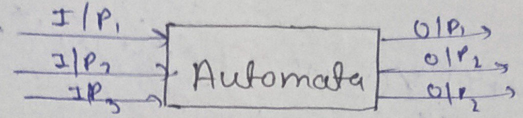
TOC



Input : Finite sets of i/p (Ɛ)
Output : Finite sets of O/P
State : Finite sets of states
State Relation : Input symbol and Previous state
Output Relation : either i/p or o/p

Output Depends

i/p Symbols	State	Automata
✓	X	DFA / NFA / moore (memory X)
✓	✓	PDA, LBA, Turing (memory ✓)
✓	✓	mealy (memory X)

Grammer (V, Ɛ, S, P)

V → Set of variable (capital letter)
Ɛ → set of i/p alphabets (small letter
S → Starting symbol
P → Production Rule

non-terminal

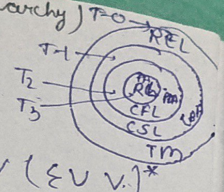
$S \rightarrow aS \mid b$

$S \rightarrow \epsilon$ terminal

$S \rightarrow aS$
$S \rightarrow b$
$S \rightarrow \epsilon$,

$S \rightarrow aS \mid b \mid \epsilon$

Types of Grammer — (Chomsky hierarchy)



① Type-0 (Understricted Grammer)

$\alpha \rightarrow B$

$\alpha \in (V+T)^* V (V+T)^*$ / $(\Sigma \cup V)^* V (\Sigma \cup V)^*$

$B \in (V+T)^*$

② Type-1 (Context Sensitive Grammer)

$\alpha \rightarrow B$

$|\alpha| \le |B|$ except null

③ Type-2 (Context Free Grammer)

$\alpha \rightarrow B$

↓

one variable

④ Type-3 (Regular Grammer)

$S \rightarrow aS \mid b$ except null

or

$S \rightarrow Sa \mid b$

Q Find Type of Grammer

(i) $S \rightarrow AB$
$A \rightarrow a$
$B \rightarrow b$

T_2

(ii) $S \rightarrow aS$
$S \rightarrow b$

T_3

(iii) $S \rightarrow AB$
$A \rightarrow a$
$B \rightarrow b \mid \epsilon$

T_2

(iv) $B \rightarrow \epsilon$

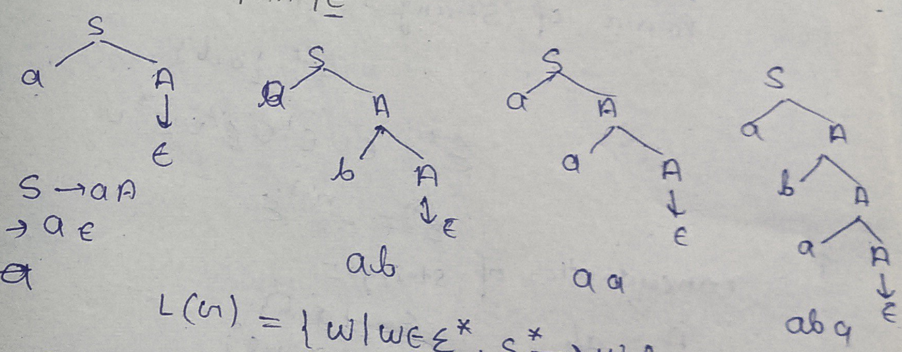
T_3

v) $S \rightarrow ASBS$
$AS \rightarrow SB$
$BS \rightarrow AA$
$01 \rightarrow BA$
$11 \rightarrow AB$
$A \rightarrow 0$
$B \rightarrow \epsilon$

T_1

Que $S \rightarrow aA$
$A \rightarrow aA \mid bA \mid \epsilon$

Non-terminal $\{S, A\}$
Terminal (a, b)



$S \rightarrow aA$
$S \rightarrow a\epsilon$
a

$L(G) = \{ w \mid w \in \Sigma^*, S \xrightarrow{*} w \}$

$S \rightarrow aA$
$A \rightarrow aA$
$A \rightarrow bA$
$A \rightarrow \epsilon$

four Production

Production valid or not

$\alpha \rightarrow B$

↓

atleast one non-terminal

String —

Collection of Alphabet

Properties of string –

1. Empty String (no string generated)
   ϵ (null production)
2. length of string = no of symbols
3. Power of String = 2^n $\Sigma\{a,b\}$

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

$$\Sigma^* = \epsilon \cup \epsilon^1 \cup \Sigma^2 \cup \Sigma^3 \cup$$

$$\Sigma^n = \Sigma^{n-1}\Sigma$$

4. Concentration of strings –

$A = ab$ $\quad\Sigma = \{a,b\}$

$B = abc$ $\quad\Sigma = \{a,b,c\}$

$A \cdot B = ab \cdot abc$

$BA = abc \cdot ab$

$$AB \neq BA$$

it means 2 strings concentration is not always the same.

DFA (Deterministic Finite Automata)

5 tuples $(Q, \Sigma, \delta, q_0, F)$

Q = Partition of states used in DFA

Σ = input symbols

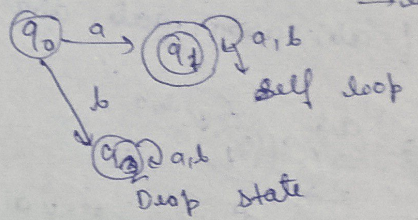
δ = transition function $Q \times \Sigma \to Q$

$q_0 \to$ initial state

$F \to$ final state $\to$ q_1

Q. Design a DFA using $\Sigma = \{a,b\}$ Starting with a

$L = \{a, ab, aa, ab, abb \ldots\}$ length not decided



State transition diagram.

Transition function

$\delta(q_0, a) \to q_1$

$\delta(q_0, b) \to q_2$

$\delta(q_1, a) \to q_1$

$\delta(q_1, b) \to q_1$

$\delta(q_2, a) \to q_2$

$\delta(q_2, b) \to q_2$

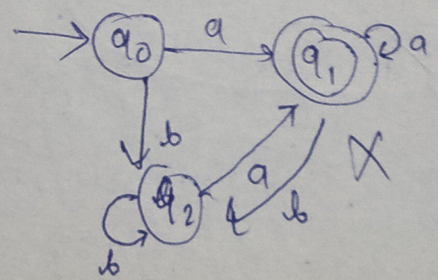
Transition Table

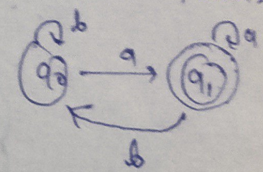
State	a	b
q_0	q_1	q_2
q_1	q_1	q_1
q_2	q_2	q_2

All three can be auto connected

Q. Design a DFA using $\Sigma = \{a,b\}$ ending with a

$L = \{a, aa, aaa, ba, baa, aba \ldots$

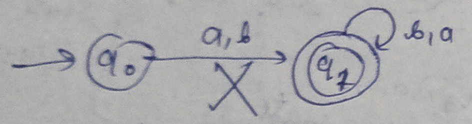


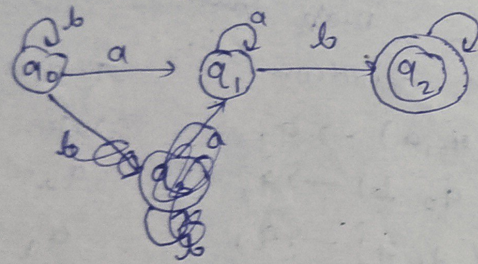


Transition

Q. Design a DFA using input $\Sigma(a,b)$ that except all string containing 'ab' as a substring

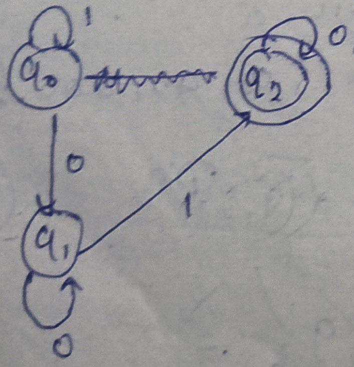
$L = \{ab, abba, aab, aba \ldots\}$





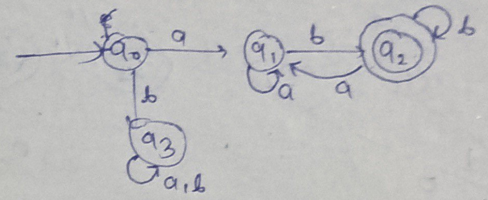
Q. Design a DFA from Transition Table

	0	1
q_0	q_0	q_0
q_1	q_1	q_2
q_2	q_2	q_2



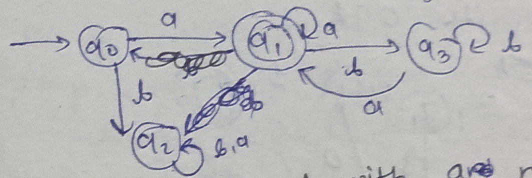
Q. Design a DFA using input $\Sigma(a,b)$ that except all string starting with a and with b

$L = \{ab, abab, aab, abbab, aab \ldots\}$



Q. Starting with a & end with a

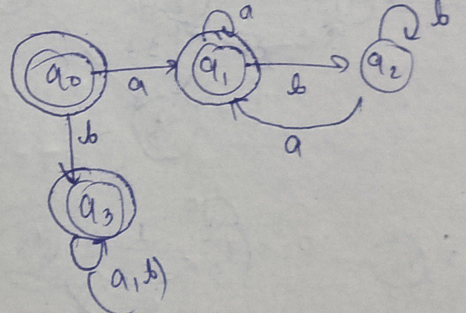
$L = \{a, aa, aba, abba, aaa, \ldots\}$



Q. neither start with a nor end with b

$L = \{ba, bba, baba, \ldots\}$

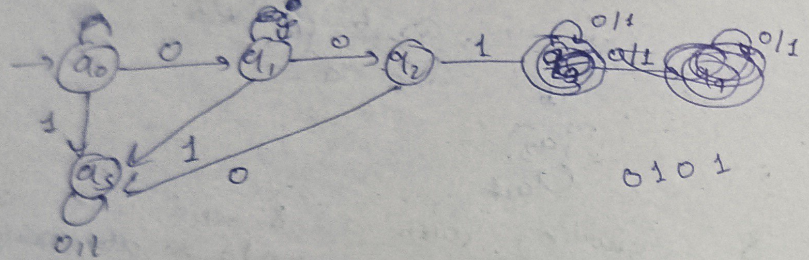
Start with a & $\Rightarrow$ {ab, abb, aab,
end with b



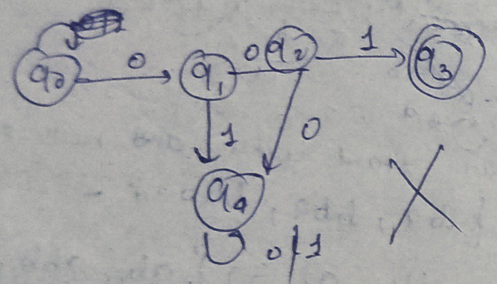
Compliment of a DFA

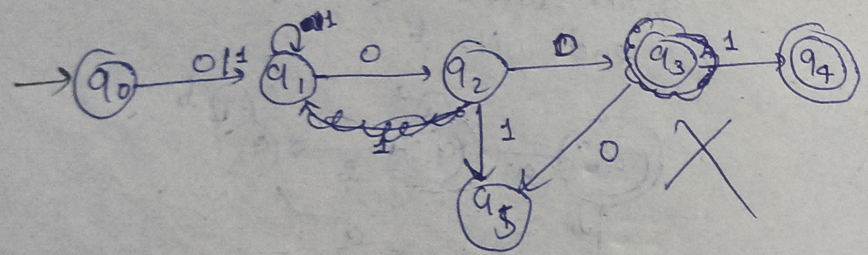
Final state $\Longleftrightarrow$ Non Final state

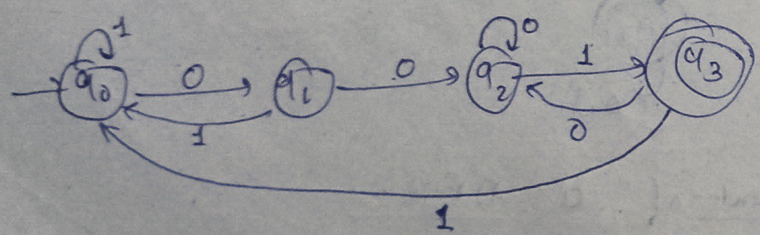
Q. Design a DFA using input $\Sigma = \{0,1\}$ where all string except string starting with '001' string



Q. end with 001

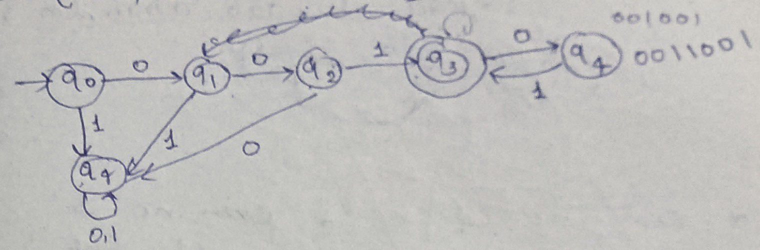




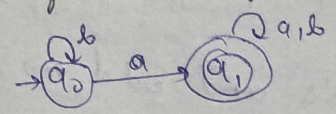


Start with 001 & end with 01

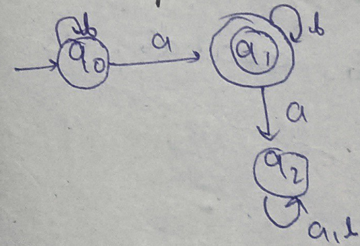
$L = \{001, 001 \ldots 01\}$



Q. Design a DFA using $\Sigma = \{a, b\}$ where all strings containing atleast one a



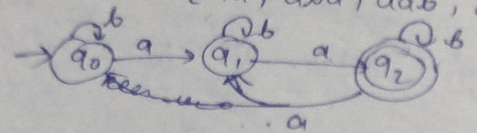
Q exactly 1 a



Q. almost 1 'a'

$\Sigma = \{a, b\}$ where string have even no of a

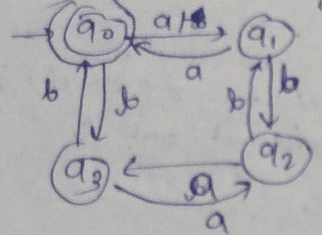
$L = \{aa, aba, aab, abab, baa\}$



aaab
aab

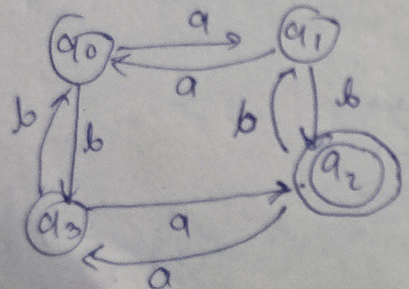
Q even no of 'a' & even no 'b'

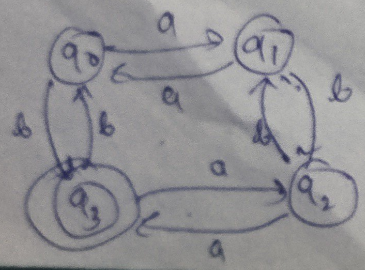
$L = \{aabb, abab, abba, baab\}$
ϵ, aa, bb



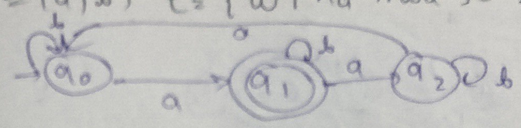
Q. odd no of a's and odd no b's
even no of a's and odd no b's

$L = \{a, b, ab, ababab,$





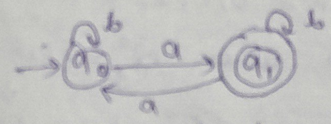
$\Sigma = \{a, b\}$ $L = \{w \mid n_a \bmod 3 = 1\}$



0 1 2

$L = \{a, aaa, ba, ab$

Q. $\Sigma = \{a, b\}$ Design a DFA where $\{w_a \bmod 2 = 1\}$



0 1

Minimization of DFA

Equavalency method

Q. Present state — Next state a, b

$\rightarrow q_0$ — q_1 1,2,2 q_5 1,3,3

q_1 — q_6 1,1,4 q_2 2,4,5

q_2 (final) — q_0 — q_2

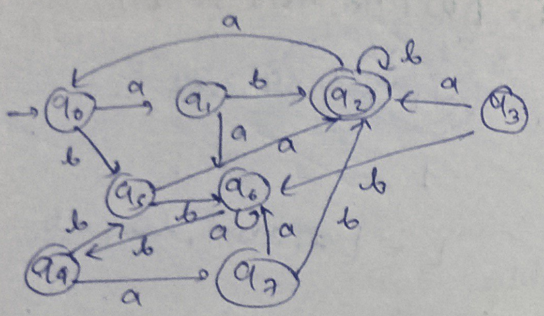
q_3 — q_2 2,4,5 q_6 1,1,4

q_4 — q_7 1,2,2 q_5 1,3,3

q_5 — q_2 2,4,5 q_6 1,1,4

q_6 q_6 q_4 1,1,4

q_7 q_6 q_2 1,1,4 3 4 5



Equivalence method

$\pi_k = \pi_{k-1}$ $\quad k \in 0, 1, 2, 3 \ldots$

$\pi_0 = \{ Q_1^0, Q_2^0 \}$ where $Q_1^0 = \{$ set of non-final state

$Q_2^0 = \{$ set of final state

$\pi_0 = \{ \overset{1}{\{q_0, q_1, q_3, q_4, q_5, q_6, q_7\}}, \overset{2}{\{q_2\}} \}$

$\pi_1 \Rightarrow \{ \overset{1}{\{q_0, q_4, q_6\}}, \overset{2}{\{q_1, q_7\}}, \overset{3}{\{q_3, q_5\}} \overset{4}{\{q_2\}} \}$

$\pi_2 = \{ \overset{1}{\{q_0, q_4\}}, \overset{2}{\{q_1, q_7\}}, \overset{3}{\{q_3, q_5\}} \overset{4}{\{q_6\}} \overset{5}{\{q_2\}} \}$

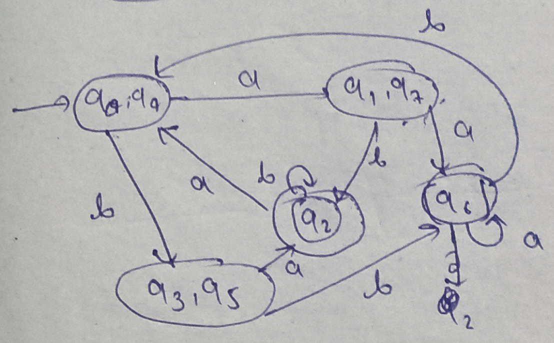
$\pi_3 = \{ \underset{1}{\{q_0, q_4\}}, \underset{2}{\{q_1, q_7\}}, \underset{3}{\{q_3, q_5\}}, \underset{4}{\{q_6\}}, \underset{5}{\{q_2\}} \}$

$\pi_2 = \pi_3$ $\quad$ total 5 state

$k \Rightarrow 3$

$\rightarrow \{q_0, q_4\}$ initial state

	a	b
$[q_0 q_4]$	$[q_1, q_7]$	$[q_3, q_5]$
$[q_1, q_7]$	$[q_6]$	$[q_2]$
$[q_3 q_5]$	$[q_2]$	$[q_6]$
$[q_6]$	$[q_6]$	$[q_0 q_4]$
$[q_2]$	$[q_0, q_4]$	$[q_2]$



Myhill Nerode Theorem (Table-filling method)

(q_0 to q_5)

Step 1 Remove all unreachable states

Step 2 Design a table for all pairs of states

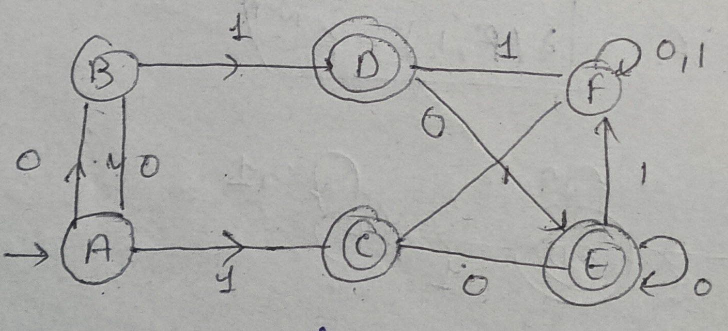
Step 3 mark all pairs where $A \in$ Final & $B \notin$ Final and vice versa

Step 4 If there are any unmarked pair (A, B) such that $[\delta(A, x), \delta(B, x)]$ is marked then we have to do mark (A, B) also marked

Repeat this process until no more marking can be made.

Step 5 Combine all unmarked pair & mark them as a single state

Q.



(1) Unreachable state nhi h

(2)

	A	B	C	D	E	F
A	×	×	×	×	×	×
B	•	×	×	×	×	×
C	✓	✓	×	×	×	×
D	✓	✓		×	×	×
E	✓	✓			×	×
F	✓	✓	✓	✓	✓	×

due to CF

Final state : $\{C, D, E\}$

Non-final : $\{A, B, F\}$

(A, B)

$\delta(A, 0) \Rightarrow B^{Non f}$ | $\delta(A, 1) = C^{f}$

$\delta(B, 0) \Rightarrow A^{N.f}$ | $\delta(B, 1) = D^{f}$

(C, D)

$\delta(C, 0) = E^{f}$ $(C, 1) = F^{nf}$

$\delta(D, 0)\ E^{f}$ $(D, 1) = F^{nf}$

Present state	a	b
$\rightarrow q_0$	q_1 1, ~~2~~, 2	q_5 1, ~~3~~, 3
q_1	q_6 1, ~~1~~, 4	q_2 ~~1~~, ~~4~~, 5
$*q_2$	q_0	q_2
q_3	q_2 2, ~~4~~, 5	q_6 ~~1~~, ~~1~~, 4

(C, E)

$\delta(C,0) = E$ $\quad\quad \delta(C,1) = F$

$\delta(E,0) = E$ $\quad\quad \delta(E,1) = F$

(D, E)

$\delta(D,0) = E^{F}$ $\quad\quad \delta(D,1) = F^{N.F}$

$\delta(E,0) = E$ $\quad\quad \delta(E,1) = F^{N.F}$

(A, F)

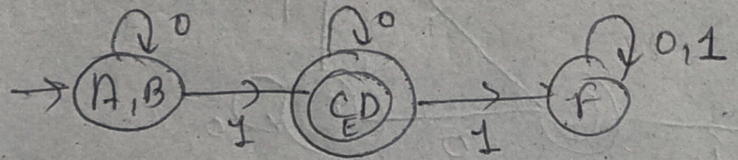
$\delta(A,0) = B^{N.F}$ $\quad\quad \delta(A,1) = C^{F}$

$\delta(F,0) = F^{N.F}$ $\quad\quad \delta(F,1) = F^{N.F}$

(B, F)

$\delta(B,0) = A$ $\quad\quad \boxed{\delta(B,1) = D^{R}}$

$\delta(F,0) = F$ $\quad\quad \boxed{\delta(F,1) = F^{N.F}}$



Q Minimise the DFA using equivalence & MNT

Present state	a	b
$\rightarrow q_0$	q_1 ~~1~~, ~~2~~, 2	q_5 ~~1~~, ~~3~~, 3
q_1	q_6 ~~1~~, ~~1~~, 4	q_2 ~~2~~, ~~4~~, 5
$^* q_2$	q_6	q_2
q_3	q_2 ~~2~~, ~~4~~, 5	q_6 ~~1~~, ~~1~~, 4
q_4	q_7 ~~1~~, ~~2~~, 2	q_5 ~~1~~, ~~3~~, 3
q_5	q_2 ~~2~~ ~~4~~ 5	q_6 ~~1~~, ~~1~~, 4
q_6	q_6 ~~1~~, ~~1~~, 4	q_4 ~~1~~, ~~1~~, 1
q_7	q_6 ~~1~~, ~~1~~, 4	q_2 ~~2~~, ~~2~~ 5

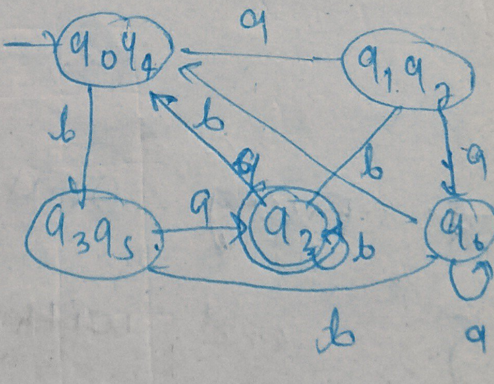
$\pi_0 = \{\{q_0, q_1, q_3, \overset{1}{q_4}, q_5, q_6, q_7\}, \{\overset{2}{q_2}\}\}$ set of final state $= \{q_2\}$

$\pi_1 = \{q_0, \overset{1}{q_4}, q_6\}, \{q_1, \overset{2}{q_7}\}, \{q_3, \overset{3}{q_5}\}, \{\overset{4}{q_2}\}$ non-final $= \{q_0, q_1, q_3, q_4, q_5, q_6, q_7\}$

$\pi_2 = \{q_0, \overset{1}{q_4}\} \{q_1, \overset{2}{q_7}\} \{\overset{3}{q_3}, q_5\} \{\overset{4}{q_6}\} \{\overset{5}{q_2}\}$

$\pi_3 = \{q_0, q_4\} \{q_1, q_7\} \{q_3 q_5\}. \{q_6\} \{q_2\}$

	a	b
$q_0 q_4$	$q_1 q_7$	$q_3 q_5$
$q_1 q_7$	q_6	q_2
$q_3 q_5$	q_2	q_6
q_6	q_6	$q_0 q_4$
q_2	$q_0 q_4$	q_2



NFA (Non-Deterministic Finite Automata)

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

$Q \rightarrow$ Finite set of states

$\Sigma \rightarrow$ i/p alphabets

$\delta \rightarrow$ transition $f x^n$

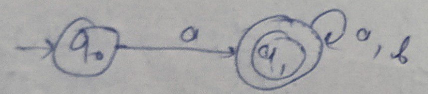
$\delta : Q \times \Sigma \longrightarrow 2^Q$

$q_0 \rightarrow$ Initial states

$F \rightarrow$ Final states

Design a NFA using $\Sigma = \{a, b\}$ that except all string starting with a

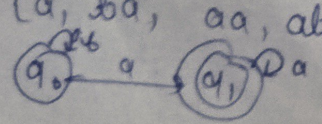
$L = \{a, ab, a \ldots\}$



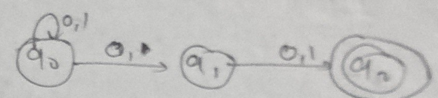
☆ Every DFA will be NFA

Q. end with 'a'

$L = \{a, ba, aa, aba \ldots\}$



$\Sigma = \{0, 1\}$ second alphabet from right end '0'



Transition Table

	0	1
q_0	q_0	q_1
q_1	q_2	q_2
q_2	—	—

Transition $f x^n$

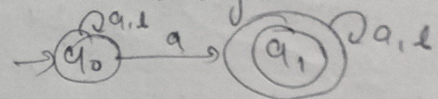
$\delta(q_0, 0) \rightarrow q_0$ or q_1

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

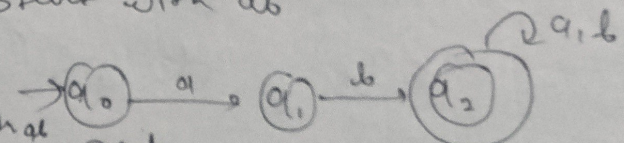
$\delta(q_1, 0) \longrightarrow q_2$

$\delta(q_p, 1) \rightarrow q_1$

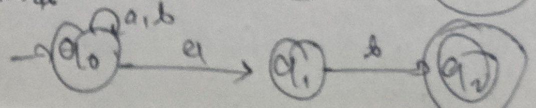
Q $\Sigma = \{a, b\}$
all string contain a



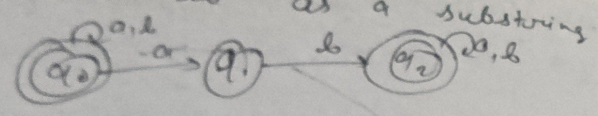
Q. $\Sigma = \{a, b\}$ using where all string start with ab



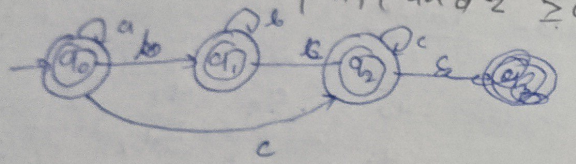
end with ab



Contain ab as a substring



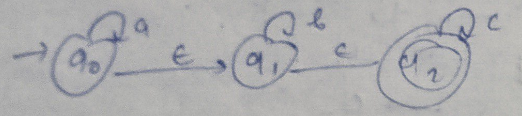
Q. Design NFA
$L = \{a^n b^y c^z \mid n, y \text{ and } z \geq 0\}$



$\{\phi, abc, bca, aa, ab, bc, ca\}$

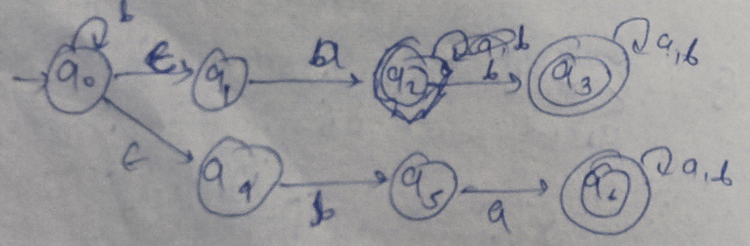
ε - NFA

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



Q. ε-NFA that except substring ab or ba in the language

ab, ba, aab, bba, bab..



How to convert ε-NFA to NFA

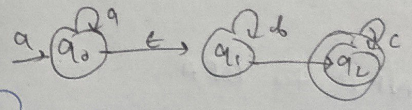
Step 1 - Find all ε closure for each state

Step 2. Compute transition for all states i.e calculated in step 1

Step 3 - Again calculate ε closure

Step 4 - Perform union operation for previous step (calculate states)

Step 5 - Design equivalent NFA



q_0

ε closure	a	ε closure
q_0	q_0	$\{q_0, q_1, q_2\}$
q_1	ϕ	
q_2	ϕ	

q_0

ε-closure	b	ε-closure
q_0	ϕ	$\{q_1, q_2\}$
q_1	q_1	
q_2	ϕ	

q_0

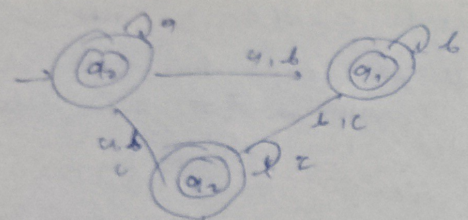
q_0		
q_1	ϕ	$\{q_2\}$
q_2	ϕ	
	q_2	

q_1

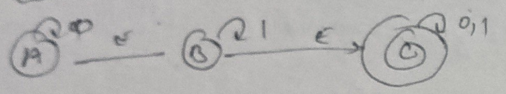
ε-closure	a	b	c
q_1	ϕ	q_1	ϕ
q_2	ϕ	ϕ	q_2

ε-closure $\phi, q_1 q_2, q_2$

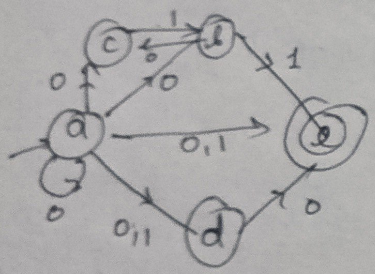
	a	b	c
q_0	$\{q_0, q_1, q_2\}$	$\{q_1, q_2\}$	$\{q_2\}$
q_1	ϕ	$\{q_1, q_2\}$	$\{q_2\}$
q_2	ϕ	ϕ	$\{q_2\}$



H.W Convert ε-NFA to NFA



Conversion of NFA to DFA

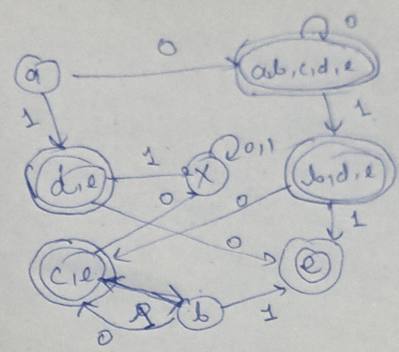


State	0	1
a	{a,b,c,d,e}	{d,e}
b	{c}	{e}
c	ϕ	{b}
d	{e}	ϕ
e	ϕ	ϕ

DFA

State	0	1
→a	[a,b,c,d,e]	[d,e]
[a,b,c,d,e]	[a,b,c,d,e]	[b,d,e]
[d,e]	[e]	ϕ
[b,d,e]	[c,e]	[e]
final→[e]	ϕ	ϕ
[c,e]	ϕ	[b]
[b]	[c]	[e]
[c]	ϕ	[b]

ϕ → dead state



H.W Conversion of NFA to DFA

* → final

(1)

State	a	b
→q_0	*q_0	q_1
q_1	q_1	q_0 or q_1

(2)

State	a	b
→q_0	q_0 or q_1	q_2
q_1	q_0	q_1
*q_2	ϕ	q_0 or q_1

(3)

State	0	1
→p	p or q	p
q	ϕ	r
*r	p or r	q