{i+1}| \geq i+1$ [~~each P is atleast of~~ length 1]

Therefore $w \notin RP^{i+1}$ [$\because |w| = i$]

$\Rightarrow w \in (Q + QP + QP^2 + \dots + QP^i)$

$\Rightarrow R = Q(\Lambda + P + P^2 + \dots\dots + P^i)$

$R = QP^*$

Hence Proved

Method : FA to RE

Write eq^n for each state $q_i \in Q$ describing its reachability from other states

$q_1 = \alpha_{11} q_1 + \alpha_{12} q_2 + \alpha_{13} q_3 + \dots + \alpha_{1n} q_n$

$q_2 = \alpha_{21} q_1 + \alpha_{22} q_2 + \alpha_{23} q_3 + \dots + \alpha_{2n} q_n$

$\vdots$

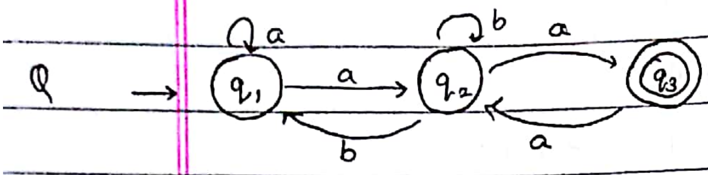
$q_n = \alpha_{n1} q_1 + \alpha_{n2} q_2 + \alpha_{n3} q_3 + \dots + \alpha_{nn} q_n$

Teacher's Signature

Apply identities and Arden Theorem & generate

If $q_f = q_0 \alpha_{ij}$ Remove q_0

This is final ~~automata~~ answer



(How to reach q_1)

$q_1 = aq_1 + q_2 b$ — ①

$q_2 = q_1 a + q_2 b + q_3 a$ — ②

$q_3 = q_2 a$ — ③

finally, get in form of $\underline{q_3 = q_1 \alpha_{ij}}$.

① & ② : $q_2 - q_1 = q_3 a$

$\frac{q_3}{a} - q_1 = q_3 a$

q_3

② & ③ :

$q_2 = q_1 a + q_2 b + q_2 aa$

$\underset{R}{q_2} = \underset{Q}{q_1 a} + \underset{R}{q_2} \underset{P}{(b + aa)}$

Arden's theorem : $q_2 = q_1 a (b+aa)^*$ — ④

① & ④

$q_1 = q_1 a + q_1 a (b+aa)^* b$

$\underset{R}{q_1} = \underset{R}{q_1} \underset{P}{(a + a(b+aa)^* b)} + \underset{Q}{\Lambda}$ $(\because R + \Lambda = R)$

Teacher's Signature

Multiple final set : perform for each, take union

DATE : / /
PAGE NO. :

Arden :

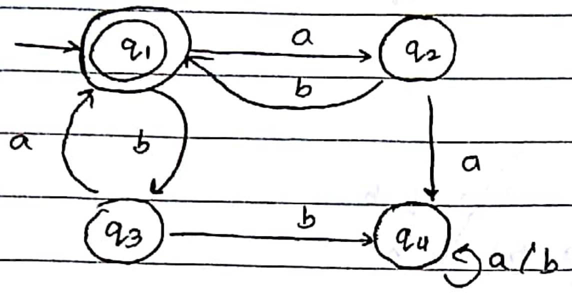
$$q_1 = \wedge \, (a + a(b+aa)^* b)^* \quad — \text{⑤}$$

$$= (a + a(b+aa)^* b)^* \qquad [\because R \cdot \wedge = R]$$

③ : $q_3 = q_1 \, a \, (b+aa)^* a$

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

Q.



$q_1 = q_2 b + q_3 a$

$q_2 = q_1 a$

$q_3 = q_1 b$

$q_4 = q_4 a + q_4 b$

$q_1 = q_1 ab + q_1 ba$

$q_1 = q_1 (ab)$

$= \cancel{q_1} \, ab \, [q_1 (ab)]$

$= \cancel{(ab)^*} q_1$

$q_1 = q_1 ab + q_1 ba$

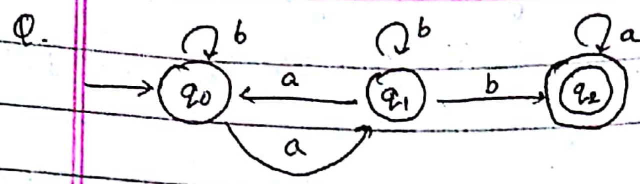
$$\underset{R}{q_1} = \underset{R}{q_1} \underset{P}{(ab + ba)} + \underset{Q}{\wedge}$$

Arden :

$$q_1 = \wedge (ab + ba)^*$$

$$\rightarrow q_1 = (ab + ba)^*$$

Q.



$q_0 = q_0 b + q_1 a$ —— ①

$q_1 = q_0 a + q_1 b$ —— ②

$q_2 = q_1 b + q_2 a + q_2 b$ —— ③

$q_2 = (\ )\, q_0$

$\cancel{q_0 = q_0 b + q_1 a}$

① + ② $\Rightarrow$ $q_0 + q_1 = q_0 b + q_0 a + q_1 a + q_1 b$

$\Rightarrow \underbrace{q_0 + q_1}_{R} = \underbrace{(q_0 + q_1)}_{R}\underbrace{(a+b)}_{P} + \underbrace{\wedge}_{Q}$

$\Rightarrow q_0 + q_1 = \wedge (a+b)^*$

$q_0 + q_1 = (a+b)^*$ —— ④

$\underbrace{q_2}_{R} = \underbrace{q_1 b}_{Q} + \underbrace{q_2}_{R}\underbrace{(a+b)}_{P}$

$\Rightarrow q_2 = q_1 b (a+b)^*$ —— ⑤

$\Longrightarrow$ $q_0 = q_0 b + (q_0 a + q_1 b)\, a$

$q_0 = q_0 b + q_0 a \cdot a + q_1 b a$

$\underbrace{q_0}_{R} = \underbrace{q_0}_{R}\underbrace{(b + a \cdot a)}_{P} + \underbrace{q_1 b a}_{Q}$

$q_0 = q_1 b a (b + aa)^*$ —— ⑥

⑥ & ②

$\underbrace{q_1}_{R} = q_0 a + q_1 b$

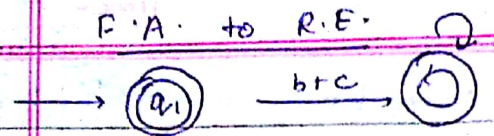
$\Rightarrow q_1 = q_1 ba(b+aa)^* a + q_1 b$

$\Rightarrow \underbrace{q_1}_{R} = \underbrace{q_1}_{R}\underbrace{(ba(b+aa)^* a + b)}_{P} + \underbrace{\wedge}_{Q}$

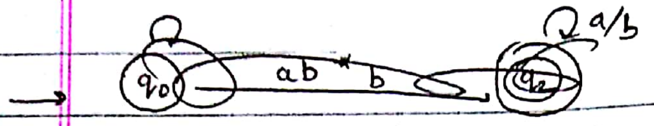
$\Rightarrow q_1 = (ba(b+aa)^* a + b)^*$ —— ⑦

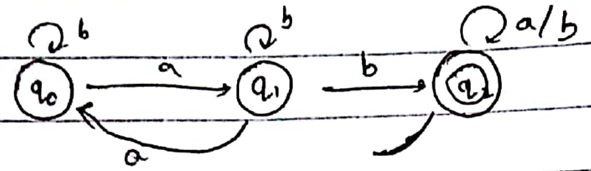
DATE: 31 / 1 / 18
PAGE NO.:

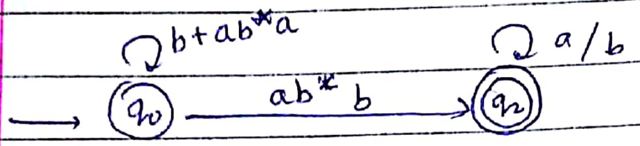
F.A. to R.E.



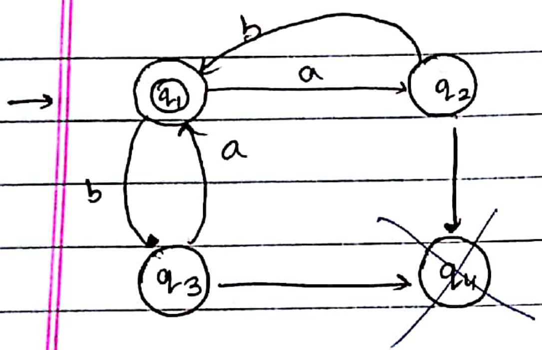
$(a^* + (b+c)a^*)^*$

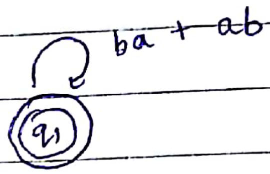






$(b+ab^*a)^* ab^* b (a+b)^*$





$(ab + ba)^*$

<u>Closure Properties on regular language</u> (If L_1 & L_2 : R.L. any opⁿ will give R.L.)

L_1 & L_2 are R.L.

1.) Union $L = L_1 \cup L_2$ ✓

2.) Concatenation $L = L_1 \cdot L_2$ ✓

3.) Kleene closure $L = L_1^*$ ✓

4.) Reversal :

$L = L_1^R$

$L_1 = (ab)^*$

$L_1^R = (ba)^*$

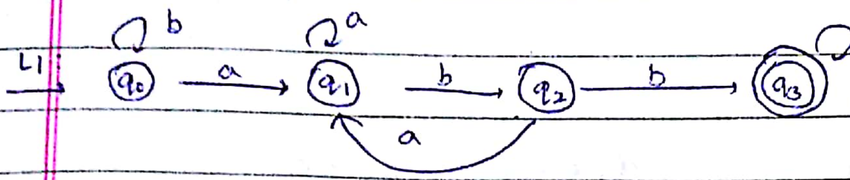
<u>Procedure</u> :

① Reverse the move

② Make IS as FS

③ Make FS as IS, if there are more than 1 FS then a new IS & $\wedge$ - move to all FS

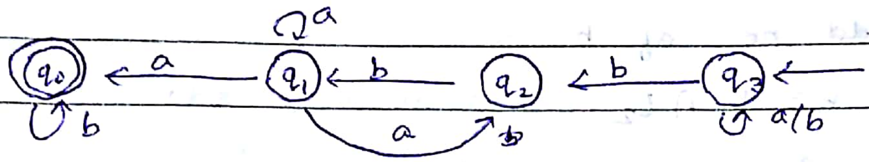
L_1:



abb

$(a+b)^*$ ~~aab~~ b $(a+b)^*$

Reversal : L_1^R :

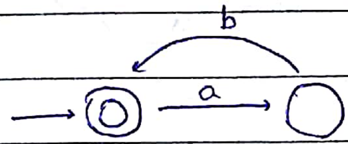


$(a+b)^* b\, ba (a+b)^*$

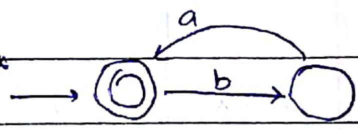
5.) Complement :

$L = \overline{L_1}$

$(ab)^*$



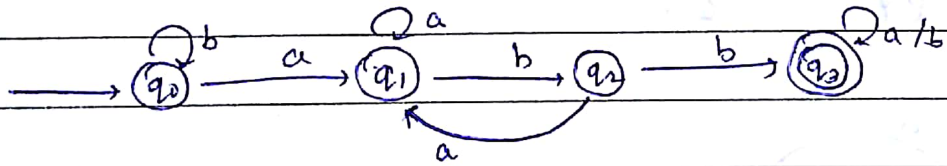
$(ba)^*$



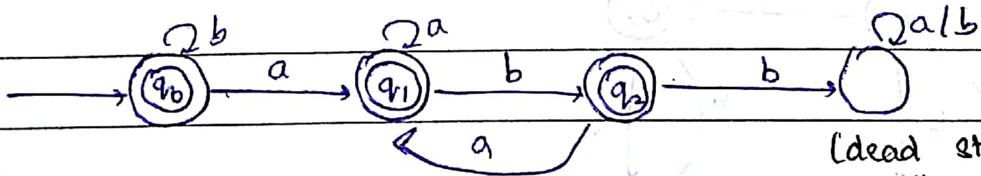
Procedure :

1. Make your F.S. as non-F.S.
2. Make all non-F.S. as F.S.

L_1 :



Complement :



(dead state)

⇓

will reject all strings containing 'abb'.

6.) Intersection : (AND)

$L = L_1 \cap L_2 = \overline{\overline{L_1} \cup \overline{L_2}} \equiv$ Regular language

$L_1 = \{q_{01}, q_{f1}, \Sigma, \delta_1, Q_1\}$

$L_2 = \{q_{02}, q_{f2}, \Sigma, \delta_2, Q_2\}$

Procedure :

→ Make $[q_{01}, q_{02}]$ as initial state for FA L.

→ $\delta([q_{1i}, q_{2i}], a) = \{\delta_1(q_{1i}, a), \delta_2(q_{2i}, a)\}$

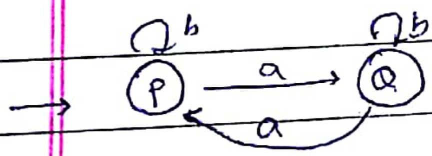
→ $[q_{f1}, q_{f2}] \in FS$

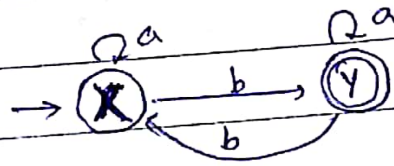
→ $Q = [q_i, q_j]$; $q_i \in Q_1$, $q_j \in Q_2$

Eg. L_1 : even no. of a

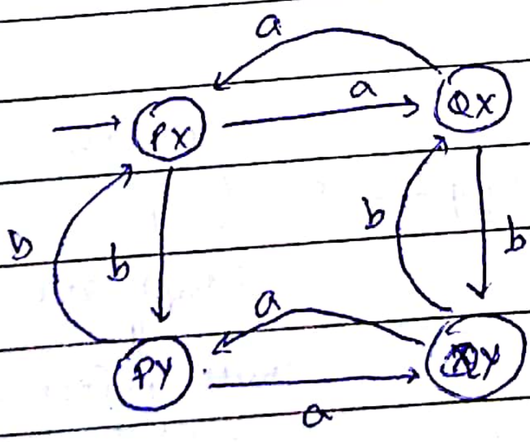
L_2 : odd no. of b

$L = L_1 \cap L_2$ ≡ even a & odd b



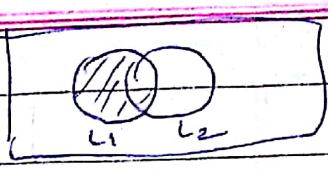


	a	b
→ PX	QX	PY
QX	PX	QY
PY	QY	PX
QY	PY	QX



PAGE NO. :

7) Set Difference :



$$L = L_1 - L_2$$
$$= L_1 \cap \overline{L_2}$$

01/02/18

Pumping Lemma :

(Based on Pigeonhole Principle)

we don't accept automata based on length of string

we accept automata based on property it follows

If L is Regular then each string $w \in L$ can be written as :

xyz ie, $w = xyz$

where $|y| \geq 1$, $|w| \geq m$

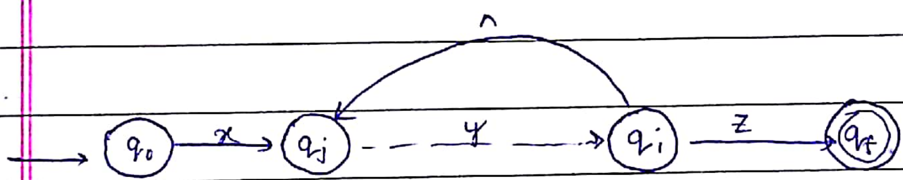
and $|xy| \leq m$

[string length : $\geq m$
no. of states $= m$]

then $\forall\ i \geq 0$ $\quad \underline{xy^iz \in L}$

→ Assuming y is repeated, not x & z.

string length : infinite
⇒ some part is repeated
⇓
y here



If we can find such y ⇒ It is regular
can't " ⇒ not "

Eg $L = 0^n 1^n$

Automata

* when not bothered about a symbol ∴ we do self-loop for it

→ F.A. : Every ° is remembered by state

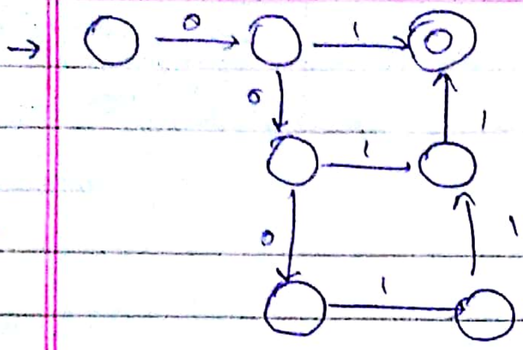
$n=1$ 01

$n=2$ 0011

Scanned by CamScanner

PAGE NO. :

$n > 0$



↳ string infinite

→ no. of states : ∞

→ Not Regular language

$|Q| = K$

$W = xyz = 0^m 1^m \in L$

If we find any i when $\notin L \Rightarrow$ Not Regular

$K \geq M$

let $y = 0$ $\quad |w| = 2m > K$

$\Downarrow$

$|y| \geq 1$

$W = \underset{x}{0^{m-1}} \; \underset{y}{0} \; \underset{z}{1^m}$

K : no. of states
any string is accepted with max. states K.

$i = 0 \quad w = 0^{m-1} 1^m \notin L$

$i = 1 \quad w = 0^m 1^m$

$i = 2 \quad W = 0^{m-1} 0^2 1^m = 0^{m+1} 1^n \notin L$

→ $y = 1$

$0^m 1 \, 1^{m-1}$

$i = 2 \quad 0^m 1^2 1^{m-1} \notin L$

→ $y = 01$

$i = 2 \quad 0^{m-1} (01)^2 y^{m-1}$

$0^{m-1} 01 \, 01 \, y^{m-1} = 0^m 101^m \notin L$

→ we can't find any such y which makes it L.

Teacher's Signature

* check for all possibilities of y.
If you find any y ~~which~~ for which we can iterate ~~which~~ for all i → It is regular language

Q. $L = 0^{2n}$ $n \geq 0$

$y = 00$

$x = \wedge$ → have to make no. of states finite

$z = 0^{2(n-1)}$

$xy^iz \in L$

$i = 0$ ✓

$i = 1$ ✓

$i = 2$ ✓ → $(00)^2 (0)^{2(n-1)} = 0^{2(n+1)} \in L$

→ $L = 0^p 1^p$ $p > 0$ $|Q| = n$

$|w| = m \geq n$ m : length of string

$w = xyz$ $|y| \geq 1$ $|xy| \leq n$

and xy^iz

5/2/17

Regular Grammer

$G(V, \Sigma, P, S)$

V → is set of NT (Non-terminal) [Represent class]

Σ → set of terminal

P → production

S → start symbol

S → NVN ⇒ V : {S, N, V}

N → Ram | Apple | Book Σ = {Ram, apple, book, ate, read}

V → ate | read

Eg. Ram ate apple

$S \rightarrow N\,V\,N$

$\rightarrow$ Ram V N

$\rightarrow$ Ram ate N

$\rightarrow$ Ram ate apple → this belongs to grammer.

Regular grammer :

a grammer G is regular if all the productions are of form

$A \rightarrow a \mid aB$ (Right Linear)

or

$A \rightarrow a \mid Ba$ (Left Linear)

[NT should be 1, either in xtreme left or in xtreme right]

Eg. $S \rightarrow abA$

$A \rightarrow a \cdot A$

$A \rightarrow a$

$S \rightarrow abA$

$S \rightarrow aba$

$S \rightarrow abA$

$S \rightarrow abaA$

$S \rightarrow abaaA$

$L = abaa^*$

Eg. $S \rightarrow Aba$

$A \rightarrow a$

$A \rightarrow Ba$

$B \rightarrow Ba$

$B \rightarrow a$

$S \rightarrow Aba$ $\quad\quad (a \ldots)^* \, aba$

$S \rightarrow aba$ —

$S \rightarrow Aba$

$S \rightarrow Baba$ $\quad\quad S \rightarrow Baba$

$S \rightarrow aaba$ —

Eg. $S \rightarrow aA$

$A \rightarrow Bb \mid b$

$B \rightarrow aB \mid a$

$S \rightarrow aA$

$S \rightarrow ab$ $\quad\quad S \rightarrow aBb$

$S \rightarrow aab$ $\quad\quad S \rightarrow aaBb$

$S \rightarrow aaaBb$

a^*ab

Eg. $S \rightarrow aA$

$A \rightarrow Bb \mid b$

$B \rightarrow aA$

$\boxed{L = a^n b^n, \; n > 0}$

↓

Not Regular Language

$S \rightarrow aA$

$S \rightarrow ab$ $\quad\quad S \rightarrow aBb$

$S \rightarrow aaAb$

$S \rightarrow aabb$ $\quad\quad S \rightarrow aaBbb$

$S \rightarrow aaaAbb$

$S \rightarrow aaabbb$

* If some in left linear & some in right linear ∴ grammer may be ~~linear~~ regular or not.

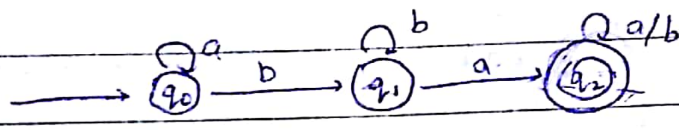
* If all are left linear (or right linear) ∴ grammer is regular

FA to Regular Grammer (NFA / DFA)

$\delta(q_i, a) \rightarrow q_j$ when $q_j \notin F$ P : $A_i \rightarrow aA_j$

$\delta(q_i, a) \rightarrow q_j$ when $q_j \in F$ P : $A_i \rightarrow aA_j \mid a$

Eg.



P : $A_0 \rightarrow aA_0$

$A_0 \rightarrow bA_1$

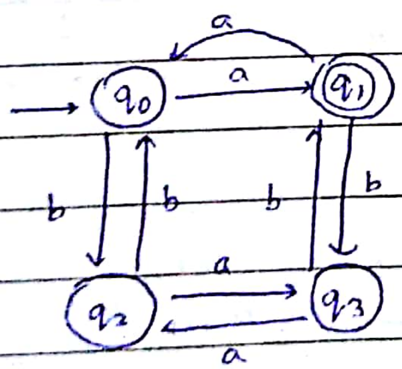
$A_1 \rightarrow bA_1$

$A_1 \rightarrow aA_2 \mid a$

$A_2 \rightarrow aA_2 \mid a$

$A_2 \rightarrow bA_2 \mid b$

Eg.



P : $A_0 \rightarrow aA_1 \mid a$

$A_0 \rightarrow bA_2$

$A_1 \rightarrow aA_0$

$A_1 \rightarrow bA_3$

$A_2 \rightarrow bA_0$

$A_2 \rightarrow aA_3$

$A_3 \rightarrow aA_2$

$A_3 \rightarrow bA_1 \mid b$

Teacher's Signature

Regular Grammar to FA

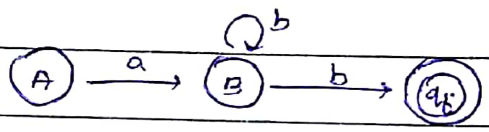
$A_1 \rightarrow aA_2$ then $\delta(q_1, a) \rightarrow q_2$

$A_1 \rightarrow a$ then $\delta(q_1, a) \rightarrow q_f$ (make new state)

$q_f \in F$

Eg. $A \rightarrow aB$

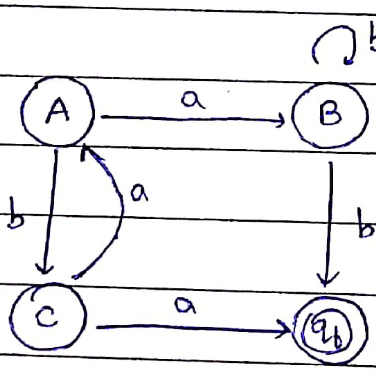
$B \rightarrow bB \mid b$



Eg. $A \rightarrow aB \mid bC$

$B \rightarrow bB \mid b$

$C \rightarrow aA \mid a$



06/02/18

Context free grammar

→ A grammar $G(V_n, \Sigma, P, S)$ is CFG if all the productions are of the forms

$A \longrightarrow \alpha$ where $\alpha \in (V_n \cup \Sigma)^*$

single non-terminal   → set of NT / T

(made less constraint)

Scanned by CamScanner

→ Finite automata can't accept n!, prime no., etc.

Eg

$S \rightarrow AB$

$A \rightarrow aaA$

$A \rightarrow \Lambda$

$B \rightarrow Bb$ string : aaab

$B \rightarrow \Lambda$

* Deriving String with help of grammar :

Derivation

left most

Right most

AaB

everytime picking left NT

everytime picking right NT

$S \rightarrow AB$

$S \rightarrow aaAB$ ⇒ aaab doesn't belong to grammar

$S \rightarrow aaaaAB$

→ aab

$S \rightarrow AB$: Left most derivation

$S \rightarrow aaAB$ $(A \rightarrow aaA)$

$\rightarrow aaB$ $(A \rightarrow \Lambda)$

$\rightarrow aaBb$ $(B \rightarrow Bb)$

$\rightarrow aab$ $(B \rightarrow \Lambda)$

→ aab Right most

$S \rightarrow AB$

$S \rightarrow ABb$ $(B \rightarrow Bb)$

$S \rightarrow Ab$ $(B \rightarrow \Lambda)$

$S \rightarrow aaAb$ $(A \rightarrow aaA)$

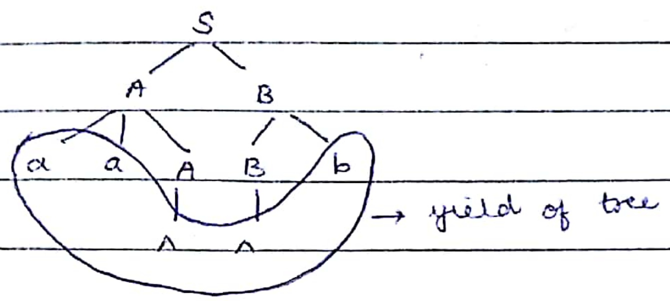
$S \rightarrow aab$ $(A \rightarrow \Lambda)$

Teacher's Signature

Scanned by CamScanner

Derivation Tree

→ For same eg.



Eg. $S \rightarrow abScB$ string : ababcbcbb

$S \rightarrow \Lambda$

$B \rightarrow bB \mid b$

$S \rightarrow abScB$

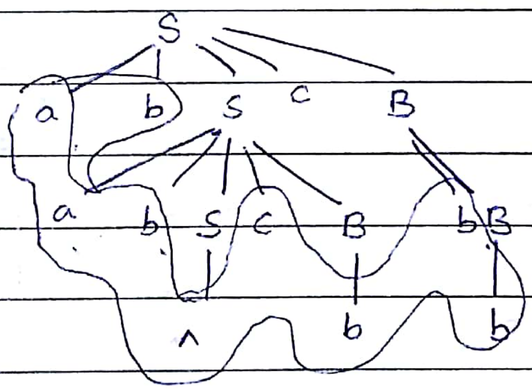
$\rightarrow ab(abScB)cB \quad (S \rightarrow abScB)$

$\rightarrow ababcBcB \quad (S \rightarrow \Lambda)$

$\rightarrow ababcbBcB \quad (B \rightarrow bB)$

$\rightarrow ababcbcbB \quad (B \rightarrow bB)$

$\rightarrow ababcbcbb \quad (B \rightarrow b)$



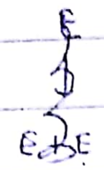
max

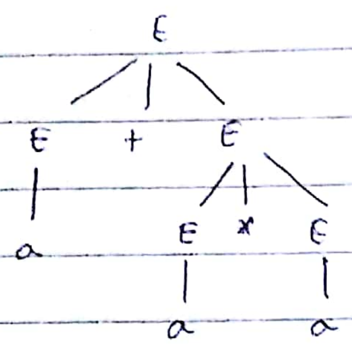
* If you have a NT at same level multiple times no of $\wedge$ → ambiguous.

<u>Ambiguous grammar</u> : For 1 ~~tree~~ string, we've more than 1 derivation tree.

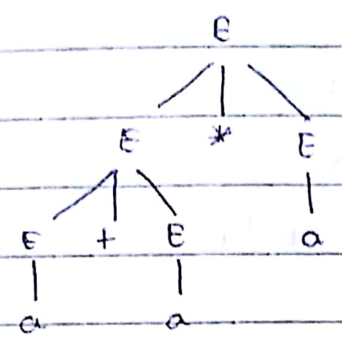
$E \rightarrow E + E \mid E * E \mid a$

String $a + a * a$





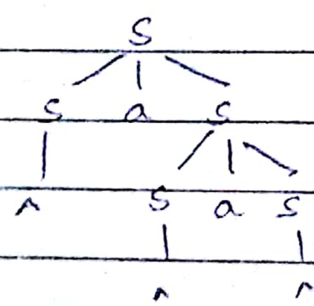
or

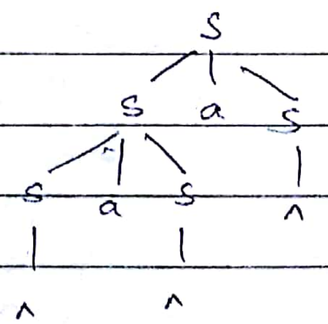


2 methods ⇒ no string path to follow → ambiguity

Eg. $S \rightarrow SaS \mid \wedge$

ambiguous for $\underline{\underline{aa}}$





Eg.

$S \rightarrow 0A$

$S \rightarrow 1B$ "001101"

$A \rightarrow 0AA$

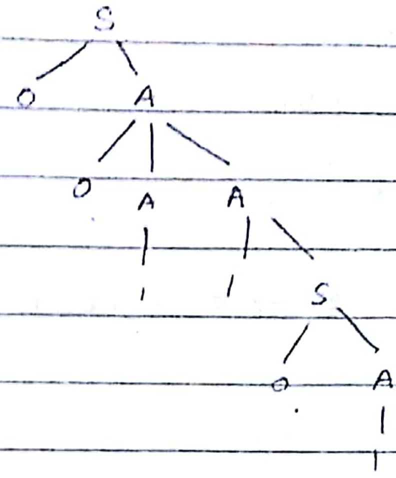
$A \rightarrow 1S$

$A \rightarrow 1$

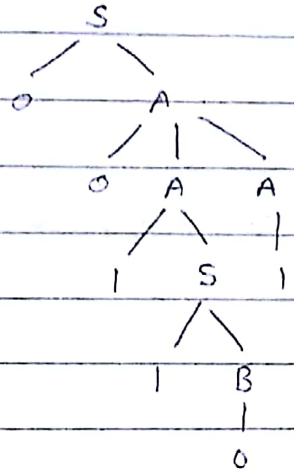
$B \rightarrow 1BB$

$B \rightarrow 0S$

$B \rightarrow 0$



OR



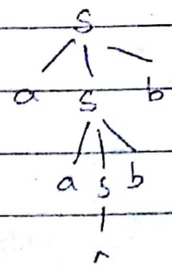
→ $L = a^n b^n, \; n \geq 0$

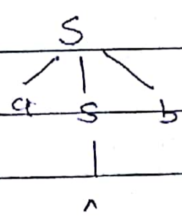
$S \rightarrow aSb \mid \wedge$

start with a ↳ end with b

→ if $n > 0$ ⇒ not empty string

$S \rightarrow aSb \mid ab$





Teacher's Signature

odd length : a/b instead of null

Eg. $L = ww^R$ $w \in \{a,b\}^*$

$\downarrow$ string, $\hookrightarrow$ reverse string

Even length

abba ⇒ we are nesting

babbab

abbbba — If a is coming in starting, it should come in the end

$S \rightarrow aSa \mid bSb \mid \wedge$

$S \rightarrow aSa$
$\rightarrow abSba$
$\rightarrow abba$

Eg. $L = wcw^R$ → c toh aayega hi

$S \rightarrow aSa \mid bSb \mid c$

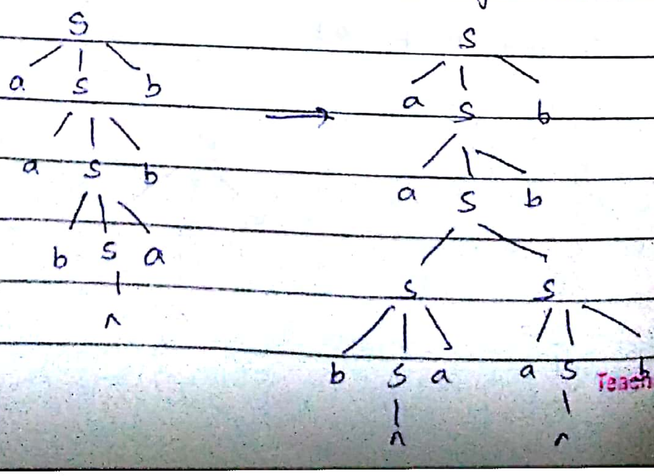
Eg. L = all string having equal no. of a & b.

whenever ∃ a, b should be there & vice versa

$S \rightarrow aSb \mid bSa \mid \wedge \rightarrow SS$

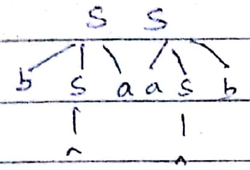
can appear in any order

aabaabbb → can't be generated



→ If following nesting property → CFG
else, not CFG

b a a b × can't be generated



9/2/17

Eg. 1.) $L = a^n b^m \quad n > m$

2.) L = all string having unequal no. of a & b

1.) $S \rightarrow aSb \mid A$ → to increase no of a

$A \rightarrow aA \mid a$ (Pattern ~~aa.. bb..~~)

2) $S \rightarrow \underbrace{aSb \mid bSa \mid SS}_{\text{equal}} \mid \underbrace{aAb \mid aBb \mid bAa \mid bBa}_{\text{unequal}}$

$A \rightarrow aA \mid a$

$B \rightarrow bB \mid b$

Eg. $L = a^{2n} b^n \quad n > 0$

$S \rightarrow aaSb \mid aab$

Eg. $L = a^n b^{m+n} c^m$:→ $\underbrace{a^n b^n}_{A} \underbrace{b^m c^m}_{B}$

$S \rightarrow AB$

$A \rightarrow aAb \mid \lambda$

$B \rightarrow bBc \mid \lambda$

Eg. $L = a^n b^{2n} c^n$ $\qquad a^n b^n b^n c^n$: Not CFG

~~$S \rightarrow aSbbSc$~~

A want to check a & b and a & c also.
I can only match 2 things

Teacher's Signatu

Eg. $L = a^n b^m c^m d^n$: Nested ✓ $m, n \geq 0$

$S \rightarrow aSd \mid A$ → after generating all a & d, do the

$A \rightarrow bAc \mid \wedge$

Eg. $L = a^m b^n c^m d^n$: Not Nested

⇒ Not C.F.G.

Eg. $L = a^n b^m$ $n \leq m+3$ $n = m$ ✓

$n = 1$ $m =$ n

$S \rightarrow aAb \mid \wedge$

Eg. $L = a^n b^m$ $2n \leq m \leq 3n$

a^2b^4 → double

a^2b^5 → b/w

a^2b^6 → triple

$S \rightarrow \underbrace{\underset{\text{double}}{aSbb} \mid \underset{\text{triple}}{aSbbb}}_{\text{comb}^n :\ \text{b/w का}} \mid \wedge$

9/2/18

<u>Closure Properties of CFL</u>

$L_1 : G_1 (P_1, S_1, \Sigma_1, V_1)$

$L_2 : G_2 (P_2, S_2, \Sigma_2, V_2)$

1) <u>Union</u> $L_1 \cup L_2$

If start from L_1

if start from L_2

G (P, S, Σ, V)

$S \rightarrow S_1 \mid S_2$

$\Sigma = \Sigma_1 \cup \Sigma_2$

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

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

Eg. $L_1 = a^n b^n c^m$ $n \geq 0, m \geq 0$

$L_2 = a^n \cdot b^{2K} c^n$ $n, K \geq 0$

$L = L_1 \cup L_2 : G$

Teacher's Signature

2) Concatenation

$L = L_1 \cdot L_2$

$S \rightarrow S_1 S_2$

1st part with L_1 ; 2nd part with L_2

3) Kleene Closure

$L = L_1^*$

$S \rightarrow SS_1 \mid \wedge$ (Repeat S_1 any no. of times & end by $\wedge$)

$S \rightarrow SS_1$
$S \rightarrow SS_1S_1$
$S \rightarrow SS_1S_1S_1 \ldots$

4) Reversal

$L_1 = a^n b^n c^m \quad n \geq 0 \quad m \geq 0$

$L = L_1^R$

$L_1 : S_1 \rightarrow AC$
$A \rightarrow aAb \mid \wedge$
$C \rightarrow cC \mid \wedge$

(only reverse prodn)

$S_1 \rightarrow CA$
$C \rightarrow Cc \mid \wedge$
$A \rightarrow bAa \mid \wedge$

5) Intersection

$L_1 = a^n b^n c^m \quad n, m \geq 0$ → a & b should be equal
$L_2 = a^m b^n c^n \quad n \geq 0, \; m \geq 0$ → b & c " "

⇓

$L_1 \cap L_2$ = a & b & c : eqn

$L = a^n b^n c^n$: Not CFL

$L_1 = a^n b^m c^m d^k \quad m, n, k \geq 0$

$L_2 = a^n b^k c^p d^n \quad n, p, k \geq 0$

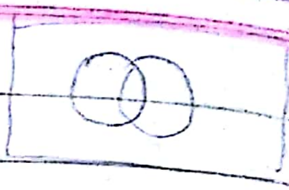
$L = a^n b^m c^m d^n$ → closed (CFL)

⇒ CFL is not closed under intersection (may / may not be CFL)

Teacher's Signature

Scanned by CamScanner

6) Complement



$\overline{L_1 \cap L_2} = \overline{L_1} \cup \overline{L_2}$

$L_1, \overline{L_1}, L_2, \overline{L_2} \rightarrow$ CFL

$\overline{L_1} \cup \overline{L_2}$: CFL

↓

may or may not be closed

But we can't say so ~~for $L_1 \cap L_2$~~

Simplification of CFL

→ Eliminate Useless Symbol

Eg. $S \rightarrow ABD$

$A \rightarrow aAB \mid c$

$C \rightarrow BC$: no string is generated (abc...) C won't be removed ⇓ end string will have C ⇒

$D \rightarrow a$

$B \rightarrow bB \mid L$

$E \rightarrow a \mid b$: None of my prodⁿ is reaching E

Useless symbol

1.) Non - Generating : C

2.) Non - Reachable : E

→ Non - Generating :

V_1 = set of all NT deriving a terminal

$V_i = V_{i-1} \cup \{ A$ where $A \rightarrow \alpha$ & $\alpha \in (\Sigma \cup V_{i-1})^* \}$

(all NT generating T)

$V_1 = \{E, B, D\}$ [not A is generating NT. would be ∵ $A \rightarrow d$]

$V_2 = \{E, B, D\} \cup \{S\}$ ⇒ Start symbol should always be there

↳ S is generating B, D which is already in V_1.

$S \rightarrow BD$

$D \rightarrow a$

$B \rightarrow bB \mid b$

$E \rightarrow a \mid b$

Eg. $S \rightarrow AB \mid CA$

$B \rightarrow BC \mid AB$

$C \rightarrow aB \mid b$

$A \rightarrow a$

$V_1 = \{A, C\}$ because it has CA

$V_2 = \{A, C\} \cup \{S\} = \{A, C, S\}$

$S \rightarrow CA$

$C \rightarrow \cancel{aB} \mid b$

$A \rightarrow a$

* Kissi case me S nhi include hota h ⇒ Empty Grammar

2/02/18

→ Non - Reachable

$W_1 = \{S\}$ can be both T & NT

$W_{i+1} = W_i \cup \{A \mid B \rightarrow A \text{ and } B \in W_i\}$

Repeat till $W_i = W_{i+1}$

$V_N' = V_N \cap W_i$

$\Sigma' = \Sigma \cap W_i$

P' = include all prod^n's which have T & NT only from W_i

Eg. $S \rightarrow aAa$

$A \rightarrow Sb \mid bCC$

$C \rightarrow aC \mid b$

$E \rightarrow a \mid b$

① First, Remove de-generating

$\cancel{W_1} = \cancel{\{S\}}$

W_2 E, C, S, A

② Non- Reachable

$W_1 = \{S\}$

$W_2 = \{S\} \cup \{a, A\} = \{S, a, A\}$

$W_3 = \{S, a, A, b, C\}$

$W_4 = \{S, a, A, b, C\} \cup \{a, C, b\} = W_4$

* Modified NT set :

$V_N' = \{S, A, C, E\} \cap W_1$

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

$\Sigma' = \{a, b\} \cap W_i$

$= \{a, b\}$

$P' = \quad S \rightarrow aAa$ {all included in W_i}

$A \rightarrow Sb$

$A \rightarrow bCC$

$C \rightarrow aE$

$C \rightarrow b$

→ $S \rightarrow Aa \mid f$

$A \rightarrow SbB$

$B \rightarrow bCC$

$C \rightarrow eD \mid Ec \mid d$

$E \rightarrow aA \mid bB$

$F \rightarrow bE \mid c$

1) de-generating :

$\{F, C\}$ ~~$\cup \{B\}$~~ S

~~$\{F, B,$~~ $= \{F, C,$

2) Remove $F \rightarrow bE \mid c$

Elimination of Null Production

W_i = find set of all nullable variable

$A \rightarrow \Lambda \rightarrow$ add A to W_1

$W_1 = \{A \mid A \rightarrow \Lambda \in P\}$

$W_{i+1} = W_i \cup \{B \mid B \rightarrow A$ and $A \in W_i\}$

Repeat till $W_i = W_{i+1}$

$A \rightarrow \Lambda$

$B \rightarrow aA$ $\xrightarrow{\Lambda}$ B: not nullable (won't generate Λ)

$B \rightarrow A$ ✓

$P' =$ if $A_1 \in W_i$ and $B \rightarrow A_1 A_2 \ldots A_n \in P$

$P' = B \rightarrow A_2 \ldots A_n \mid A_1 A_2 \ldots A_n$

: write new P including & excluding nullable variable

Q. $S \rightarrow aS \mid AB$

$A \rightarrow \Lambda \mid aD$

$B \rightarrow \Lambda \mid bDa$

$D \rightarrow b$

$W_1 = \{A, B\}$

$W_2 = W_1 \cup \{S\} = \{A, B, S\}$

S is nullable

$S \rightarrow \underset{\text{include}}{aS} \mid \underset{\text{remove}}{a}$

$S \rightarrow \underset{\text{both null}}{AB} \mid \underset{\text{remove B}}{A} \mid \underset{\text{remove A}}{B}$ (won't consider Λ)

$A \rightarrow aD$

$B \rightarrow bDa$

$D \rightarrow b$

Q. $S \rightarrow ABC$

$A \rightarrow aAb \mid \Lambda$

$B \rightarrow bC \mid CA$

$C \rightarrow aC \mid \Lambda$

$W_1 = \{A, C\}$

$W_2 = \{A, C\} \cup \{B\} = \{A, C, B\}$

$W_3 = \{A, C, B\} \cup \{S\} = \{A, C, B, S\}$

$S \rightarrow ABC \mid A \mid B \mid C \mid AB \mid BC \mid CA$

$A \rightarrow aAb \mid ab$

$B \rightarrow bC \mid b$

$B \rightarrow CA \mid C \mid A$

$C \rightarrow aC \mid a$

Teacher's Signature

Elimination of unit Production

just try the height of tree

$A \rightarrow B$ (single NT → single NT) ⇒ Unit Prod^n

$B \rightarrow C$

$C \rightarrow a$

⇒ $A \rightarrow a$, $B \rightarrow a$, $C \rightarrow a$

$A \rightarrow BC$: Not Unit Prod^n

$A \rightarrow B|C$: Unit Prod^n

Method:

Construction of $W_i(A)$ set defining all NT derived from A in unit Prod^n.

$W_1(A) = A$

for individual non-terminal

$W_{i+1}(A) = W_i(A) \cup \{B \mid A \rightarrow B \;\&\; B \in W_i\}$ (NT, NT)

Repeat till $W_i = W_{i+1}$

Construct^n of P' :

if $B \in W_i(A)$ and $B \rightarrow \alpha \in P$

then $A \rightarrow \alpha \in P'$

Eg. 1

$A \rightarrow B$ $B \rightarrow \alpha$	$\equiv$ $A \rightarrow \alpha$ $B \rightarrow \alpha$

Eg.

$S \rightarrow AB$ $\qquad W_1(S) = \{S\}$

$A \rightarrow a$ $\qquad W_1(A) = \{A\}$

$B \rightarrow C|b$ $\qquad W_i(B) = \{B, C, D, E\}$

$C \rightarrow D$ $\qquad W_i(C) = \{C, D, E\}$

$D \rightarrow E$ $\qquad W_i(D) = \{D, E\}$

$E \rightarrow a$ $\qquad W_i(E) = \{E\}$

$S \to AB$

$A \to a$

$B \to b$

$E \to a$ 1st write all which aren't unit

$D \to a \mid bC$ $D = \{E, D\}$

$C \to a \mid bC$ from D ↳ introduce all T from E ~~from D~~

$B \to a \mid bC$

Q. $A \to BC \mid B$ $W_1(A) = \{A, B\}$

$B \to e \mid bd$ $W_1(B) = \{B\}$

$C \to eC \mid d$ $W_i(C) = \{C\}$

$A \to BC \mid e \mid bd$

$B \to e \mid bd$

$C \to eC \mid d$

Procedure: Remove in this manner only!

1. Null
↓
2. Unit
↓
3. Useless
   4. Non-Generating
   5. Non-Readable

NT not in single form

Q. $S \rightarrow AB \mid a$ — $W_i\{S\} = \{S\}$ → include, which are NT in single form

$A \rightarrow b \mid B$ — $W_i\{A\} = \{A, B, C\}$

$B \rightarrow cD \mid C$ — $W_i\{B\} = \{B, C\}$ → copy all C from B

$C \rightarrow eD \mid d$ — $W_i\{C\} = \{C\}$ ②

$D \rightarrow g$ — $W_i\{D\} = \{D\}$ ①

go like this

P' : if $B \in W_i(A)$ NT

~~if B∈ Wi(A)~~ whatever prodⁿ B is having, derive that prodⁿ from A also.

Here, P' :

$S \rightarrow AB \mid a$

$A \rightarrow b \mid cD \mid eD \mid d$

$B \rightarrow cD \mid eD \mid d$

$C \rightarrow eD \mid d$

$D \rightarrow g$

add non-unit as it is

$A \rightarrow B \rightarrow C$

go with smallest set → D, C.

Normal Form

→ prodⁿ appear on specific format

→ gives that format

Chomsky Normal Form (CNF)

→ A grammar G is in 'CNF' if all the productions are in following form :

$A \rightarrow BC$ → 2 NT

or $A \rightarrow a$ → NT

eg:

$S \rightarrow AB \mid a$

$A \rightarrow b \mid cD \mid eD \mid d$

$B \rightarrow cD \mid eD \mid dC$ ⇒ have to convert in CNF

$C \rightarrow eD\,d\,B \mid a$

$D \rightarrow g \mid c \mid b$

Procedure :

① simplify grammar (already in above eg.)

② Removing intermediate terminals :

for each terminal a, introduce new NT X & a prodⁿ $X \rightarrow a$

replace a by X

$S \rightarrow AB \mid a$ $\quad X \rightarrow c \quad Y \rightarrow e \quad Z \rightarrow d$

$A \rightarrow b$

$A \rightarrow cD$ — can't replace it by D because D is generating others also

$A \rightarrow XD$ $\quad B \rightarrow XD \quad C \rightarrow YDZB$

$A \rightarrow YD$ $\quad B \rightarrow YD \quad C \rightarrow a$

$A \rightarrow d$ $\quad B \rightarrow ZC \quad D \rightarrow g \mid c \mid b$

③ Restricting no. of NT :

$A \rightarrow A_1, A_2, \ldots, A_n$

rewrite as :

$A \rightarrow A_1 Z_1$

$Z_1 \rightarrow A_2 Z_2$

$Z_2 \rightarrow A_3 Z_3$

⋮

$Z_{n-2} \rightarrow A_{n-1} A_n$

above eg: We have $C \rightarrow YDZB$ ⇒ $C \rightarrow YC_1$

$C_1 \rightarrow DC_2$

$C_2 \rightarrow ZB$

Teacher's Signature

Eg. $A \rightarrow aAcB \mid cDc$

$B \rightarrow \lambda \mid aA$

$D \rightarrow B \mid aDA \mid b$

Simplified grammar.

a) NULL :

$\{B, D\}$

$A \rightarrow aAcB \mid aAc$ $\{A\} = \{A\}$

$A \rightarrow cDc \mid cc$ ϕ

$B \rightarrow aA$ $\{B\} = \{B\}$

$D \rightarrow B \mid aDA \mid b \mid aA$ $\{D\} = \{D, B\}$

b) Unit :

$A \rightarrow aAcB \mid aAc$

$A \rightarrow cDc \mid cc$

$B \rightarrow aA$

$D \rightarrow \cancel{B} aDA \mid b \mid aA$

c) Useless

$\{D, A, B\}$ all useful

$\rightarrow$ $X \rightarrow a$ $Y \rightarrow c$ $Z \rightarrow e$

$A \rightarrow XAYB$

$A \rightarrow XA_2$ $A_2 \rightarrow AA_3$ $A_3 \rightarrow YB$

~~$A \rightarrow XAZ$~~ $A \rightarrow XA_1$ $A_1 \rightarrow AY$

$B \rightarrow XA$

$D \rightarrow XD_1$ $D_1 \rightarrow DA$

$D \rightarrow XA$

$D \rightarrow b$

Teacher's Signature

$A \rightarrow YZ$ $A \rightarrow YZ$

$A \rightarrow YDZ$ $A \rightarrow YA_1$ $A_1 \rightarrow DZ$

Greibach Normal Form (GNF)

A grammar G is in GNF if all the prod^ns are of form

$A \rightarrow a\alpha_1\alpha_2 \ldots \alpha_n \mid a$: start with T followed by any no. of NT

$a \in \Sigma$

$\alpha_1, \alpha_2, \ldots \alpha_n \in NT$

Procedure :

① Simplify grammar

② Remove intermediate terminals (use CNF Rule ②)

③ i) If $A \rightarrow B\alpha \in P$ then $B \rightarrow \beta \in P$ then $A \rightarrow \beta\alpha \in P'$

ii.) if $A \rightarrow A\alpha \mid \beta$

then

$A \rightarrow \beta A' \mid \beta$

$A' \rightarrow \alpha A' \mid \alpha$

$A \rightarrow A\alpha$: left

$\rightarrow A\alpha\alpha$

$\rightarrow A\alpha\alpha\alpha$

$\rightarrow \beta\alpha\alpha\alpha$

Eg.

$S \rightarrow AB \mid b$

$A \rightarrow cBd \mid gF$

$F \rightarrow a$

$B \rightarrow bB \mid c$

$\Rightarrow$

$S \rightarrow AB \mid b$ $\quad X \rightarrow d$

$A \rightarrow cBX \mid gF$ (haven't removed c because we've to again add it later on)

$F \rightarrow a$

$B \rightarrow bB \mid c$

$\rightarrow$

$S \rightarrow cBXB \mid gFB \mid b$ (copy prod^ns of A)

$A \rightarrow cBX \mid gF$

$B \rightarrow bB \mid c$

$F \rightarrow a$

already in GNF form

Ques.

$S \to Ab \mid c$

$A \to Sa \mid d$] → already simplified) → start with anyone, you'll get ans automatically

(2) → $X \to a$ $Y \to b$

$S \to AY \mid c$

$A \to SX \mid d$

recursion

(3) → $\underset{A}{S} \to \underset{A}{S}\underset{\alpha\,(NT)}{XY} \mid \underset{\beta}{dY} \mid \underset{\beta}{c}$ (copy prodn of A)
other prod's : β

B → $S \to dY \mid c$ write β as it is

$S \to dYS' \mid cS'$ add S'

$S' \to XYS' \mid XY$ (apply X prodⁿ to get GNF)

~~$S' \to aYS' \mid aY$~~

α → $A \to SX \mid d$

$A \to dYX \mid cX \mid dYS'X \mid cS'X \mid d$

Quiz

(1) $L = a^l b^m$ $|l - m| = 2$

$l - m = 2$ or $l + m = 2$

$a^m b^{m+2}$ or $a^{m+2} b^m$

⇓   ⇓

$a^m b^m b^2$ or a^m

② a) $L = a^{n+2}\, b^{m}\, c^{n+m}$

$= a\,a\,a^n\, b^m\, c^n\, c^m$

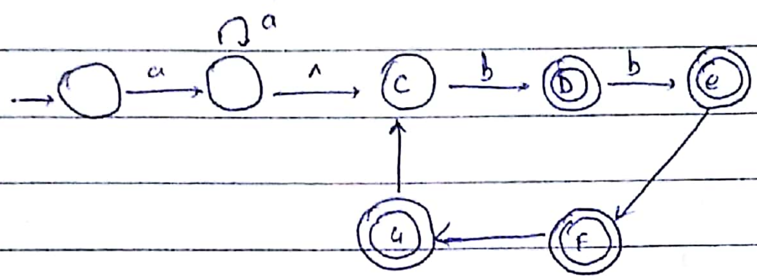
$S \rightarrow aaA$

$A \rightarrow aAc \mid \wedge \mid B$

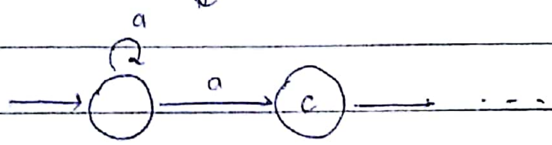
$B \rightarrow bBc \mid \wedge$

b) $L = a^m b^n$ $m > 0$ & n is not multiple of 5

~~S → aSb | aSbb | aSbbb | aSbbbb~~



⇓



$A \rightarrow aA \mid aC$

$C \rightarrow bD \mid b$ $\qquad G \rightarrow bC$

$D \rightarrow bE \mid b$

$E \rightarrow bF \mid b$

$F \rightarrow bG \mid b$

c) $L = a^m b^n c^k$ $\quad |m-n| = k$ $\quad m - n = -k$

$m = n+k$ or $m = n-k$

$n = m+k$

⇓ $\qquad$ ⇓

$a^k a^n b^n c^k$ $\qquad a^m b^m b^k c^k$

$S \rightarrow S_1 \mid S_2$

$S_1 \rightarrow aS_1c \mid A$

$A \rightarrow aAb \mid bc$

$S_2 \rightarrow CD$

$C \rightarrow aCb \mid \wedge$ $\qquad D \rightarrow bDc \mid \wedge$

Teacher's Signature

(3) $S \to bSe \mid PQR$

$P \to bPc \mid \Lambda$

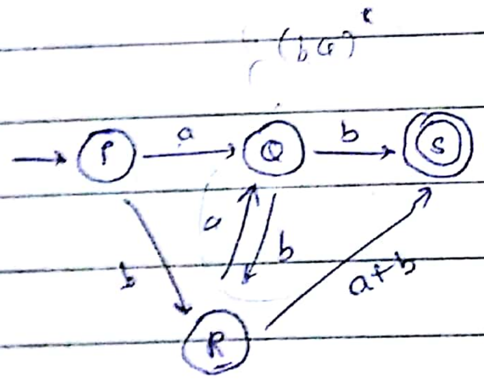
$Q \to cQd \mid \Lambda$

$R \to dRe \mid \Lambda$

$$b^{n+k} c^{k+m} d^{m+p} e^{n+p}$$

→ shortest = ~~BI~~ bcde

(4)



→ $a(ba)^* b + b(ab)^*(a+b)$