3. Nothing else in RE.

$\boxed{r^+ = rr^*}$

- **Regular Language** – Language generated by RE is called by RL.

Q/ Let r, s, t are RE. Which of following is TRUE.

(i) $(r+s)^* = r^* s^*$

(ii) $r(s+t) = rs+t$

(iii) $(r+s)^* = r^* + s^*$

(iv) $(rs+r^*)^* r = r(sr+r)^*$

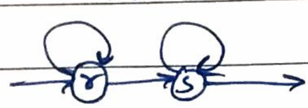
(v) $(r^* s)^* = (rs)^*$

(i) $(r+s)^*$ $\qquad\qquad$ $r^* s^*$

sr ✓ $\qquad\qquad$ sr ✗

$(r+s)(r+s)$

sr $\qquad\qquad$ NOT TRUE.



(ii) $r(s+t)$ $\qquad\qquad$ $rs+t$

rt ✓ $\qquad\qquad$ rt ✗

t ✗ $\qquad\qquad$ t ✓ $\qquad$ NOT TRUE

(iii) $(r+s)^* = r^* + s^*$ $\qquad$ NOT TRUE

rsr ✓ $\qquad$ rsr ✗

rs ✓ $\qquad$ rs ✗

$rsrr$

$r s^*$

(iv) $(rs+r^*)^* r$ $\qquad\qquad$ $r(sr+r)^*$ $\qquad$ TRUE

(v) $(r^* s)^*$ $\qquad\qquad$ $(rs)^*$

s ✓ $\qquad\qquad$ s ✗

rrs ✓ $\qquad\qquad$ rrs ✗

Q. which REs are equivalent.

(i) $(00)^*(\Lambda+0)$

(ii) $(00)^*$

(iii) 0^*

(iv) $0(00)^*$

~~(a)~~ (i) & (ii)

~~(b)~~ (ii) & (iii)

(c) (i) & (iii)

~~(d)~~ (iii) & (iv)

Q. RE $0^*(10^*)^*$ denotes the same set as

$\lambda(1\lambda)^*(100)^* = 1100$

1100

$(110)^* 1 = 1101$

(a) $(1^*0)^*1^*$

(b) $0(0+10)^*$

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

(d) None of these

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

which of the foll^n defined the same language as R.

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

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

(c) $(a(ba)^*(a+bb) + b(ab)^*(b+aa))(a+b)^*$

(d) $(a(ba)^*(a+bb) + b(ab)^*(b+aa))(a+b)^+$

* **Theorem**:- Every finite language is Regular language.

no. of elements finite

Regular expression

$L = \{a, b, c, d, bc\}$

$RE = a+b+c+d+bc$

- Language Associated with RE

1. Language $(r_1 r_2) = L_1 L_2$
2. Language $(r_1 + r_2) = L_1 + L_2$
3. Language $(r_1^*) = L_1^*$

FINITE AUTOMATA (FA)

It consists of 3 things-

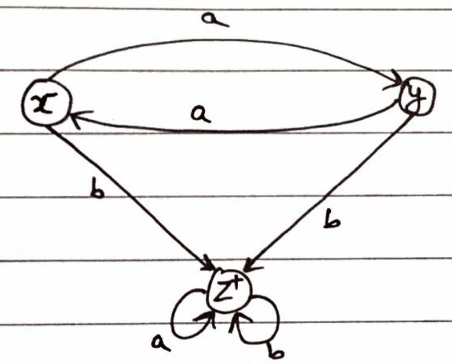
i) A finite set of states → Start state, Final state, Simple state

ii) An alphabet Σ of possible input letters.

iii) A finite set of transitions that tell for each state and for each letter of the input alphabet which state to go next.

(incoming arrow) Start state

final state (double circle)

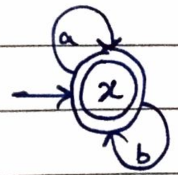


1. $\Sigma = \{a, b\}$
2. States, $Q = \{x, y, z\}$

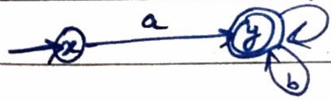
− means start state
+ means final state

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

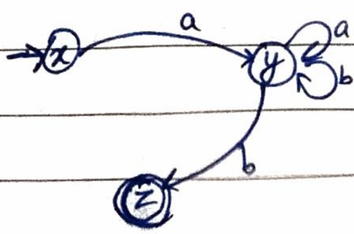
Q) $RE = (a+b)^*$ contains Λ so start state should be final state



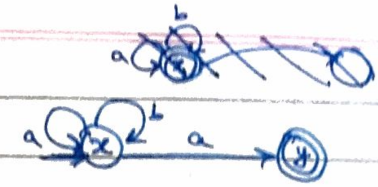
Q) $RE = a(a+b)^*$



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



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



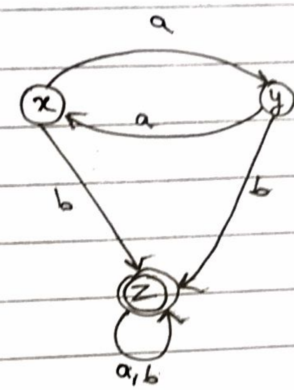
FA
↳ DFA (Deterministic FA)
↳ NFA (Non-deterministic FA)

*- FA is used acceptor or language recognizer.

Transition Function

$\delta(\text{state}, \text{symbol}) = \text{state}$

eg:- $\delta(y, b) = z$

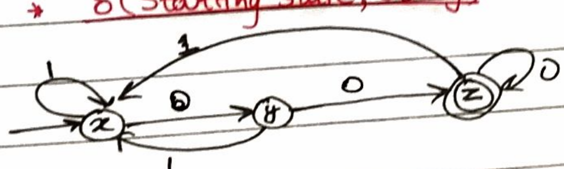


Transition Table

State \ Symbol	a	b
→x	y	z
y	x	z
(z)	z	z

* δ(Starting state, String)

Q)



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

a) all string begin with either 0 or 1 — $L_a = \{0, 1, 00, 01, 10, 11, \ldots\}$ FALSE (FA not accepting all string)

b) end with 0 — $L_b = \{0, 00, 10, 010, 110, \ldots\}$ FALSE

c) end with 1 — $L_c = \{1, 01, 11, 011, 001, \ldots\}$ FALSE

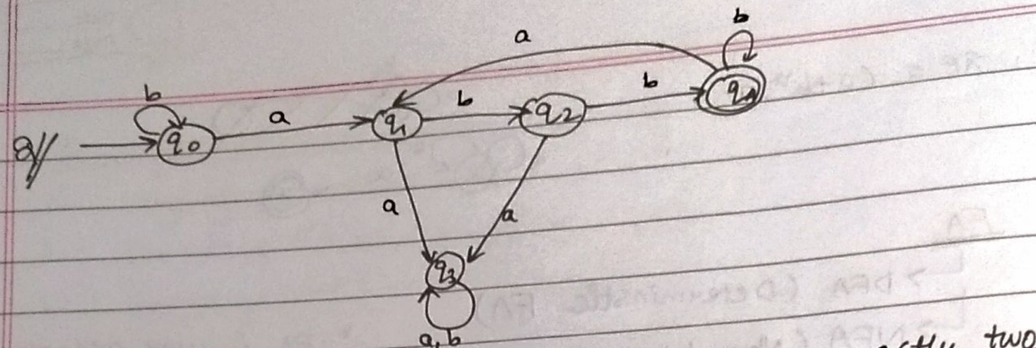
d) contains the substring 00. — $L_d = \{00, 000, 100, 1100, 1001 \ldots\}$ FALSE

e) end with 00 — $L_e = \{00, 100, 000, 1100, 1000, \ldots\}$ TRUE

RE
⇕
FA ⟷ Transition Table

⇒ language recognized by all these is Regular language.

Q/



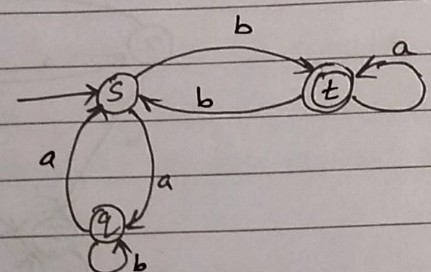
x a) { $w \in \{a,b\}^*$ / every a in w followed by exactly two b's } abbb accepted by FA but have 3 b's.

✓ b) { $w \in \{a,b\}^*$ / every a in w followed by atleast 2 b's }

x c) { / w contains substring 'abb' } abbba not accepted by FA

x d) { / w does not contain 'aa' as substring } contains ab but not accepted by FA

Q/



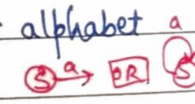
✓ u = abbaba

v = bab

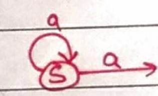
w = aabb

Finite Automata

↳ DFA – For each state and for each symbol of its input alphabet exactly one edge $\bar{c}$ that symbol leaving the state.

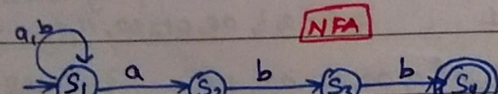


↳ NFA – No restriction on the label on edge

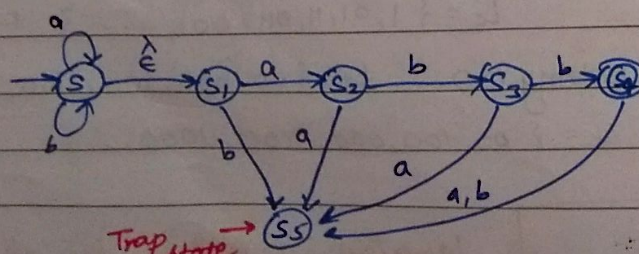


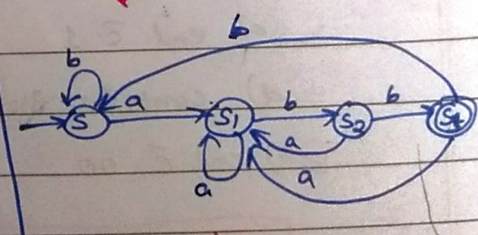
(a/b)* abb or (a+b)* abb

NFA



DFA

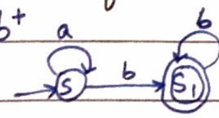




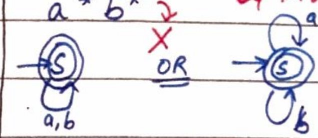
(Actual) DFA
Minimal

* Constructing NFA from RE

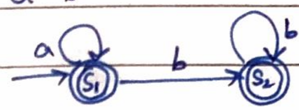
1. $a^* b^+$



$a^* b^* \not\Rightarrow$ becoz ababb will be accepted by Finite automata but ~~ababb~~ $abbb \notin a^*b^*$

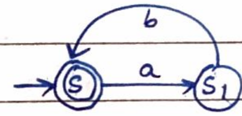


2. $a^* b^*$



3. $(ab)^*$

OR



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

$S = abc$ string $|S| = 3$

$S = \wedge$ or $\in$ $|S| = 0 \rightarrow$ NULL string

$L = \{a, aa, aba, \ldots\ldots\}$ set of strings

$L = \phi \rightarrow$ NULL ~~string~~ language $|L| = 0$ i.e., language containing ~~NULL~~ / No string

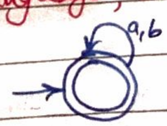
$L \subseteq \Sigma^*$ → Universal language

$L = \{\wedge\}$ is not a NULL language $|L| = 1$

LANGUAGE
- $L \neq \phi$
  - finite
  - infinite
- $L = \phi$ Empty/NULL

Finite Automata in which every state is final state accepts Universal language, Σ^*

eg:-

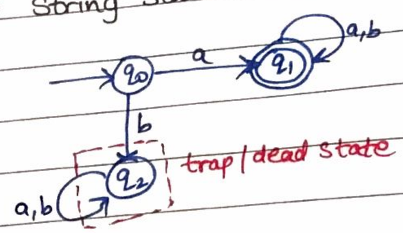


Σ^*

Finite Automata in which every state is non-final state accepts null language.

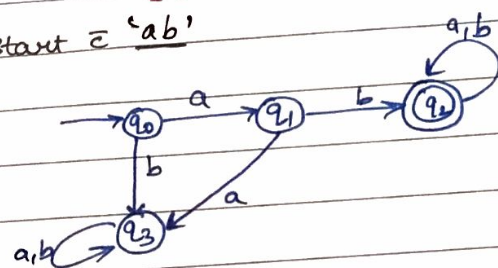
eg:- → ○ → ○ → ϕ

Example – • String start $\bar{c}$ 'a' over $\Sigma = \{a, b\}$

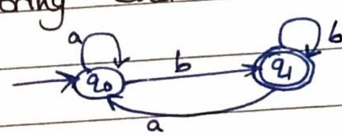


trap / dead state

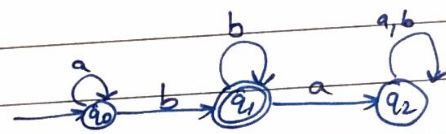
• start $\bar{c}$ 'ab'



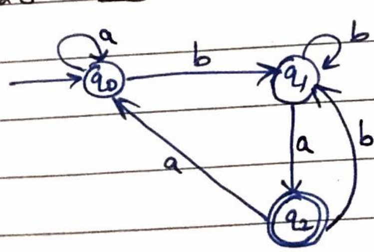
• String ends $\bar{c}$ 'b'



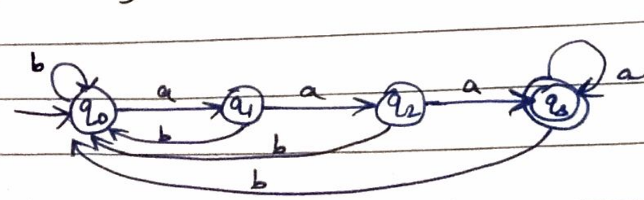
OR



• ends $\bar{c}$ 'ba'

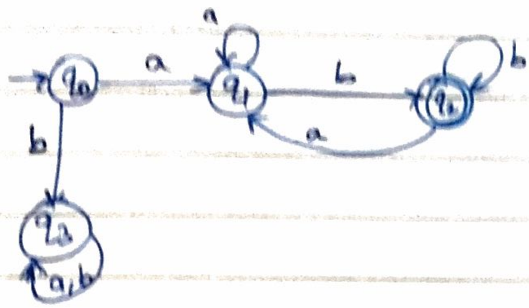


• strings ending $\bar{c}$ 'aaa'



* Only those strings comes in language ~~those~~ which satisfies ~~both~~ the rules of language as well as which are accepted by Finite Automata.

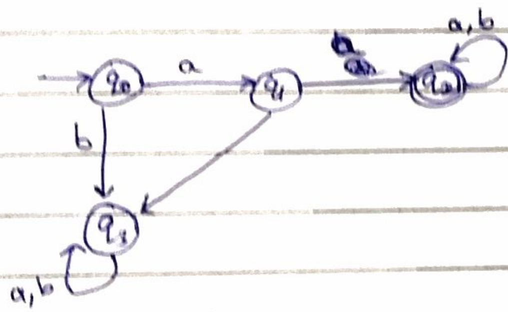
- Start $\bar{c}$ 'a' & end $\bar{c}$ 'b'



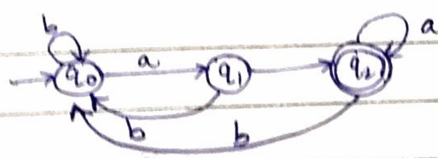
abab

- In DFA(FA), each state have exactly n(the no. of symbols) outgoing edges.

- Start $\bar{c}$ 'aa'   aaX

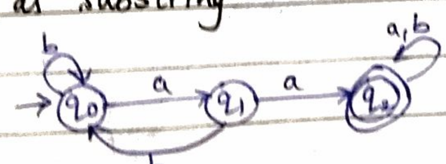


- end $\bar{c}$ 'aa'   Xaa

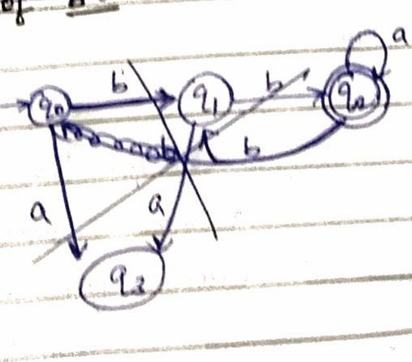


aaaba ×

- 'aa' as substring   XaaY   babaaba

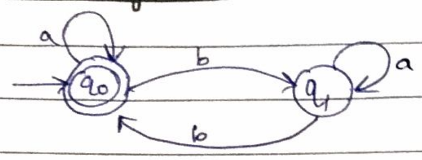


- even no. of b's



Minimal DFA — · DFA $\bar{c}$ minimum no. of states.
· always unique.

- **Even no. of b's** $b = 0, 2, 4, \ldots$



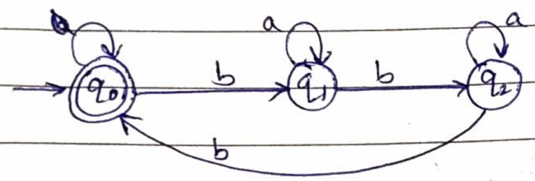
$r = 0$ $r = 1$

mod 2

Λ, a, aa, abba, abb, abab

no. of states → no. specified in ques.
↓
Find mod(2) of the b's (or any symbol).
Then remainder will give the final state.

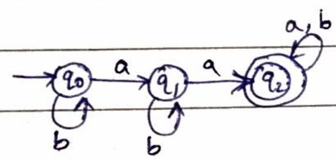
- **no. of b's divisible by 3** (mod 3)



$r=0$ $r=1$ $r=2$

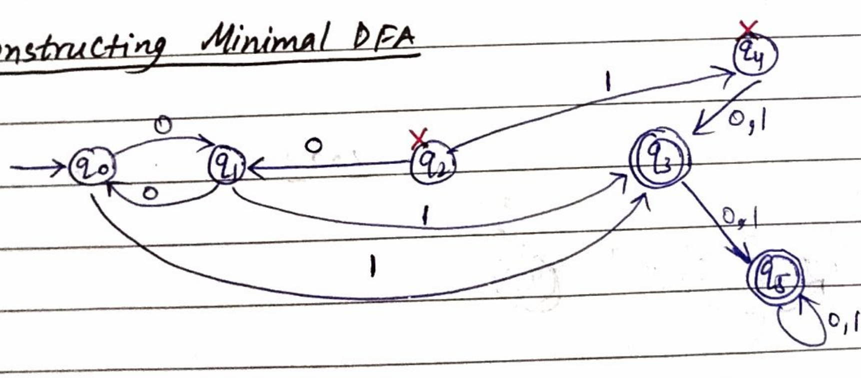
ababab

- **Contain atleast 2a's**



aaa aab
baa baba
abba

→ **Constructing Minimal DFA**



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

- Remove unreachable states. (q_2, q_4)
- Construct transition table.

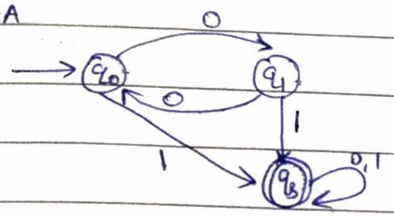
	0	1
$\rightarrow q_0$	q_1	q_3
q_1	q_0	q_3
$*q_3$	$\cancel{q_5}$ q_3	$\cancel{q_5}$ q_3 →
$*q_5$	q_5	q_5 → remove q_5

(equivalent states)
Check for that ~~final~~ states which have same destination states for same symbols.

Date

Page

- Draw FA



set of symbols

starting symbol

set of final state

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

FA

set of all state

set of all transition

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

$L(A) \leftarrow$ language represented by A.

→ **Operation on FA/ Regular Language**

- Union 'U'
- Intersection '∩'
- Complement '—'

• **Theorem:-** $A = (Q, \Sigma, \delta, q_0, F)$

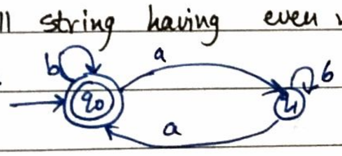
$\bar{A} = (Q, \Sigma, \delta, q_0, Q - F)$ } make every non-final state to final & vice-versa.

and corresponding language is

$$\boxed{L(\bar{A}) = \Sigma^* - L(A)}$$

Example:- all string having even no. of a's.

A =

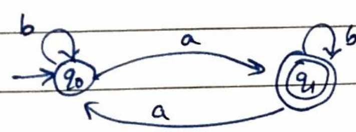


$A = (Q, \Sigma, \delta, q_0, F)$

$F = \{q_0\}$

$Q = \{q_0, q_1\}$

$\bar{A}$ =



$F_1 = \{q_1\}$

$\bar{A} = (Q, \Sigma, \delta, q_0, F_1)$

• **Product of Automata**

$L(A_1 \times A_2) = L(A_1) \cap L(A_2)$ ← Intersection

$A_1 = (Q_1, \Sigma, \delta_1, q_1, F_1)$ ← no. of a's multiple of 2

$A_2 = (Q_2, \Sigma, \delta_2, q_2, F_2)$ ← no. of b's multiple of 3.

$A = A_1 \times A_2$

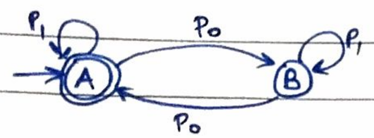
→ $A = (Q, \Sigma, \delta, q_0, F)$

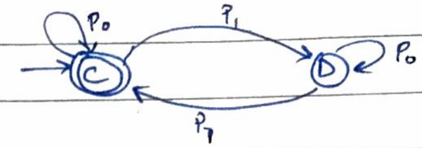
any states

$(r_1, r_2).a = (r_1.a, r_2.a)$

$Q = Q_1 \times Q_2$

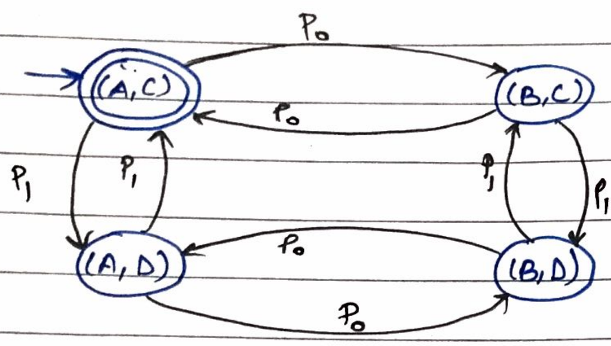
$q_0 = (q_1, q_2)$ $F = F_1 \times F_2$





$Q_1 = \{A, B\}$ $\qquad$ $Q_2 = \{C, D\}$

$Q = Q_1 \times Q_2 = \{(A,C), (A,D), (B,C), (B,D)\}$



$F = F_1 \times F_2 = \{A\} \times \{C\}$
$= \{(A,C)\}$

$q_0 = (q_1, q_2) = (A,C)$

Transitions -

$\delta(A,C) \cdot P_0 = (\delta(A \cdot P_0), \delta(C \cdot P_0)) = (B, C)$

$\delta(A,C) \cdot P_1 = (\delta(A, P_1), \delta(C, P_1)) = (A, D)$

$\delta(B,C) . P_0 = (\delta(B, P_0), \delta(C, P_0)) = (A, C)$

$(B,C), P_1 = (B, P_1), (C, P_1) = (B, D)$

$(A,D) . P_0 = (A, P_0), (D, P_0) = (B, D)$

$(A,D) . P_1 = (A . P_1), (D . P_1) = (A, C)$

$(\cancel{A,B}\ B,D) . P_0 = (B . P_0), (D . P_0) = (A, D)$

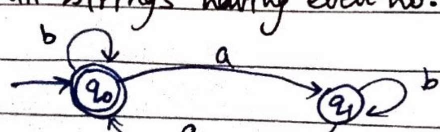
$(B,D) . P_1 = (B . P_1), (D . P_1) = (B, C)$

Transition Table

	P_0	P_1
(A,C)	(B,C)	(A,D)
(B,C)	(A,C)	(B,D)
(A,D)	(B,D)	(A,C)
(B,D)	(A,D)	(B,C)

Q/ FA of all strings having even no. of a and no. of b's is multiple of 3

→



even no. of a's

b's multiple of 3

$Q = Q_1 \times Q_2 = \{(q_0,s_0),(q_0,s_1),(q_0,s_2),(q_1,s_0),(q_1,s_1),(q_1,s_2)\}$

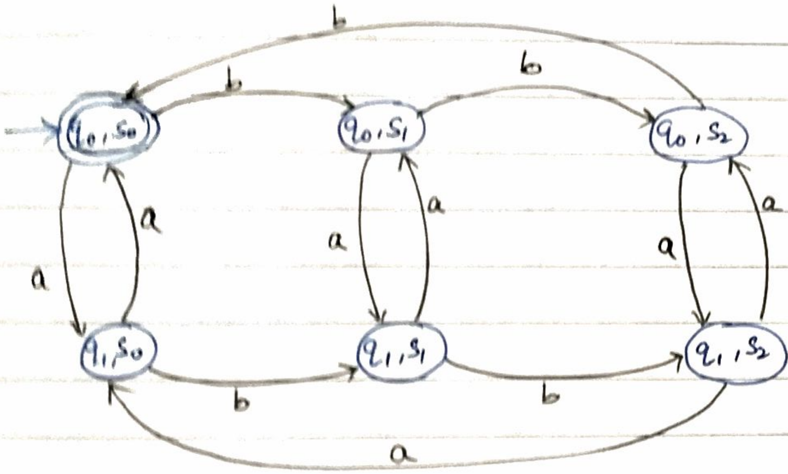
$F = F_1 \times F_2 = \{q_0\} \times \{s_0\} = \{(q_0,s_0)\}$

$q_0 = (q_0, s_0)$

Transitions:-

$(q_0,s_0).a = (q_0.a),(s_0.a) = (q_1,s_0)$

$(q_0,s_0).b = (q_0.b),(s_0.b) = (q_0,s_1)$



Q/ no. of a's multiple of 2 or no. of b's multiple of 2.
Union

P_1

P_2

$L(P_1 \oplus P_2) = L(P_1) \cup L(P_2)$ ← Union

$P_1 \oplus P_2 = P = (Q, \Sigma, \delta, q_0, F)$

$Q = Q_1 \times Q_2$ $\quad q_0 = (q_1, q_2)$

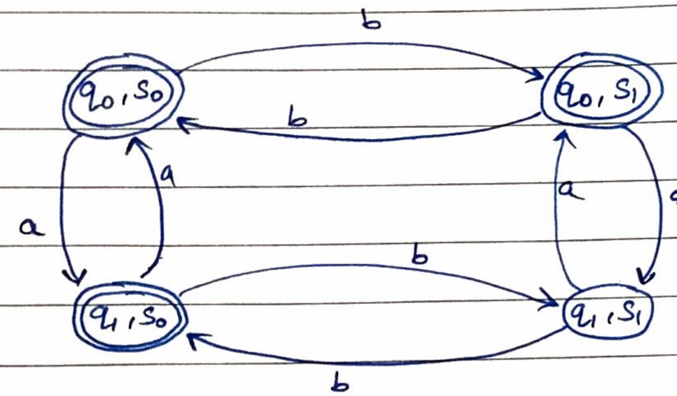
$(q_i, q_j) \in F$ iff $q_i \in F_1$ or $q_j \in F_2$

$P = (Q, \Sigma, \delta_P, q_0, F)$

$Q = (Q_1 \times Q_2) = \{(q_0, s_0), (q_0, s_1), (q_1, s_0), (q_1, s_1)\}$

$q = (q_0, s_0)$

$F = \{(q_0, s_0), (q_0, s_1), (q_1, s_0)\}$



(closure property)

NOTE:- If $L(A_1)$ & $L(A_2)$ is a Regular language

Then,

1. $L(\bar{A}_1)$ is R.L.
2. $L(A_1) \cup L(A_2)$ is R.L.
3. $L(A_1) \cap L(A_2)$ is R.L.

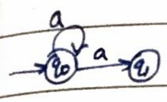
Group :- set operator

- A $(G, *)$ is a simplest algebric structure is called Group if it satisfy
  1. Closure property $(\forall\, a, b \in G \Rightarrow a * b \in G)$
  2. Associative property
  3. Existence of ~~??~~ identity
  4. Existence of inverse element.

If 5. Commutative property is also followed $\Rightarrow$ Abelian Group.

Conversion NFA to DFA

NFA



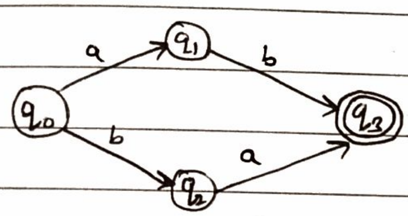
Λ-NFA (ε-NFA)

(NFA containing Λ)

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

where $\delta : Q \times \Sigma \to \underline{2^Q}$ → power set of Q

Λ-NFA $\delta = Q \times (\Sigma \cup \{\Lambda\}) \longrightarrow 2^Q$

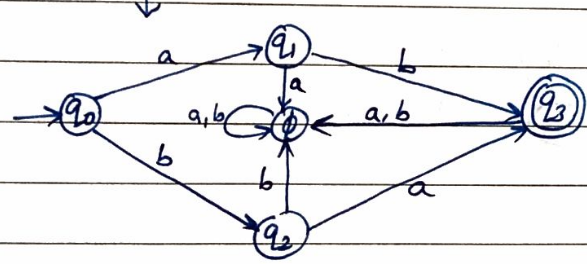


NFA

	a	b
$\to q_0$	q_1	q_2
q_1	ϕ (–)	q_3
q_2	q_3	ϕ or (–)
$*q_3$	ϕ	ϕ

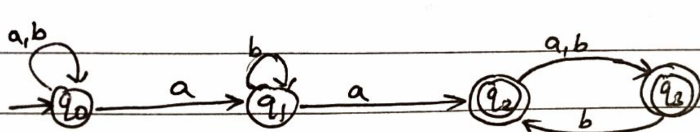
$\phi \to$ trap/dead state

DFA



(From transition table)

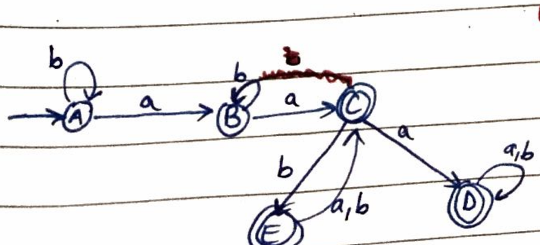
Q//



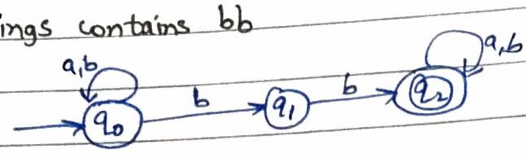
NFA → DFA

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

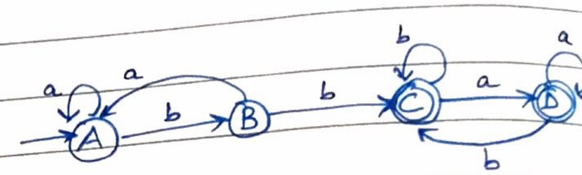
	a	b
$\to q_0$ A	$\{q_0, q_1\}$ B	q_0 A
$\{q_0, q_1\}$ B	$\{q_0, q_1, q_2\}$ C	$\{q_0, q_1\}$ B
$*\{q_0, q_1, q_2\}$ C	$\{q_0, q_1, q_2, q_3\}$ D	$\{q_0, q_1, q_3\}$ E
$*\{q_0, q_1, q_2, q_3\}$ D	$\{q_0, q_1, q_2, q_3\}$ D	$\{q_0, q_1, q_3, q_2\}$ D
$*\{q_0, q_1, q_3\}$ E	$\{q_0, q_1, q_2\}$ C	$\{q_0, q_1, q_2\}$ C



Q/ all strings contains bb



NFA



DFA

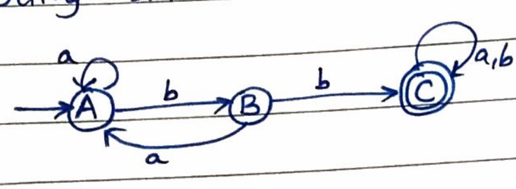
	a	b
$\rightarrow q_0$ A	q_0 A	$\{q_0, q_1\}$ B
$\{q_0, q_1\}$ B	$\{q_0$ A	$\{q_0, q_1, q_2\}$ C
* $\{q_0, q_1, q_2\}$ C	$\{q_0, q_2\}$ D	$\{q_0, q_1, q_2\}$ C
* $\{q_0, q_2\}$ D	$\{q_0, q_2\}$ D	$\{q_0, q_1, q_2\}$ C

	a	b
A	A	B
B	A	C
* C	D	C
~~* D~~	~~D~~	~~C~~

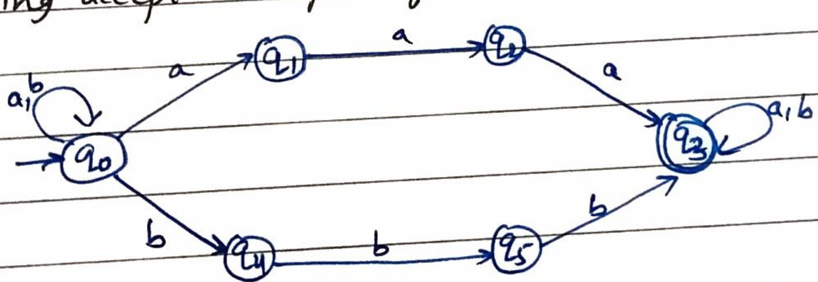
C ≠ D are equivalent States

Converting DFA to Minimal DFA

∴



Q/ String accept all triplet of letters i.e., 'aaa' or 'bbb' as substring



	a	b
$\rightarrow q_0$	$\{q_0, q_1\}$	$\{q_0, q_4\}$
$\{q_0, q_1\}$	$\{q_0, q_1, q_2\}$	$\{q_0, q_4, q_5\}$
$\{q_0, q_4\}$	$\{q_0, q_1\}$	$\{q_0, q_4, q_5\}$
$\{q_0, q_1, q_2\}$	$\{q_0, q_1, q_2, q_3\}$	~~$\{q_0$~~ q_0
$\{q_0, q$		

closure properties –

If L_1 & L_2 are regular language then,

$L_1 + L_2$

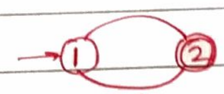
$L_1 L_2$

& L_1^* are also regular language.

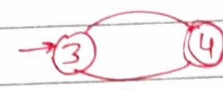
regular language
= accepted by finite automata
= transition graph

Let r_1 is regular ~~exparsi~~ expression & r_2 is also RE.

$\Leftrightarrow$ $L_1(r_1)$ & $L_2(r_2)$ is R.L.



$L_1 \leftarrow F_1$



$F_2 \to L_2$

r_1 & r_2 are RE

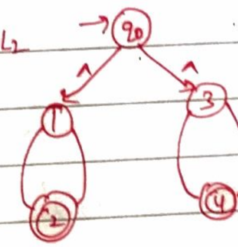
$\Rightarrow$ $r_1 + r_2$ is RE

$\Rightarrow$ $r = r_1 + r_2$ is RE

$\Rightarrow$ $L(r) = L(r_1 + r_2)$ is ~~RE~~ RL.

$\Rightarrow$ $L(r_1) + L(r_2)$ is RL

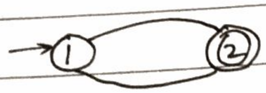
$\Rightarrow L_1 + L_2$



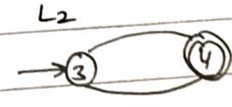
$A = \{a^p : p \text{ is prime no.}\}$ ✗ RE $\overset{\times}{(RL)}$

$B = \{a^n : n \text{ is even}\}$ ✓ RE (RL)

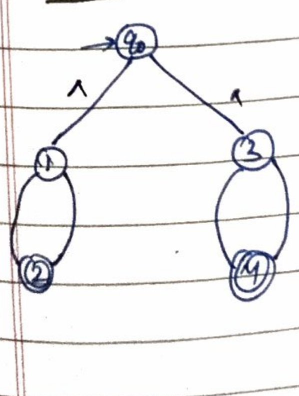
L_1



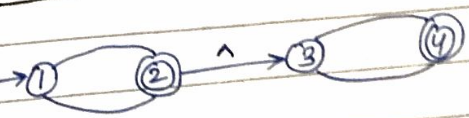
L_2



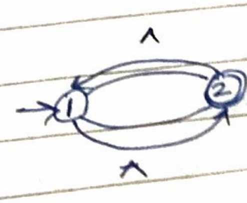
$L_1 + L_2$



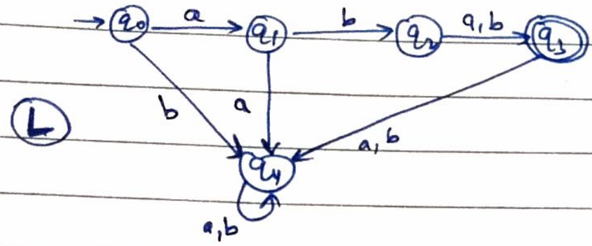
$L_1 L_2$



L_1^*



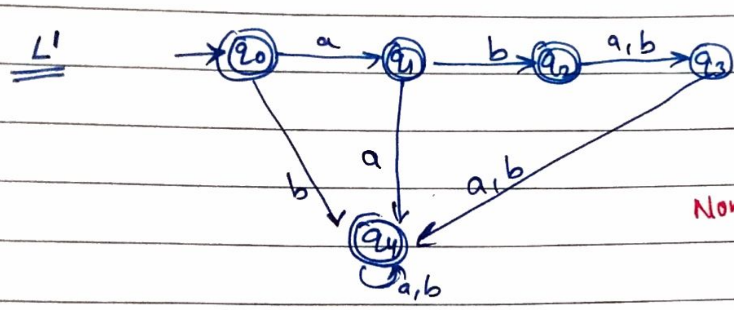
$L = \{aba, abb\}$



$\boxed{(L')' = L}$

$L' = \Sigma^* - L$

Set of all languages over Σ



Final → Non-final

Non-final → Final.

Theorem:- If L_1 & L_2 are RL then, $L_1 \cap L_2$ is also RL

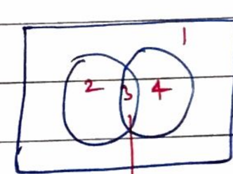
let L_1 & L_2 are RL

$\Rightarrow$ L_1' & L_2' is RL

$\Rightarrow$ $(L_1' + L_2')$ is RL

$\Rightarrow$ $(L_1' + L_2')'$ is RL

$\Rightarrow$ $L_1 \cap L_2$ is RL.



$L_1 \cap L_2 = (L_1' + L_2')'$

$L_1 = 3, 2$ $\quad L_1' = 1, 4$

$L_2 = 3, 4$ $\quad L_2' = 1, 2$

$L_1' + L_2' = 1, 2, 4$

PUMPING LEMMA (language in not Regular)

Let L be any Regular language that has infinitely many words. Then there exists some three strings x, y, z (where y is not the null string) s.t. all the strings of forms

xy^nz, $n = 0, 1, 2, 3, \ldots$

are words in L.

Q $L = \{a^n b^n, \; n = 0, 1, 2, 3, \ldots\}$

let $w = \underset{a}{x} \; \underset{ab}{y} \; \underset{b}{z}$

as, $xyz \in L$

so, Acc. to lemma $xy^2z \in L$

Take $xy^2z = \underbrace{a}_{x}\underbrace{ab}_{y}\underbrace{ab}_{y}\underbrace{a}_{z} \notin L$

$\because xyz \in L$ but $xy^2z \notin L$ $\therefore$ not a regular language.

Q/ $L = \{a^n b a^n\}$ $n = 0, 1, 2, \ldots$ $\quad \wedge b \wedge \quad \wedge b^2 \wedge$

Let $w = \begin{matrix} x & y & z \\ a & b & a \end{matrix}$

$xyz = w = aba \in L$

$\therefore xy^2z = ab^2a = \underbrace{a}_{x}\underbrace{b}_{y}\underbrace{b}_{y}\underbrace{a}_{z} \notin L$ $\quad \therefore$ not RL.

Regular Expression

$\Updownarrow$

FA

$\Updownarrow$

R.L.

$w = xyz$ such that

1. $y \neq \wedge$ or $y \neq \epsilon$
2. $|xy| \leq$ no. of states
3. $\forall n \geq 0$, $xy^nz \in L$

Q/ $L = \{a^p : p$ is a prime no.$\}$

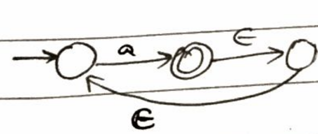
$w = \begin{matrix} x & y & z \\ a & a & a \end{matrix}$

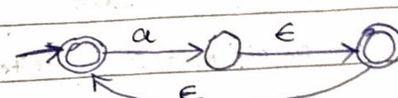
$xyz = w = aaa \in L$

$\therefore xy^2z = a(aa)a = a^4 \notin L$ as 4 is non-prime.

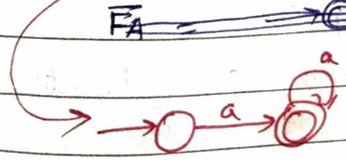
non-regular language

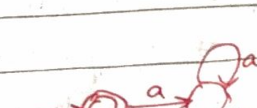
Q/

FA  a^+

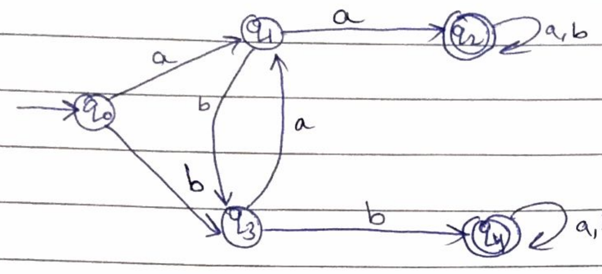
$\overline{FA} =$ 

$\overline{FA}$ 

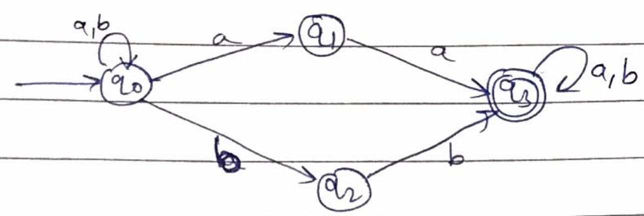
 a^+

$\overline{FA}$ 

Q. Draw FA for $(a+b)^* (aa+bb)(a+b)^*$



DFA



- $a^* b^* \neq (a+b)^*$
- $(a+b)^* = (a^*+b)^*$

- Regular
  - Output
    - Mealey
    - Moore
  - Checker
    - Yes
    - No

CONTEXT - FREE LANGUAGE

Context free language (CFG)

Terminal

Non-terminal

Defn:-

A CFG is a collection of three things-

1. An alphabet Σ of letters of terminal symbol
2. A set of non-terminal sysbols, in which 's' is the starting symbol
3. A finite set of productions

one non-terminal $\rightarrow$ finite string of terminal / Non-terminal set of terminal symbol → T; V → set of non-terminal symbols; S → starting symbol; P → set of productions.

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

Context Free language – A <u>language</u> generated by CFG is called CFL.
(language → string of terminal symbols)

Q) $S \to aS$ (S = starting symbol)
$S \to \wedge$ } CFG   Find CFL.

$S \Rightarrow aS$
$\Rightarrow aaS$
$\Rightarrow aaaS$
$\Rightarrow aaaaS$
$\Rightarrow aaaaaS$
$\Rightarrow aaaaa\wedge$

$\{\wedge, a, aa, aaa, \ldots\}$

language generated is a^*

Q) CFG :- $S \to SS$
$S \to a$
$S \to \wedge$

$S \Rightarrow SS$
$\Rightarrow aS$
$\Rightarrow aSS$
$\Rightarrow aaS$
$\Rightarrow \ldots$

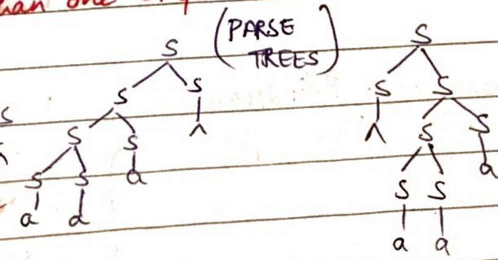
CFL is $a^* = \{\wedge, a, aa, aaa, \ldots\}$

\# For a CFL we can have more than one CFG.

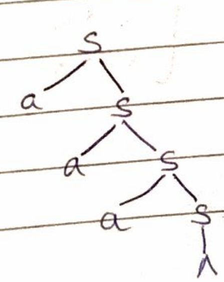
GRAMMER → Ambiguous (more than 1 parse tree) / Unambiguous

Ambiguous: 'aaa'
$S \to SS$
$S \to a$
$S \to \wedge$

(PARSE TREES)



Unambiguous: 'aaa'
$S \to aS$
$S \to \wedge$
→ unique parse tree

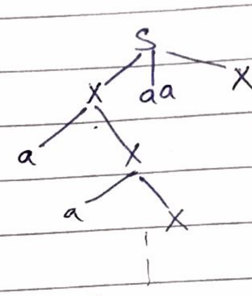


'$\wedge$' is neither a terminal symbol nor a non-terminal symbol.

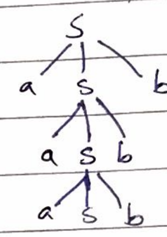
Q) $S \to aS$
$S \to bS$
$S \to a$
$S \to b$
} $\to (a+b)^+$

If we add $S \to \wedge \Rightarrow (a+b)^*$

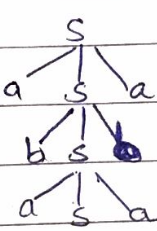
Q/ $S \to XaaX$ $\quad (a+b)^* aa (a+b)^*$
$X \to aX$
$X \to bX$ } $\Rightarrow$ any combⁿ of a, b.
$X \to \wedge$



Q/ $S \to aSb$
$S \to \wedge$
} $a^n b^n : n \geq 1$



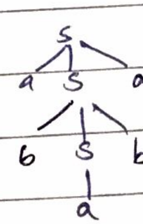
Q/ $S \to aSa$
$S \to bSb$
$S \to \wedge$
} Even palindrome
$\Downarrow$
length is even.



abaab
aSa

Odd palindrome { $S \to aSa$
$S \to bSb$
$S \to a$
$S \to b$

(Unambiguous grammer) Palindrome (can be even or odd) { $S \to aSa$
$S \to bSb$
$S \to a$
$S \to b$
$S \to \wedge$



Q/ $S \to aSa$, $S \to b$ } $a^n b a^n$

NOTATION —

$A \to B$ $\qquad$ $A ::= B$

or $\langle A \rangle \to \langle B \rangle$

BNF (Backus Normal Form or Backus Naur Form) containing arrow, vertical bars, terminal, non-terminal form.

$A \to AB/a$

$\Rightarrow A \to AB$

$A \to a$

• John W. Backus invented Algol (programming language) — in this language, CFG notation was first used.

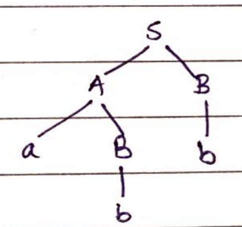
Naur — ~~ed~~ editor of report in which this appear.

TREE —

$S \to AB$ $\qquad$ abb

$A \to aB$

$B \to b$



- Syntax tree
- Parse tree
- Generation tree
- Production Tree
- Derivation tree

GRAMMER

Unambiguous
- for all ~~lang~~ word in language must have only one parse tree.

eg:- Palindrome grammer

$S \to aSa/bSb/a/b/\Lambda$

Ambiguous
- A CFG is called unambiguous if for at least one word in language we have two or more parse trees or derivation tree.

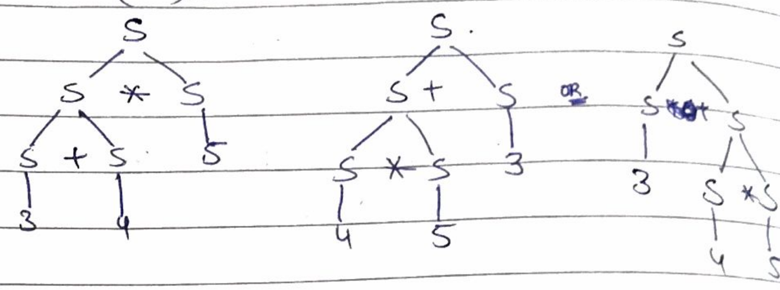
eg:- $S \to aS/Sa/a$

Q. S → aS/a <u>Unambiguous</u>

Q/ S → S+S / S*S / Number 3+4*5

Ambiguous

(3+4)*5 3+(4*5)



Q) S → */+/Number

+ → ++/+*/+number/**/*number/number+/number*/number number

* →

3+(4*5) ⇒ 3+(*45) ⇒ +3*45

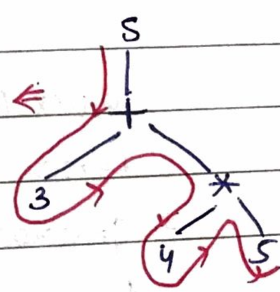
Prefix grammer

S → + —①

+ → number * —②

+ → number number —③

+3*45 ⇐



* Prefix notation follow 000 rule to solve the expression.

0 0 0

operator operand operand

Lukasiewicz Notation

The clever ~~paransiewic~~ paranthesis-free scheme invented by Polish logician Jan Lukasiewicz and it is also called Polish notation.

classmate
Date
Page

$* + * \underbrace{+ 1 2} + 3 4 5 6$

$* + * 3 \underbrace{+ 3 4} 5 6$

$* + \underbrace{* 3 7} 5 6$

$* \underbrace{+ 21 5} 6$

$\underbrace{* 26 6}$

156

Q/ $X \to X \oplus Y / Y$

$Y \to Z \circledast Y / Z$

$Z \to id$

Which of the following is true.

a) $\oplus$ is left associative while $\circledast$ is Right associative

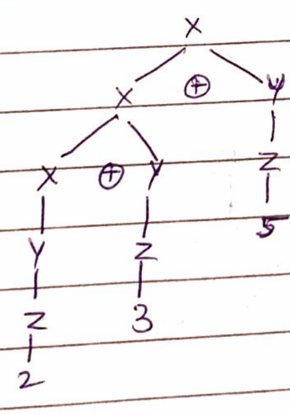
b) $\oplus$ & $\circledast$ are left associative

c) $\oplus$ is Right associative which $\circledast$ is left associative.

Right ass.

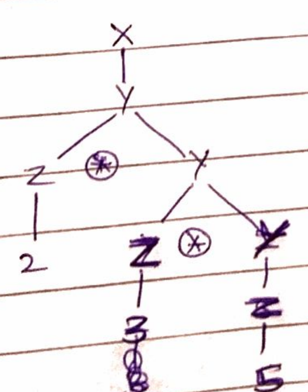
$2 \oplus 3 \oplus 5$

left ass.



$\oplus$ is left associative as $2 \oplus 3$ is solved first.

$2 \circledast 3 \circledast 5$



$\circledast$ is Right associative.

Q/ $S \to T * P$

$T \to U / T * U$

$P \to Q + P / Q$

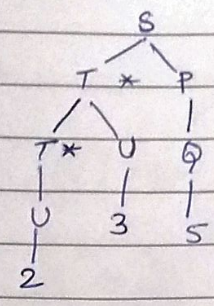
$Q \to id$

$U \to id$

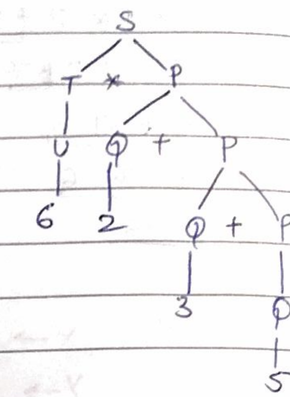
$* \to$ LA ✓ or RA

$+ \to$ LA or RA ✓

2*3*5



6*2+3+5



Q/ $E \to E*F / F+E / F$

$F \to F-F / id$

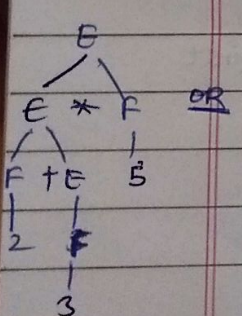
a) * is Higher precedence than +

~~b)~~ - is HP than *

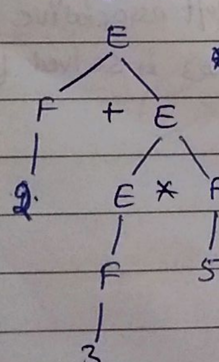
c) + & - have same precedence

d) + have HP than *

2+3*5

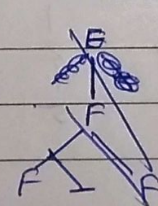


OR

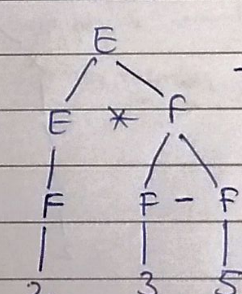


* = +

~~2-3*5~~



2*3-5



-> *

$\Rightarrow$ -> * = +

Q// int a [10] [1];

G1

D → int L;

L → id [E

E → num]

E → num] [E

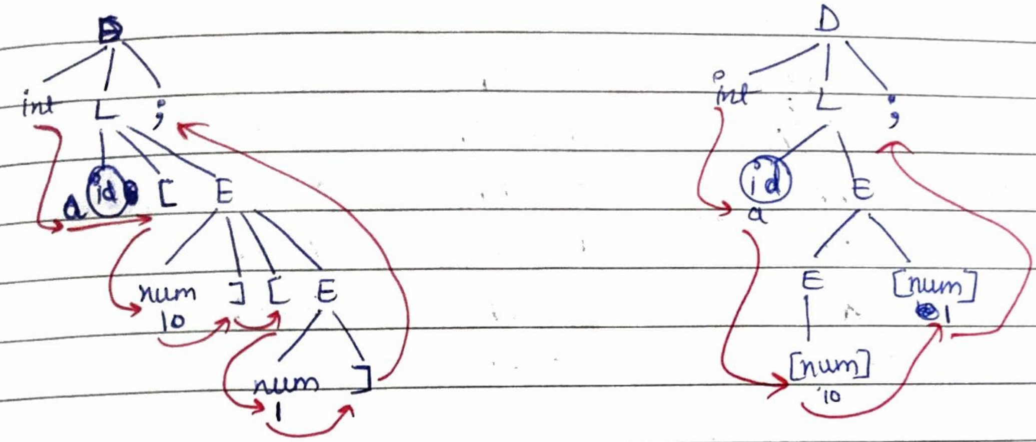
G2

D → int L;

L → id E

E → E [num]

E → [num]



CNF (Chomsky Normal Form)

- If CFG have only the production of the form

Non-terminal $\longrightarrow$ String of exactly two Non-terminal $V \to VV$

<u>or</u> Non-terminal $\longrightarrow$ one terminal $V \to T$

then it is called CNF

$S \to aSa$ (CFG) $\longrightarrow$ (CNF) $\begin{cases} S \to AB \\ B \to SA \\ A \to a \end{cases}$

Q/ $S \to aaaaS/aa$ (CFG) $\longrightarrow$ ~~$S \to AB$, $B \to S/C$~~, $A \to aaaa$, $C \to aa$

$\downarrow$

$S \to AAAAS/AA$

$A \to a$

$\Downarrow$

$S \to AAA\underbrace{AS}_{B}$, $S \to AA$, $A \to a$ $\longrightarrow$ $S \to A\underbrace{AAB}_{C}$, $B \to AS$ $\Rightarrow$ $S \to A\underbrace{AC}_{D}$, $C \to AB$, $B \to AS$ $\longrightarrow$

CNF

$\boxed{\begin{array}{l} S \to AD \\ D \to AC \\ C \to AB \\ B \to AS \\ S \to AA \\ A \to a \end{array}}$

Date ______
Page ______

Q/ $S \to aSa / bSb / aa / bb / a / b$ (CFG to CNF)

$A \to a$; $B \to b$

$S \to A\underbrace{SA}_{A_1}$

$S \to B\underbrace{SB}_{B_1}$

$S \to AA$

$S \to BB$

$S \to a$

$S \to b$

$\Rightarrow$

$S \to AA_1$

$A_1 \to SA$

$S \to BB_1$

$B_1 \to SB$

$S \to AA$

$S \to BB$

$S \to a$

$S \to b$

$A \to a$

$B \to b$

Q/ $S \to bA / aB$

$A \to bAA / aS / a$

$B \to aBB / bS / b$

$X \to a$

$Y \to b$

$S \to YA / XB$

$A \to YAA / XS / a \Rightarrow A \to YA_1$; $A_1 \to AA$

$B \to XBB / YS / b \Rightarrow B \to XB_1$; $B_1 \to BB$

Q/ $S \to aaaaS / aaaa$

$A \to a$

$S \to AAAAS$

$S \to AAAA$

Take $A_1 \to AA$

$\Rightarrow S \to A_1 \underbrace{A_1 S}_{A_2} \Rightarrow S \to A_1A_2$, $A_2 \to A_1S$

$S \to A_1 A_1$

$\Rightarrow$ CNF is

$S \to A_1A_2$

$S \to A_1A_1$

$A_2 \to A_1S$

$A_1 \to AA$

$A \to a$

classmate
Date ______
Page ______

GNF (GREIBACH Normal Form)

$$\left.\begin{array}{l} V \rightarrow TV^{+} \\ \text{or } V \rightarrow T \end{array}\right\} \Rightarrow V \rightarrow TV^{*}$$

: Non-terminal ⟶ (single terminal) (sequence of non-terminal)

or Non-terminal ⟶ single terminal

CFG → GNF

1. $S \rightarrow AB$
   $A \rightarrow aA/bB/b \Rightarrow$
   $B \rightarrow b$

$S \rightarrow aAB/bBB/bB$
$A \rightarrow aA/bB/b$
$B \rightarrow b$

~~CFG~~ → GNF
1. Convert ~~CNF~~ CFG to CNF (partially)
2. CNF → ~~CFG~~ GNF

2. $S \rightarrow abSb/aa$ CFG ⇒ CNF ⇒ GNF

1. Convert to CNF (partially)

$S \rightarrow ABSB/AA$ $A \rightarrow a$ $B \rightarrow b$

2. To GNF

$S \rightarrow aBSB/aA$
$A \rightarrow a$
$B \rightarrow b$

3. $S \rightarrow abaSa/aba$

1. Convert to CNF

$S \rightarrow ABASA/ABA$ $A \rightarrow a$ $B \rightarrow b$

2. To GNF

$S \rightarrow aBASA/aBA$
$A \rightarrow a$
$B \rightarrow b$

classmate

Date ____ Page ____

4. $S \to AB \mid BC$

$A \to aB \mid bA \mid a$

$B \to bB \mid cC \mid b$

$C \to c$

CNF

1. $S \to AB \mid BC$

$A \to AB \mid BA \mid a$

$B \to BB \mid CC \mid b$

$C \to c$

$A \to a$

$B \to b$

GNF

$S \to aBB \mid \cancel{aB} \, bAB \mid aB$

$S \to bBC \mid cCC \mid bC$

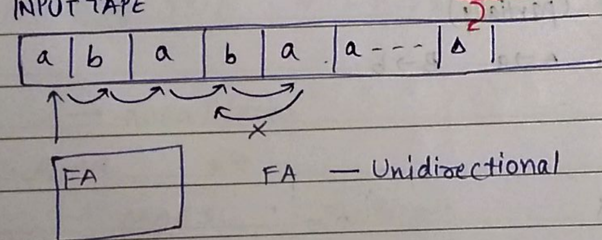
$A \to AB \mid bA \mid a$

$C \to c$

* $\boxed{CFL \longrightarrow PDA}$ PDA — Push Down Automata

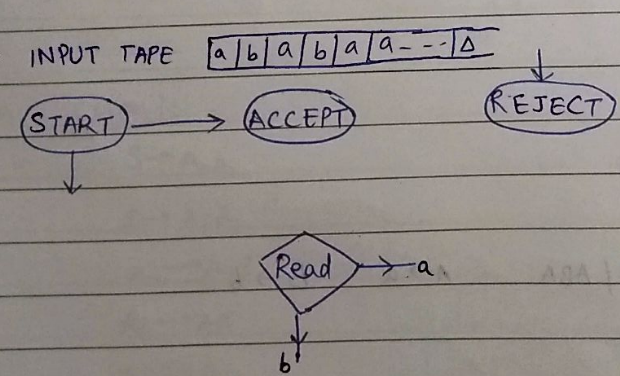
- INPUT TAPE

denotes end of the string



FA — Unidirectional

• INPUT TAPE

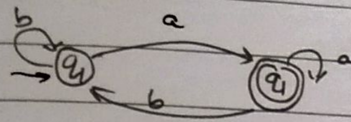


PDA - Combination of Start, Accept, Reject state & Read.

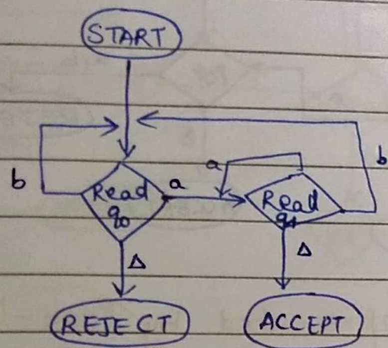
If at the end of string we are at ACCEPT, the String is accepted by PDA

If it is at REJECT, the string is not accepted by PDA.

Ques. FA → PDA

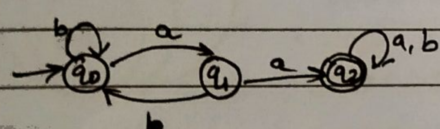


THEORITICAL MACHINE

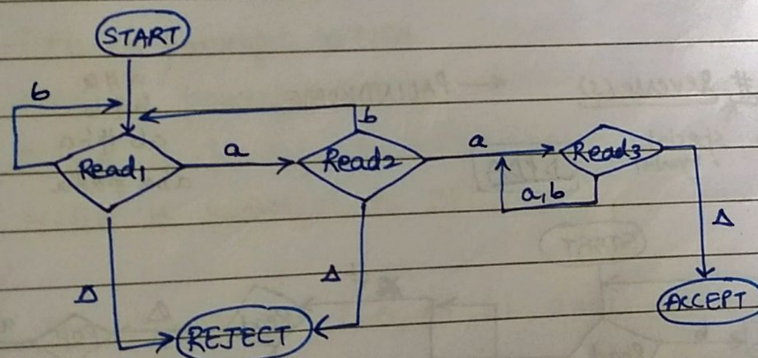


PRACTICAL MACHINE

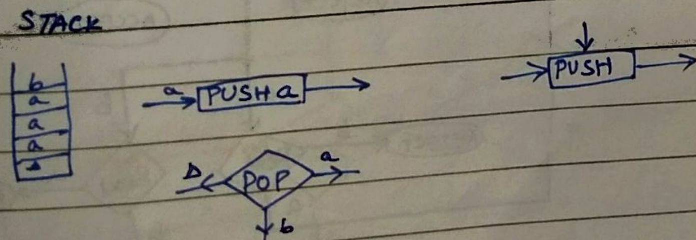
Q/



abaa

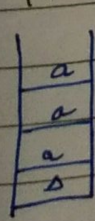


STACK



a	a	a	b	b	b	Δ

$L = \{a^n b^n : n \geq 0, 1, 2, ---\}$

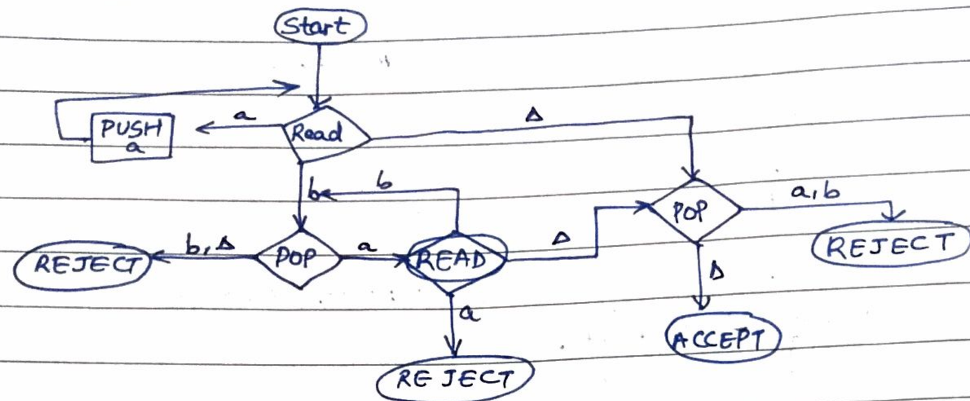


Push if u get 'a'

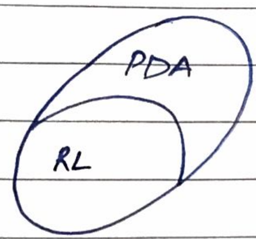
~~St~~ Pop when u get 'b'

If at last stack have only Δ then it is $a^n b^n$.

$L = a^n b^n$



* Every Regular language is represented by PUSH-DOWN AUTOMATA.



Q/ S # reverse(S) ← PALINDROME

S: string, #: special symbol

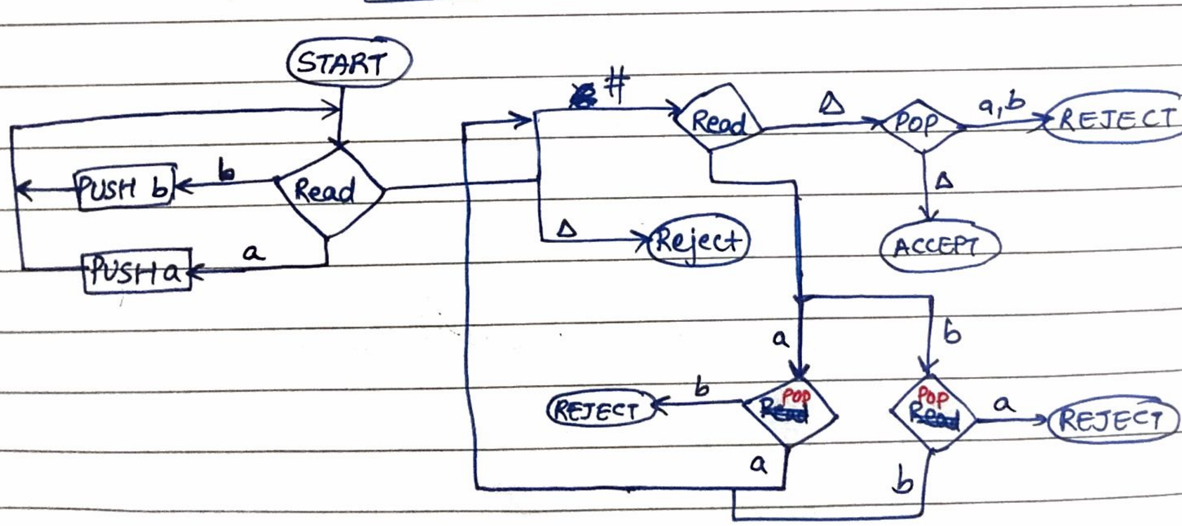
$\boxed{\text{DPDA}}$

a # a

b # b

ab # ba

aba # aba



PDA → DPDA (deterministic - PDA)

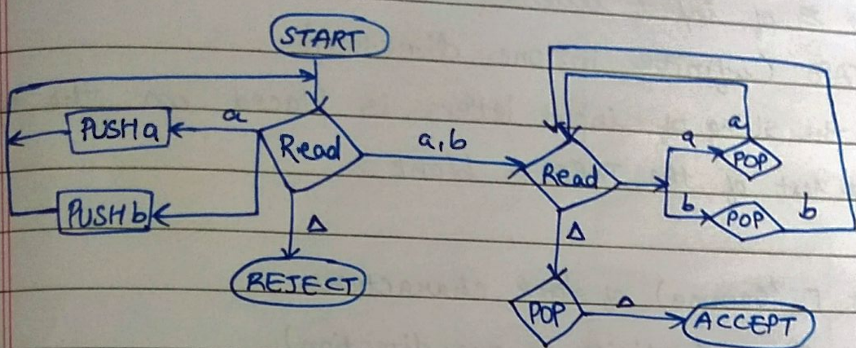
PDA → NPDA (Non-deterministic - PDA)

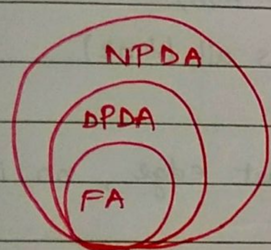
{ s X rev(s)
odd palindrome
even " }

classmate
Date
Page

ODD PALINDROME {a, b, aba, baa, aabaa, ---
s X rev(s)

NPDA





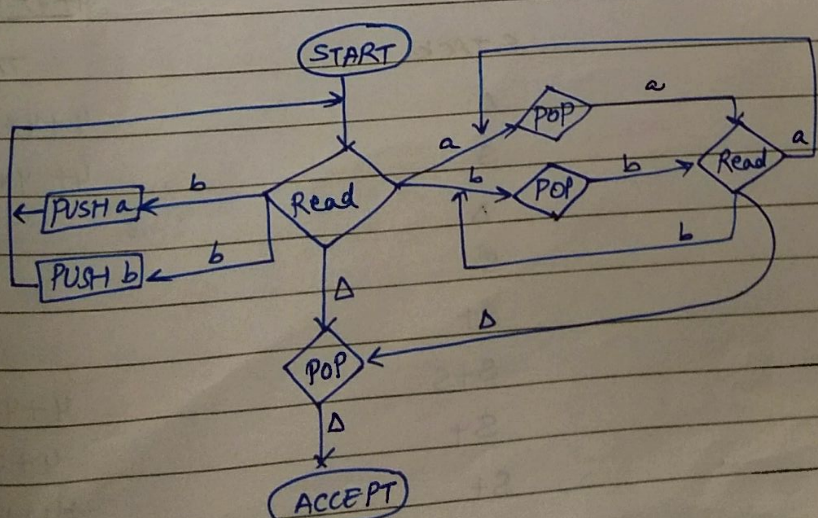
DFA = NFA → DFA & NFA are of same power.

* NPDA is most powerful
(NPDA > DPDA > FA)

Q1 Which is TRUE?

1. DFA is powerful to NFA
2. NFA is powerful to DFA
3. DPDA is powerful to NPDA.
4. NPDA is powerful to DPDA.

EVEN PALINDROME { Λ, aa, bb, aaaa, bbbb, --- }



→ <u>Definition of PDA</u>

A pushdown automata is a collection of 8 things –

i) An alphabet Σ of input letters.

ii) An input TAPE (infinite in one direction)
Initially, the string of input letters is placed on the ~~left~~ TAPE. The rest of the TAPE is blank.

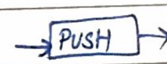
iii) An alphabet Γ (Gamma) of stack characters.

iv) A Pushdown stack (infinite in one direction)
Initially, the stack is empty (contains all blank).

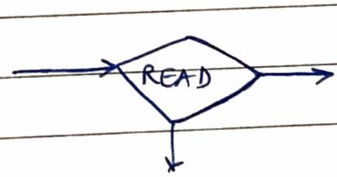
v) One start state that has only one-out edge, no in-edge.

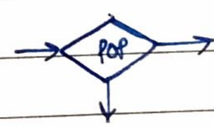
vi) Halt state of two kinds. They have in-edge, no out-edge.

vii) PUSH state introduce characters on top of stack



viii) Many branching states

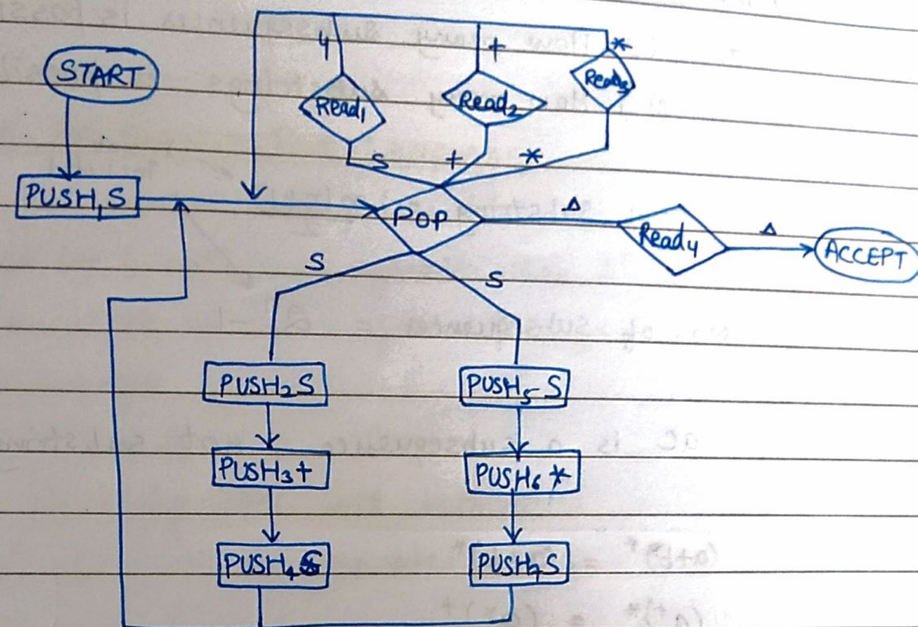




4+4*4

STATE	STACK	TAPE
START	Δ	4+4*4
$PUSH_1$, S	S	4+4*4
POP	Δ	
$PUSH_2$, S	S	
$PUSH_3$ +	S+	
$PUSH_4$ S	S+S	4+4*4
POP	S+	4+4*4 Δ
$Read_1$	S+	+4*4 Δ

$S \rightarrow S+S / S*S / 4$



POP (+)	S	+4*4 Δ
~~Read2~~	S	4*4 Δ
POP (S)	Δ	4*4 Δ
$PUSH_5$ S	S	4*4 Δ
$PUSH_6$ *	S*	4*4 Δ
$PUSH_7$ S	S*S	4*4 Δ
POP	S*	4*4 Δ
$Read_1$	S*	*4 Δ
POP	S	*4 Δ
$Read_3$	S	4
POP	Δ	4 Δ
$Read_1$	Δ	Δ
POP	Δ	Δ
$Read_4$	Δ	Δ ~~ACCEPT~~
ACCEPT	Δ	Δ

Q/ let $S = abc$

Then (i) How many subsequences is possible (7)

(ii) How many substrings is possible. (6)

a ab abc
b ac
c bc

a ab abc
b bc
c

λ not included

$$\text{No. of substring} = \frac{n(n+1)}{2}$$

λ included

$$\frac{n(n+1)}{2} + 1$$

$$\text{No. of subsequences} = 2^n - 1$$

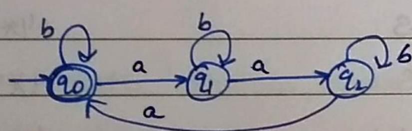
$$2^n$$

ac is a subsequence, not substring

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

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

Q/ FA where no. of a's is divisible by 3.



Q) $(a+b+c)^*$ equals to

✓ i) $(a^* + b^* + c^*)^*$

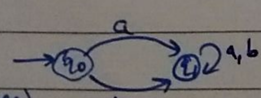
✓ ii) $(a^* b^* c^*)^*$

✗ iii) $((a+b)^* + c^*)^*$

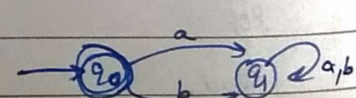
✓ iv) $(a^* b^* + c^*)^*$

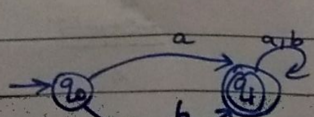
Q/ How many FA is possible if there ~~are~~ is two state (q_0, q_1) over $\Sigma = \{a,b\}$

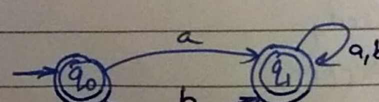
$L = \phi$ (empty language)



$L = \Lambda$ (Null string)







4 FA's

PUMPING LEMMA FOR REGUALAR LANGUAGE

$xyz \in L$ then $xy^nz \in L$

* PUMPING LEMMA FOR CONTEXT-FREE LANGUAGE

If G is any CFL in CNF with p line production and w is any word generated by G with length greater than 2^p.
then, we can break up w into five substrings –

$$w = uvxyz$$

such that 1) x is not $\wedge$

2) v & y are not both $\wedge$.

Then, uv^nxy^nz will be generated by G.

Q/ $L = \{a^n b^n a^n \ ; \ n = 0, 1, 2, ---\}$

$a^2b^2a^2 \in L$

Let $u = a$, $v = ab$, $x = b$, $y = a$, $z = a$

Then

$$uv^2xy^2z = aababbaaa = a^2bab^2a^3 \notin L$$

$\therefore$ L is not CFL.

aabb aa

u v x y z

~~aabab~~

aababbaaa

Q/ $L = \{a^n b^n c^n \ ; \ n = 0, 1, 2, --\}$

$a^2b^2c^2 \in L$

let $u = a$, $v = a$, $x = bb$, $y = c$, $z = c$

$uv^2xy^2z = aaabbccc = a^3b^2c^3 \notin L$

$\therefore$ L is not CFL

$a^2b^2c^2$

aa bb cc

u v x y z

~~uv^2xy^2z~~

~~a abab~~

$aa^2\,bb\,c^2c$

a^3bbc^3

Theorem: PUMPING LEMMA FOR CFL

Let L be a CFL in CNF $\bar{c}$ p-line production, then, any word w in L $\bar{c}$ length $> 2^p$ broken into five parts.

$$w = uvxyz$$

St.

$$\text{length}(vxy) \le 2^p$$

length(x) > 0

length(v) + length(y) > 0

and $uv^n xy^n z$ are in language L.

$$uv^n xy^n z = \begin{cases} uvxyz \\ uv^2xy^2z \\ uv^3xy^3z \\ \vdots \end{cases} \in L$$

Closure properties in CFL

Let L_1 and L_2 is CFL

then,

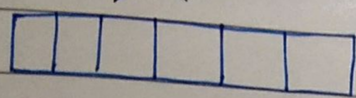
1) $L_1 \cup L_2$ $(L_1 + L_2)$ is CFL
2) L_1^* is CFL
3) $L_1 L_2$ is CFL
4) $L_1 \cap L_2$ is ~~CFL~~ may or may not be
5) $\overline{L_1}$ may or may not be

Rational no. ⟶ terminating or repeating

TURING MACHINE

(Allen Turing)

⟶ ⟵

TAPE 

left ended pt.

zerifo (zero)

Non-rational — π, e, ϕ
euler no. (e), Golden ratio (ϕ)

Fibonacci Series:- 1 1 2 3 5 8 ...

$\frac{t_{n+1}}{t_n} \approx 1.61$

Automated machine

A turing machine is a quintuple $(K, \Sigma, \delta, S, H)$ where

K is a finite set of states

Σ is an alphabet, containing blank symbol '⊔' and left end symbol 'Δ' but not containing '$\rightarrow$' or '$\leftarrow$'

$S \in K$ is the initial state

$H \subseteq K$ is the set of halting states / final states

δ is transition function.

$\delta : (K - H) \times \Sigma$ to $K \times (\Sigma \cup \{\leftarrow, \rightarrow\})$

such that

i) $\forall q \in K - H$ if $\delta(q, \Delta) = (p, b)$ then $b = \rightarrow$

ii) $\forall q \in K - H$ and $a \in \Sigma$, if $\delta(q, a) = (p, b)$ then $b \neq \Delta$

Action performed on TAPE

$(q_0, \Delta) = (q_0, \rightarrow)$

	b			
Δ	~~a~~	a	a	⊔

q_0, q_1, q_2

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

$\rightarrow$ Right / forward move

$\leftarrow$ Left / backward move

$\left.\begin{matrix} ⊔ \\ a \end{matrix}\right\} \rightarrow$ Replace it

Example – $M = \{K, \Sigma, \delta, s, \{h\})$

$K = \{q_0, q_1, h\}$

$\Sigma = \{a, \sqcup, \triangleright\}$

$s = q_0$

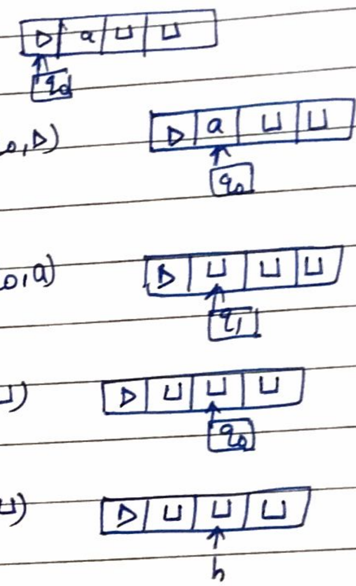
q	σ	$\delta(q,\sigma)$
q_0	a	$(q_1, \sqcup)$
q_0	$\sqcup$	$(h, \sqcup)$
q_0	$\triangleright$	$(q_0, \rightarrow)$
q_1	a	(q_0, a)
q_1	$\sqcup$	$(q_0, \rightarrow)$
q_1	$\triangleright$	$(q_1, \rightarrow)$

$(q_0, \triangleright)$

(q_0, a)

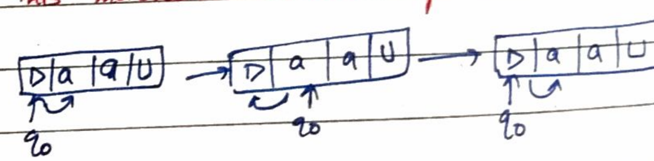
$(q_1, \sqcup)$

$\therefore (q_0, \sqcup)$



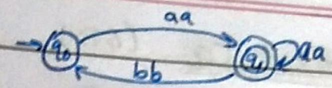
q	σ	$\delta(q,\sigma)$
q_0	a	$(q_0, \leftarrow)$
q_0	$\sqcup$	$(h, \sqcup)$
q_1	$\triangleright$	$(q_0, \rightarrow)$

← This machine never stops



Ch-6 Transition Graph

FA ⟺ TG

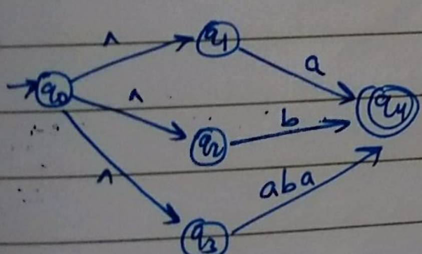


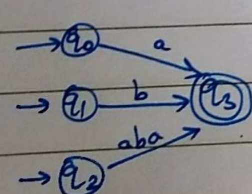
- A TG is a collection of 3 things –

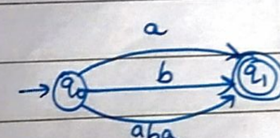
1) A finite set of states, at least one of which is designated as start state (–) and some (or maybe none) designated as final state (+).
2) An alphabet Σ of possible input letters.
3) A finite set of transitions based on reading substring of input letters (or even null).

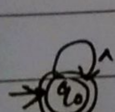
FA	TG
• String starting state	• Multiple starting state
• edge label is alphabet	• edge label is substring.

$L = \{a, b, aba\}$

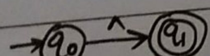




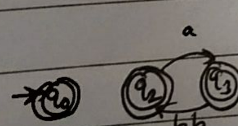




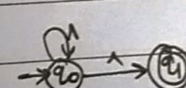
①



②



③



④

All TGs accept only Λ.