7th July 2018.

ഓം നമഃ ശിവായ

Day 2

Symbols :- Σ if $\Sigma = \{0,1\}$

Strings :- w $w = \{00, 11, 001\}$

Language :- L $w = 00$ $|w|$ indicates length of the string

$|w| = 2$

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

Consider a language containing strings of length 2.

(i) $L_1 = \{aa, ab, bb, ba\}$ 4 (2^2) L_1: set of strings of length 2

$L_2 = \{aaa, \ldots\}$ 8 (2^3) L_2: set of strings of length 3

(ii) $\Sigma = \{a,b,c\}$

$L_1 = \{aa, ab, ac, ba, bb, bc, ca, cb, cc\}$ - 9

$L_2 = \{abc, \ldots\}$ 27

(i) $|\Sigma| = 2$ (ii) $|\Sigma| = 3$

$\Sigma^i = 4$ $i=2$

$\Sigma^i = 8$ $i=3$

Powers of Σ :-

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

$\Sigma^1 = \{a, b\}$ $i=1$ i denotes string-len()

$\Sigma^2 = \{aa, ab, ba, bb\}$

$\Sigma^3 = \{$

$\Sigma^0 = \{\varepsilon\}$ epsilon

Σ^i: set of all strings over Σ which are having length 'i'

Σ^1:

Σ^0: $\{\varepsilon\}$ set of all strings over Σ which are having length 0.

Klene closure and positive closure:-
(*) (+)

Σ^*, Σ^+ — Σ^* 0 or more elements are indicated.
Klene Positive. — Σ^+ 1 or more elements are indicated.

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

$\Sigma^* = \Sigma^0 . \Sigma^1 . \Sigma^2 ....$

$= \{\varepsilon, a, b, aa ....\}$

$\Sigma^+ = \Sigma^1 . \Sigma^2 . \Sigma^3 ....$

$= \{a, b, aa, ...\}$ except ε everything is included.

$A = \{a, b\}$ & $B = \{c, d\}$

$A \cup B = \{a, b, c, d\}$

$A . B = \{ac, ad, bc, bd\}$ Here "." indicates concatenation operation

$A \cap B = \{\phi\}$

$A - B = \{a, b\}$

Power set of $A = \{\phi, \{a\}, \{b\}, \{a, b\}\}$

No. of elements in Power set of A is $2^{\text{Cardinality of } A}$

$2^{|A|} = 2^2 = 4$

$A \subseteq B$ A is subset of B

$A \times B = \{(a,c)\ (a,d)\ (b,c)\ (b,d)\}$

$A \times B = \{(a,b) / a \in A,\ b \in B\}$

Language is a subset of Σ^*

$L_1 =$ finite L_a

$L_2 =$ finite L

$L_3 =$ infinite

→ indicates everything along with 8

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

$L_1 = \{w /$ 'w' having length 1$\}$ } finite language

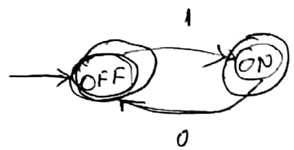
$L_2 = \{w /$ 'w' having length 2$\}$

$L_3 = \{w /$ 'w' where starts with 'a'$\}$ } infinite language

FSM and Automata

Ex

(1) Switch:-



States: ON, OFF

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

The initial state in FSM is represented by an arrow mark.

◎ → Final States

H.W.

Draw the Automata for vending machine.

ഓം നമഃ ശിവായ

Day - 3

Coffee Vending Machine -

(i) Automaton Representation

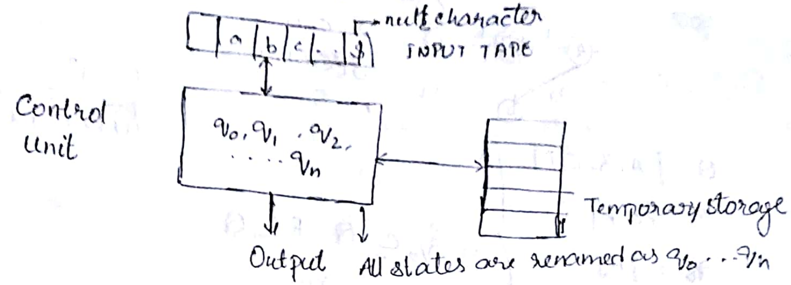
(ii) Need of Automaton

(iii) Types
- Acceptor
- Transducer

(iv) Tuple representation

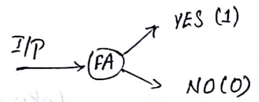
(v) Example Automata

Main Parts of Automata



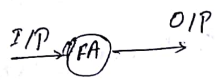
(i) Input Tape
(ii) Control Unit
(iii) Output
(iv) Temporary storage

Acceptor :- Gives an O/P of either YES (1) or NO (0)



Eg:- ~~etc~~ Digital ~~logic~~ lock.

Transducer :- Gives the O/P in the form of data like sum of 2 nos or product etc.



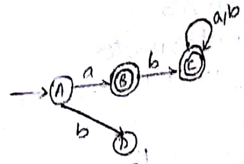
In the form of data

Eg:- Compiler, Interpreter

Tuple representation :-

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

$Q \rightarrow$ Set of finite states
$\Sigma \rightarrow$ Set of symbols
$\delta \rightarrow$ Transition function
$q_0 \rightarrow$ Starting states
$F \rightarrow$ Set of Final States
$M \rightarrow$ Machine



$\delta(C,a) = C$
$\delta(C,b) = C$

$\int \frac{x^2}{(1+x)^2 + x} \, dx$

$Q = \{A, B, C, D\}$
$\Sigma = \{a, b\}$
$q_0 = \{A\}$
$F = \{B, C\}$

$\Rightarrow q_0 \subseteq Q \quad F \subseteq Q$

δ: Transition function can be defined $\ni$

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

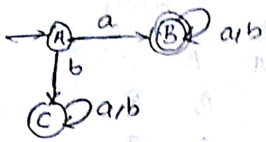
$\begin{pmatrix} A \\ B \\ C \\ D \end{pmatrix} \times \begin{pmatrix} a \\ b \end{pmatrix} \longrightarrow \begin{pmatrix} A \\ B \\ C \\ D \end{pmatrix}$

$\delta(A,a) = B$ $\quad$ $\delta(A,a)$ means A on taking input A [illegible] what state.

Types of finite Automata:-
(i) DFA Deterministic
(ii) NFA Non Deterministic
(iii) ϵ-NFA

Computers generally follow deterministic approach as they produce output in a determined manner.

DFA



$Q = \{A, B, C\}$

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

$q_0 = \{A\}$

$F = \{B\}$

1, 2, ~~4~~,

26, 32, 41, 55, 57

61, 65,

12th July, 2018.

[illegible]

Day - 4

Plastic, — 4

19

One 'a' or one 'b'

Zero or more a's

Atleast one a $\Sigma = \{a\}$

Atleast ~~the~~ one a $\Sigma = \{a, b\}$

Exactly one 'a' $\Sigma = \{a, b\}$

DFA:-

String acceptance + Transition table

Language acceptance

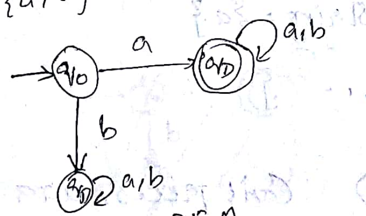
Eg:-

Empty String

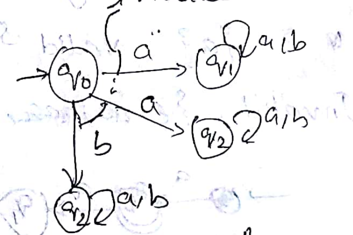
Length '2' $\Sigma = \{a, b\}$

Exactly one a $\Sigma = \{a\}$

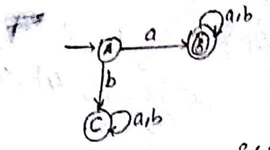
$\Sigma = \{a, b\}$



is a DFA



not a DFA



Transition Table

State	Symbols a	b
→ A	B	C
* B	B	B
C	C	C

$\delta(A,a) = B \quad \delta(B,a) = \delta(B,b) = B$

$\delta(A,b) = C \quad \delta(C,a) = \delta(C,b) = C$

'*' indicates final state

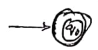
Q: Valid strings of len = 2 = $\{aa, ab, ba, bb\}$, ... aab, bbb, bab, ...

Invalid strings of ~~len 2~~ = $\{\varepsilon, a, b, aaa,$ ...

A string is said to be accepted when we start from the initial state and after some transitions we reach a final state.

DFA for accepting ε :-

1)



(i) Exactly one a $\Sigma = \{a\}$

Valid strings = $\{a\}$ Invalid strings = $\{\varepsilon, aa, aaa, \dots\}$

~~DFA~~

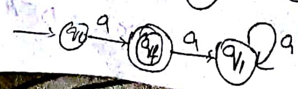
Complete DFA must accept and reject the strings. It should be able to reject ~~the~~ all the invalid string by completely scanning them.

Q. $\Sigma = \{a\}$ Valid String = $\{a\}$

Invalid = $\{\varepsilon, aa, aaa, \dots\}$

Can't process aaa

∴



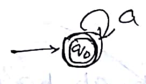
Complete DFA

(ii) Zero or more a's

$\Sigma = \{a\}$ Valid Strings = $\{\varepsilon, a, aa, \ldots\}$

Invalid Strings : $\{\}$

Obtain the least len string from valid strings and do a skeleton diagram for it then you can extend it to other valid strings.

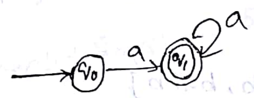


Difference b/w Language Acceptance and String Acceptance:-

(iii) Atleast one 'a' $\Sigma = \{a\}$

Valid Strings = $\{a, aa, aaa, \ldots\}$

Invalid Strings = $\{\varepsilon\}$

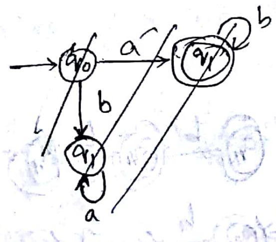


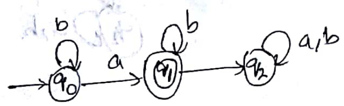
For ε to be accepted, the starting state must be a final state.

(iv) Exactly one 'a' over $\Sigma = \{a, b\}$

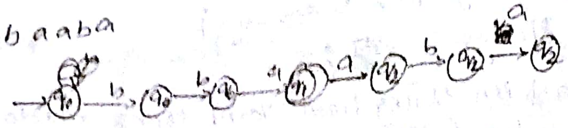
Valid Strings = $\{a, ab, abb, abbb, \ldots$ ba, bab $\}$

Invalid Strings = $\{\varepsilon, aab, aba, baa, \ldots\}$



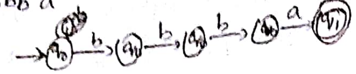


bb a a b a



(Invalid) as q_2 isn't final

bbb a

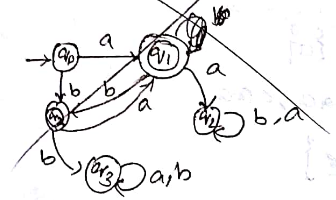


(Valid) as q_1 is final

(v) One a or One b

$\Sigma = \{a, b\}$ Valid Strings = { a, b, ab, ba }

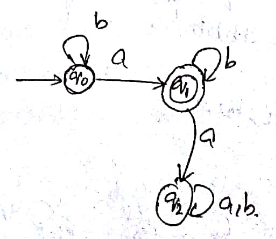
Invalid Strings : {ε, abb, baa, abab, abbba, ...}



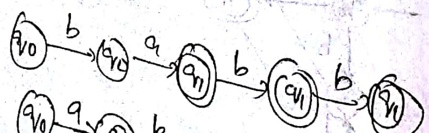
(vi) Exactly one a over $\Sigma = \{a, b\}$

Valid Strings : {a, ab, abb, ba, bab, babb}

Invalid Strings = {ε, aa, aba, abba, abbab, abbaab, ...}

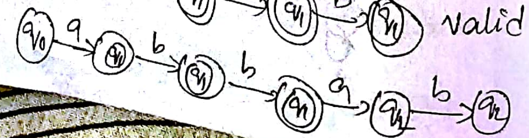


babb



Valid

abb ab

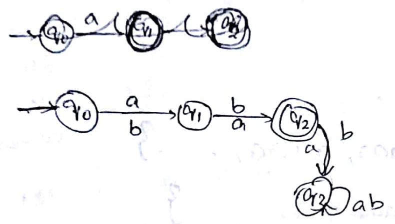


Invalid

(vii) Length 2' ~~we~~ $\Sigma = \{a, b\}$

Valid Strings = { ab, ba, aa, bb }

Invalid Strings = { ε, a, b, abb, bba, aaa, bbb, bba, ... }



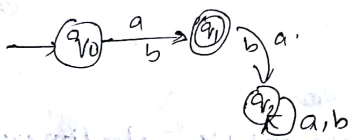
~~(viii)~~ ~~Ex~~

(v) ~~Sansel's Method i.e~~ $\Sigma = \{a, b\}$

One a or One b ⇒ Exactly one length a and Exactly one length b

Valid Strings = { a, b }

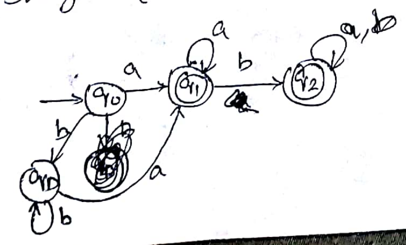
Invalid Strings = { ε, ab, abb, ... }



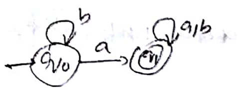
(viii) Atleast one a over $\Sigma = \{a, b\}$

Valid Strings = { a, aab, aaa, aa, ba, ab

Invalid Strings = { ε, bb, b, bbb, ... }



Alternative :-



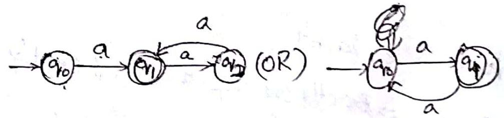
Construct a DFA which ~~can~~ accepts strings that contain

(ix) odd no. of a's

(x) even no. of a's

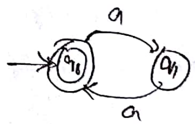
ix) Valid Strings = { a, aaa, aaaaa, ... }

Invalid Strings = { ε, aa, aaaa, . . . }



x) Valid Strings = { aε, aa, aaaa, ... }

Invalid Strings = { a, aaa, aaaaa, . . . }



) DFA to accept strings of a's and b's starting with string ab

 Valid Strings = { ab, abaa, abbb, ababab, abbb

13th July, 2018.

a) Starts with

b) ends with

c) contains 'a

starts with

d) ends with

e) contains

f) Three

a) Valid Str

Invalid

b) ends

Va

In

Alt

5th July, 2018.

ശ്രീ (Malayalam heading)

Day - 5

Starts with 'a' $\Sigma = \{a,b\}$

ends with 'a' $\Sigma = \{a,b\}$

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

starts with 'ab' $\Sigma = \{a,b\}$ → H.W. (xi)

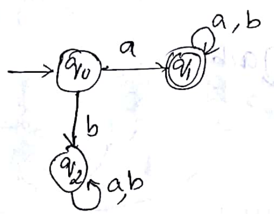
ends with 'ab' $\Sigma = \{a,b\}$

contains 'ab' $\Sigma = \{a,b\}$

Three consecutive 0's over $\Sigma = \{0,1\}$

Valid Strings = {a, ab, aa, abb, aaab, ... }

Invalid Strings = {ε, b, bb, ba, baa, bbb, ... }

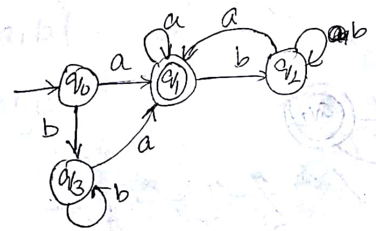


ends with a

Valid Strings = {a, ba, bba, aaa, aa, aabba, ... }

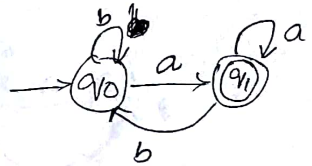
Invalid Strings = {ε, ab, abb, aab, abab, aabb, ... }

aabbb abababb



ba

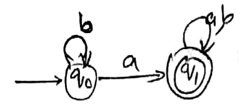
Alternate :- Minimized DFA



c) Contains 'a' $\Sigma = \{a, b\}$ $\{$a, ab, ba, baa, bab, abb, ... $\}$

Valid Strings = $\{$a, ab, ba, baa, bab, abb, ... $\}$

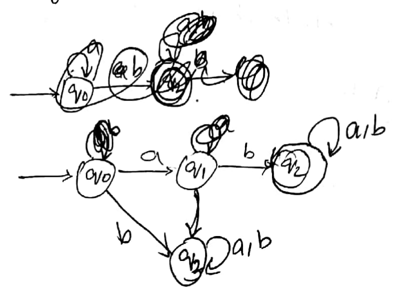
Invalid Strings = $\{\varepsilon$, b, bb, bbb, ... $\}$



Q(i) Starts with ab $\Sigma = \{a, b\}$ H.W.

Valid Strings = $\{$ ab, aba, abb, ababa, abb, ... $\}$

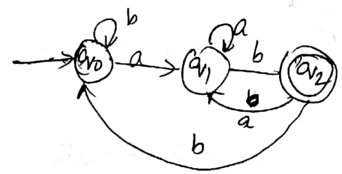
Invalid Strings = $\{\varepsilon$, a, aa, b, bb, baab; $\}$



d) ends with ab $\Sigma = \{a, b\}$

Valid Strings = $\{$ ab, bab, bbab, aaabab, ... $\}$

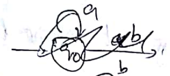
Invalid Strings = $\{\varepsilon$, ba, bba,

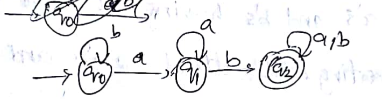


e) Contains ab $\Sigma = \{a,b\}$

Valid Strings = { ab, bab, bbab, abab, . . . }

Invalid Strings = { ε, a, aa, ~~abba~~, bba, ba, b, bb, bbb, . . . }

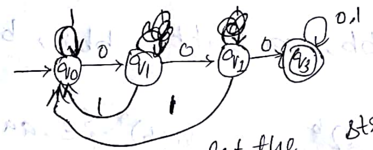




f) Three consecutive 0's over $\Sigma = \{0,1\}$

Valid Strings = { 000, 1000, 0001, 0000, 1010000, . . . }

Invalid Strings = { ε, 0, 1, 00, 11, 10, 111, 1010, . . . }



0101001001

Note :- Don't let the string to ascend to the next state in consecutive 0's. Make the state from starting

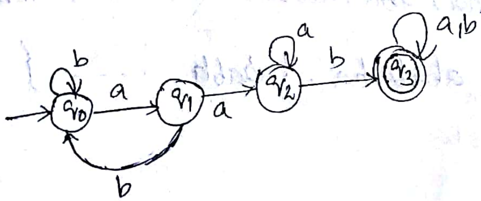
H.W.

in consecutive 0's. Make the state from starting DFA to accept strings of a's and b's having a substring aab.

g) DFA to accept strings of a's and b's having a substring aab.

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

Valid Strings = {



17th July, 2018.

ഓം നമഃശിവായ

Day-6

1) DFA to accept string of a's and b's ∋

$L = \{ awa \mid w \in (a+b)^n$ where $n \geq 0 \}$

2) DFA to accept string of a's and b's having not more than 3 a's

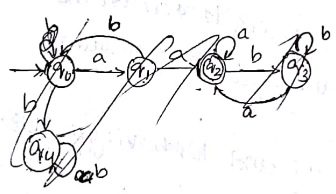
3) Strings 0's and 1's starting with atleast 2 0's and ending with atleast 2 1's

4) DFA to accept odd no. of b's and even no. of a's.

1) a+b means ~~either or~~ combination of a or b

Valid Strings = { ~~aa~~, aa, aaa, aba, ababa, abbba, abba bab ...

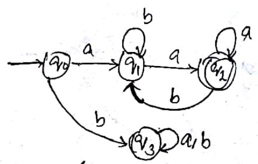
Invalid Strings = {ε, aab, abbb, ~~aa~~ babba, bbb, baaa, ...



$(a+b)^* = \{ \varepsilon, aa, ab, ba \}$

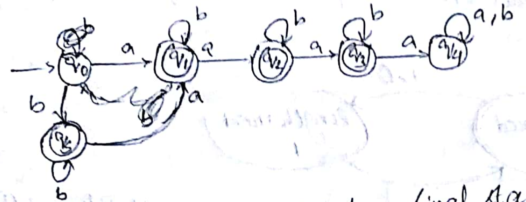
$(ab)^* = \{ \varepsilon, ab, abab, ababab \}$

$ba \notin (ab)^*$



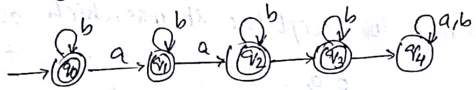
2) Valid Strings = {ε, b, a, aa, aaa, aba, abbba, aabbba, abbaa, ...

Invalid Strings = {~~ε~~, aaaa, abababa, babaaba, ... }



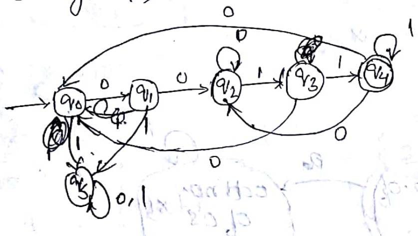
As ε is also valid first state = final state

Minimizing the above DFA



3) Valid Strings = { 0011, 000011,

Invalid Strings = { ε 0, 1, 01, 10, 11, 101 011, 001, 010, ...



010011

00.....11

length % 2 = 0

q_0 : len % 2 = 0   If the length is ...

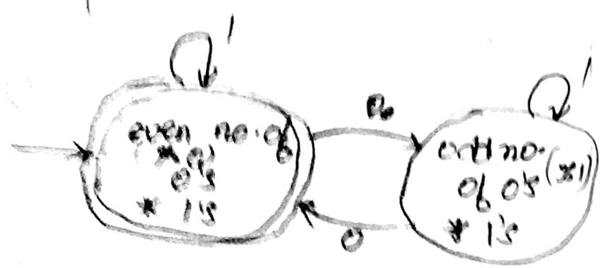
q_1 : len % 2 = 1

only 2 states ...

Q) $\Sigma = \{0,1\}$ Construct a DFA to accept the string ... no. of zeroes.

Valid = {10, 00, 0000, 000000, ε, ...}

No. of 0's	No. of 1's
even	*
odd	*



\# Odd no. of b's and even no. of a's

a	b
[illegible]	[illegible]

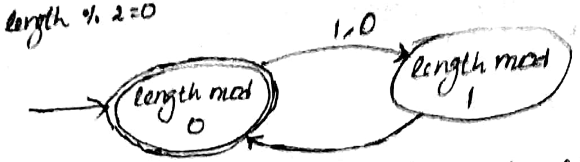
ഓം നമഃ ശിവായ

Day 7

20th July, 2018.

Q5) $\Sigma = \{0,1\}$. Construct DFA to accept even no. of symbols (length ...

length % 2 = 0



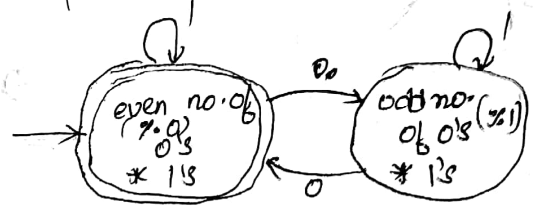
q_0 : len % 0   As the length % 2 has 2 possibilities either even / odd ∴ Only 2 states are needed.

q_1 : len % 1

Q6) $\Sigma = \{0,1\}$. Construct a DFA to accept the strings which contain even no. of zeroes.

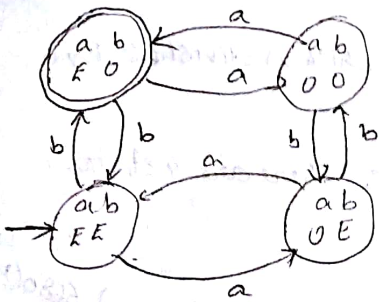
Valid = {1, 00, 0000, 000000, ε, 010, ...}

No. of 0's	No. of 1's
even	*
odd	*



4) Odd no. of b's and even no. of a's

a	b	
even	odd	
E	O	aab
E	E	aabb
O	O	ab
O	E	abb



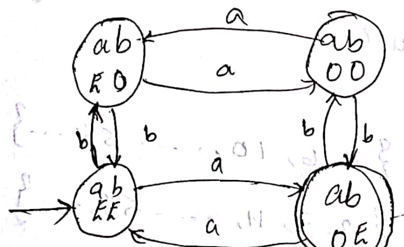
$\epsilon \to$ even a's
even b's

$\therefore$ EE is starting state

aab

Valid → babab
Invalid → abababab

1) Construct a DFA to accept strings even no. of b's and odd no. of a's.

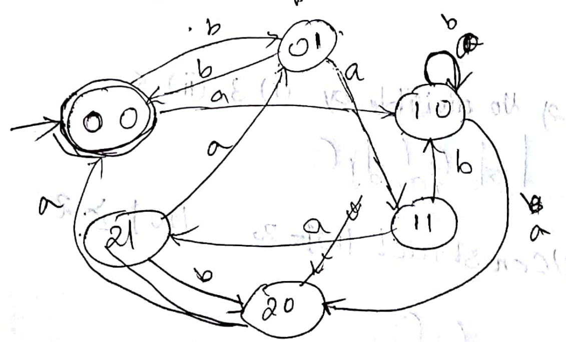


a	b	
E	E	✗
E	O	✗
O	E	✓
O	O	✗

3) Set of all strings $\Sigma = \{a, b\}$ in which no. of a's is divisible by 3 and b's divisible by 2

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

a % 3	b % 2	
0	0	✓
0	1	
1	0	
1	1	
2	0	
2	1	



ϵ no. of a's → $0 \% 3 = 0$
no. of b's → $0 \% 2 = 0$

~~baa~~ ba, bb, baa, bab
babb
a = 1
b = 0

2%3 = 2
2%2 = 0

baab
baaba 00
baabb

ഓം നമഃ ശിവായ

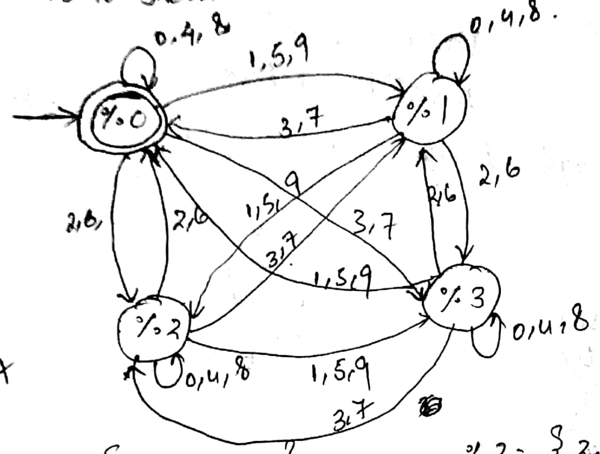
21st July .2018.

Day-8

1) Draw a DFA to accept a decimal no. which is divisible by 4

$\Sigma = \{0, 1, 2, \ldots 9\}$

as no. should be divisible by 4 ∴ There are 4 states.



6B7

%0 = {0, 4, 8, 48...}

%1 = {1, 5, 9, ... 53...}

%2 = {2, 6, 10, ... 54...}

%3 = {3, 7, 11, ... 55, ...}

4) 5680

5680

2) No. divisible by (i) 3 (ii) 5

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

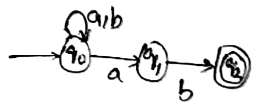
3) Construct $|w| = 2$ $|w| \geq 2$ $|w| \leq 2$

26th July, 2018.

[illegible]

Day-9

$\Sigma = \{a,b\}$ Strings ending with 'ab' construct an NFA?



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

Transition Table NFA

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

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

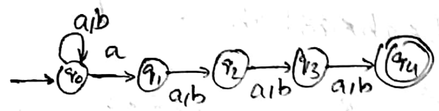
Here $Q = \{q_0, q_1, q_2\}$

$2^Q = \{\{q_0\}, \{q_1\}, \{q_2\}, \{q_0 q_1\}, \{q_1, q_2\}, \{q_2, q_0\}, \{q_0\, q_1, q_2\}\}$

Construct an NFA $\Sigma = \{a,b\}$. The language is 4th letter from the last character is 'a'.

$- - - - \; a \; - - -$

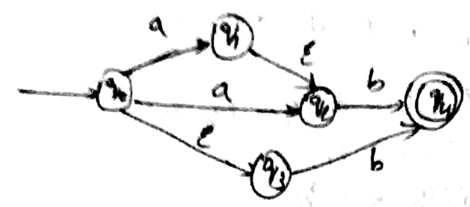
$V = \{aaaa, abbb, abab, \ldots\}$ $I = \{\varepsilon, a, aa, aaa, baba, \ldots\}$



$\underline{a}\ \underline{a/b}\ \underline{a/b}\ \underline{a/b}$

Here whatever may be the len of string we are scaling down the string to last 4 letters. So whatever a/b before the last 4th character it will be intelligently deciding to be in dead state. The last 4th character is our initial state.

ε-NFA



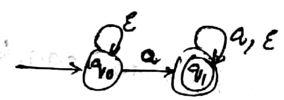
NFA	ε-NFA
$\delta : Q \times \Sigma \rightarrow 2^{Q}$	$\delta = Q \times \Sigma \cup \{\varepsilon\}$ to 2^{Q}

Transition function

Draw ε-NFA's

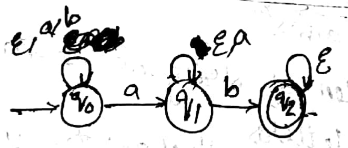
1) Construct ε-NFA for the language all strings ending with :

$\Sigma = \{a, \varepsilon\}$ $V = \{a, \varepsilon a, \varepsilon\varepsilon a, a\varepsilon a, a/\varepsilon\ a/\varepsilon \ldots a$



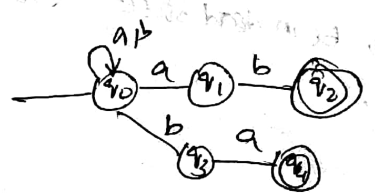
2) $\Sigma = \{a, b, \varepsilon\}$ ending with ab?

$V = \{\varepsilon ab, ab, \varepsilon ab\varepsilon ab\varepsilon a\varepsilon bab, \ldots\}$



$\varepsilon ab\varepsilon\varepsilon\varepsilon\varepsilon$ --a_b--

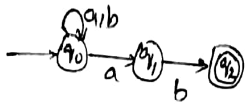
3) $\Sigma = \{a, b\}$ NFA to accept ab or ba?



26th July, 2018. ഓം നമഃ ശിവായ

Day-9

$\Sigma = \{a, b\}$ Strings ending with 'ab' construct an NFA?



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

Transition Table NFA

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

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

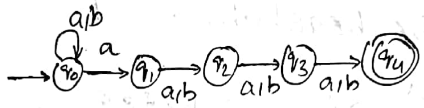
Here $Q = \{q_0, q_1, q_2\}$

$2^Q = \{\{q_0\}, \{q_1\}, \{q_2\}, \{q_0 q_1\}, \{q_1, q_2\},$
$\{q_2, q_0\}, \{q_0, q_1, q_2\}\}$

Construct an NFA $\Sigma = \{a, b\}$. The language is 4th letter from the last character is 'a'.

- - - - . (a) - - -

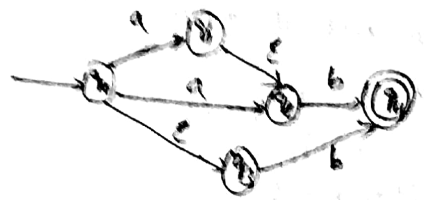
V = { aaaa, abbb, abab, ... } $I = \{\epsilon, a, aa, aaa, baba, ..\}$



a a/b a/b a/b

Here whatever may be the len. of string we are scaling down the string to last 4 letters. So whatever a/b before the last 4th character it will be intelligently deciding to be in dead state. The last 4th character is our initial state.

ε-NFA



NFA

ε-NFA

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

Transition function

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

Draw ε-NFA's

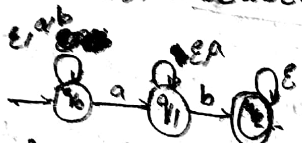
1) Construct ε-NFA for the language all strings ending with a

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

$V = \{a, \epsilon a, \epsilon\epsilon a, a\epsilon a, a/\epsilon\ a/\epsilon \ldots a$

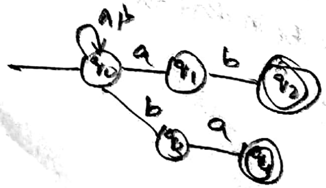
2) $\Sigma = \{a, b, \epsilon\}$ ending with ab?

$V = \{\epsilon ab, ab, \epsilon ab\epsilon ab\epsilon a\epsilon bab, \ldots\}$



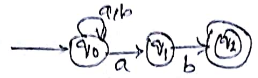
εabεεεε — —a_b—

3) $\Sigma = \{a, b\}$ NFA to accept ab or ba?



Extended Transition function:-

Draw the NFA with strings ending with ab ?



$\hat{\delta} : Q \times \Sigma^* \longrightarrow 2^Q$ $V = \{aaab\}$ $T = \{abaa\}$ aaab

Transition Table

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

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

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

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

$= \{q_0 \cup q_2\}$

$\hat{\delta}(Q, w)$

$= \{A\}$

$\delta(A, a)$

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

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

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

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

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

$= \{q_0, q_2\}$

H.W.

babab

$\hat{\delta} \quad \delta$

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

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

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

$= \{q_0 q_1\} \cup \{\phi\}$

$= \{q_0 q_1\}$

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

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

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

$= \{q_0 q_1\}$

As $q_2 \notin \{q_0, q_1\}$

$\therefore$ abaa isn't valid.

babab

$\delta(q_0, b) = \{q_0\}$

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

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

$\hat{\delta}(q_0, bab) = \delta(\hat{\delta}(q_0, ba), b)$

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

$= \{q_0, q_2\}$

$\hat{\delta}(q_0, baba) = \delta(\hat{\delta}(q_0, bab), a)$

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

$= (q_0, q_1) \cup \phi$

$= \{q_0, q_1\}$

$\hat{\delta}(q_0, babab) = \delta(\hat{\delta}(q_0, baba), b)$

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

$= \{q_0, q_2\}$

As q_2 (final state) $\in \{q_0, q_2\}$.

$\therefore$ The string babab is a valid string.

ഓം നമഃ ശിവായ

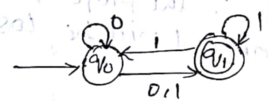
ϵ- closure of q
Set of all nodes p $\ni$ there is a path from $q \to p$ labelled ϵ.

ϵ- closure of $A = \{A, B, D\}$
" $B = \{B, D\}$
" $C = \{C\}$
" $D = \{D\}$

All are having equal power whether it is DFA, NFA or ϵ-NFA

(i) NFA to DFA Conversion:-
a) Subset construction method
b) Lazy method → better approach.

Consider the following NFA and convert into equivalent DFA



NFA Transition table: → DFA table:

δ	0	1
→ q_0	$\{q_0,q_1\}$	q_1
* q_1	ϕ	$\{q_1,q_0\}$

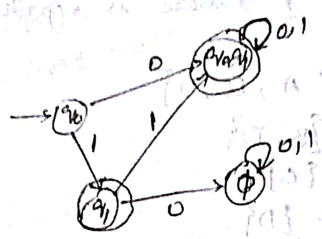
	δ	0	1
(A)	→ q_0	$\{q_0,q_1\}$	q_1
(B)	* $\{q_0,q_1\}$	$\{q_0,q_1\}$	$\{q_0,q_1\}$
(C)	* q_1	ϕ	$\{q_1,q_0\}$
(D)	ϕ	ϕ	ϕ

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

$\delta(\{q_0,q_1\}, 1) = \delta\{q_0, 1\} \cup \delta(q_1, 1) = q_1 \cup \{q_0,q_1\} = \{q_0,q_1\}$

If final state of NFA table is present in any set of δ in DFA table then mark it as final state.

A	B	C
B	B	B
C	D	B
D	D	D



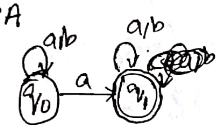
0011 → V

$\Sigma = \{a,b\}$ construct NFA to accept string $\ni$ 2nd symbol from right hand side is a

convert that NFA to DFA

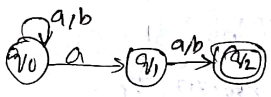
a/b a/b --- a a/b

NFA



This is wrong $\because$ the property that a is 2nd symbol from last will be lost $\therefore$

consider an additional state.



NFA transition table

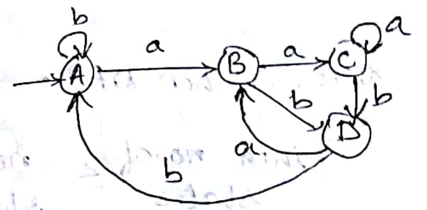
δ	a	b
→ q_0	$\{q_0,q_1\}$	q_0
q_1	q_2	q_2
* q_2	ϕ	ϕ

DFA Transition table

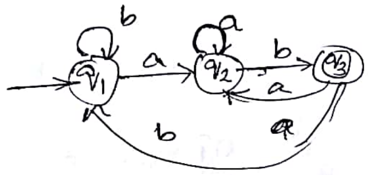
	δ	a	b
(A) →	q_0	$\{q_0,q_1\}$	q_0
(B)	$\{q_0,q_1\}$	$\{q_0,q_1,q_2\}$	$\{q_0,q_2\}$
(C) *	$\{q_0,q_1,q_2\}$	$\{q_0,q_1,q_2\}$	$\{q_0,q_2\}$
(D) *	$\{q_0,q_2\}$	$\{q_0,q_1\}$	q_0

δ	a	b
A	B	A
B	C	D
C	C	D
D	B	A

60,67,12,15



Convert the following NFA to DFA

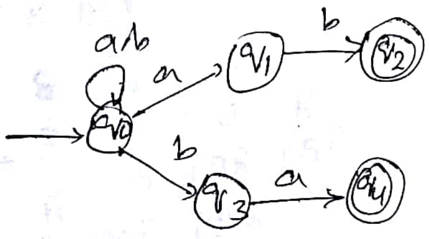


DFA transition table

NFA Transition table

δ	a	b
$\rightarrow q_1$	q_2	q_1
q_2	q_2	q_3
$* q_3$	q_2	q_1

Consider the following NFA and convert it into DFA. Describe the language i.e. accepted by this automata?

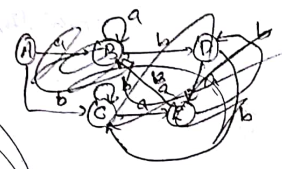


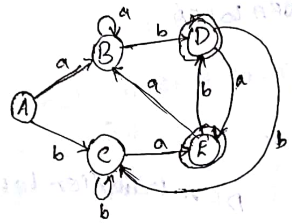
NFA transition table

δ	a	b
$\rightarrow q_0$	q_0q_1	q_0q_3
q_1	ϕ	q_2
$* q_2$	ϕ	ϕ
q_3	q_4	ϕ
$* q_4$	ϕ	ϕ

DFA transition table

	δ	a	b
A	$\rightarrow q_0$	q_0q_1 B	q_3q_0 C
B	q_0q_1	q_0q_1 B	$q_0q_2q_3$ D
C	q_0q_3	$q_0q_1q_4$ E	q_0q_3 C
D	$* q_0q_2q_3$	$q_0q_1q_4$ E	q_0q_3 C
E	$* q_0q_1q_4$	q_0q_1 B	$q_0q_2q_3$ D



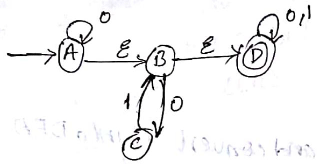


$3^{n}, 3^{m} \geq 1$

Note :- When NFA ⟶ DFA

then no. of states NFA $\leq$ no. of states DFA

ε-NFA to NFA Conversion :-



A $\quad \underline{\varepsilon^*} \quad 0 \quad \underline{\varepsilon^*}$

B

write down ε-closures of all

δ	0	1
→ A	{ABCD}	D
B	{CD}	BD
C	ϕ	BD
* D	D	D

H.W. Consider following ε-NFA

(i) Compute ε-closure of each state

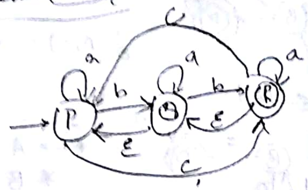
(ii) Give all strings of length ≤ 3 accepted by automata

(iii) Convert automata to DFA. (ε-NFA → NFA → DFA)

S	ε	a	b	c
→p	φ	p	q	r
q	p	q	r	φ
*r	q	r	φ	p

Question

1



	a	b	c
	P	Q	R
P	PQ	PQR	PQR
Q	PQR	QR	PR
R			

ε-closure of P = {P}
Q = {P, Q}
R = {P, Q, R}

δ	a	b	c
	P	Q	R
→P			PQR
Q	{PQ}	PQR	
R	{PQR}	PQR	QR

δ	a	b	c
	P	Q	R
→P	PQ	PQR	PQR
Q	PQR	PQR	PQR
R	PQR	PQR	PQR

P → P → a → P
Q → P → a → P → P
Q → a → Q → Q
Q → P → b → Q → R
Q → Q → b → R
Q → P → c → R
Q → Q → c → φ
R → P
R → Q
R → R

P ε* P → P → a → ? → P

Q → P → a → P → P
Q → Q → a → Q → R, Q

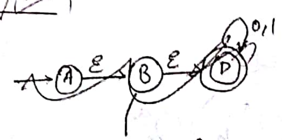
Q → P → b → Q → P, Q
Q → Q → b → R → P, Q, R

31st July, 2018.

[illegible]

Day-11

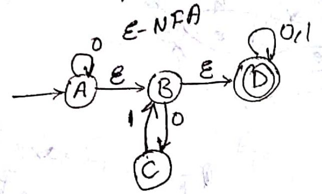
~~ε-NFA~~



$A \rightarrow \varepsilon^* 0 \varepsilon^*$

$\hat{\delta}(A,0) = \varepsilon\text{-closure}(\delta(\varepsilon\text{-closure}(A,0)))$

ε-NFA



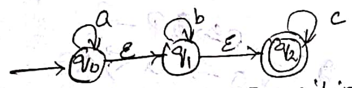
	0	1
$*\rightarrow A$	{A,B,C,D}	D
$*$ B	{C,D}	{B,D}
C	ϕ	{B,D}
$*$ D	D	D

NFA Transition Table

In order to fill NFA Transition table. take $A \rightarrow \varepsilon^* 0 \varepsilon^*$

Conversion of ε-NFA to DFA :—

When you convert ε-NFA → ~~DFA~~ NFA No. of final states ↑ but no. of states ~~are~~ will remain same.



DFA Table

a b c

ε-NFA Transition Table

	a	b	c	ε
$\rightarrow q_0$	q_0	ϕ	ϕ	q_1
q_1	ϕ	q_1	ϕ	q_2
$* q_2$	ϕ	ϕ	q_2	ϕ

Starting state of DFA = ε-closure (starting state of curren ε-NFA)

DFA Starting state = ε-closure(q_0) = $\{q_0, q_1, q_2\}$

DFA Transition Table

	a	b	c
[A] $\rightarrow * q_0 q_1 q_2$	[A] $q_0 q_1 q_2$	[B] $q_1 q_2$	q_2 [C]
$*$ [B] $q_1 q_2$	ϕ [D]	$q_1 q_2$ [B]	q_2 [C]
$*$[C] q_2	ϕ [D]	ϕ [D]	q_2 [C]
[D] ϕ	ϕ [D]	ϕ [D]	ϕ [D]

1st entry:

ϵ-closure$(\delta(q_0 q_1 q_2), a) = \epsilon[\delta(q_0, a) \cup \delta(q_1, a) \cup \delta(q_2, a)]$

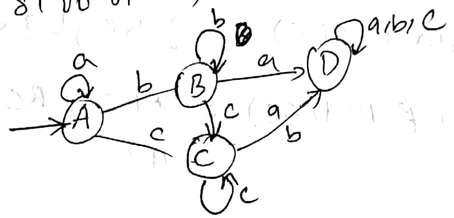
$= \epsilon[q_0\ \phi\ \phi] = \epsilon(q_0) = \{q_0, q_1, q_2\}$

2nd entry:

ϵ-closure $(\delta(q_0 q_1, q_2), b) = \epsilon[\phi, q_1, \phi] = \epsilon(q_1) = \{q_1, q_2\}$

3rd entry:

$\epsilon(\delta(q_0 q_1 q_2)\ c) = \epsilon[\phi, \phi, q_2] = \epsilon(q_2) = \cancel{\phi}\ q_2$



4th entry:

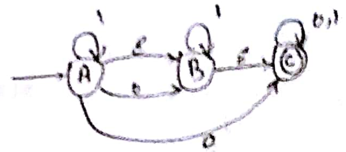
$\epsilon[\delta(q_1, a) \cup \delta(q_2, a)] = \epsilon[\phi, \phi] = \epsilon(\phi) = \phi$

$\epsilon[\delta(q_1, b) \cup \delta(q_2, b)] = \epsilon[q_1] = \{q_1, q_2\}$

$\epsilon[\delta(q_1, c) \cup \delta(q_2, c)] = \epsilon[\phi, q_2] = \{q_2\}$

As q_2 is FS in NFA so what all states contain q_2 they are final states in DFA

Convert the ε-NFA to DFA.



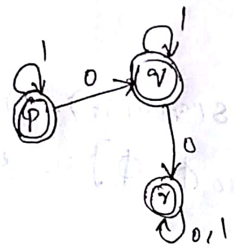
ε-NFA Transition table

	0	1
→ A	B	A
B	ϕ	B
* C	C	C

Starting state DFA: $\varepsilon(A) = \{A, B, C\}$

DFA Transition Table

	0	1
→ $\overset{*}{ABC}$ p	BC q	ABC p
* BC q	C r	BC q
* C r	C r	C r



1st entry: $\varepsilon(\delta[A,0] \cup \delta[B,0] \cup \delta[C,0])$

$= \varepsilon\{BC\} = \varepsilon(B) \cup \varepsilon(C) = \{BC\} \cup \{C\}$

$\varepsilon(A, BC) = \varepsilon(A) \cup \varepsilon(B) \cup \varepsilon(C) = ABC \cup BC \cup$

$= ABC$

2nd row

$\varepsilon(\phi \cup C) = \varepsilon(C) = C$

$\varepsilon(B \cup C) = BC \cup C = BC$

3rd row

$\varepsilon(C) = C$

$\varepsilon(C) = C$

2nd August 2018.

Minimizatio

Myhill N

language

If langu

It can be

check wh

DFA mi

(i) Pred it

(ii) Table

(i) B

a) Del

1) Tw

if

2nd August, 2018.

ఓం నమః శివాయ

Day-12

Minimization of DFA:-

Myhill Nerode Theorem:—

For a language L, defined over an alphabet Σ, the language L partitions ~~into~~ Σ^* into different classes.

If language L generates finite no. of classes then L is regular.

It can be used for DFA minimization and ~~language~~ check whether language L is regular or not.

DFA minimization can be done in 2 ways:-

(i) Partition method

(ii) Table filling method

(i) ~~Bee~~

a) Delete the unreachable states and corr. transitions.

1) Two states are said to be equal

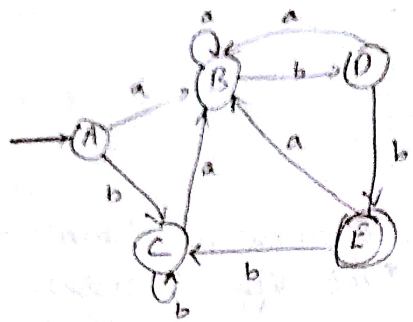
if $\delta(p, w) \in F$, $\delta(q, w) \in F$

(OR)

$\delta(p, w) \notin F$, $\delta(q, w) \notin F$

then $p \equiv q$

2)



DFA transition table

δ	a	b
→ A	B	C
B	B	D
C	B	C
D	B	E
* E	B	C

0- equivalence => |w| = 0

[ABCD] [E]
Non-final Final

1- equivalence => |w| = 1

[ABC] [D] [E]

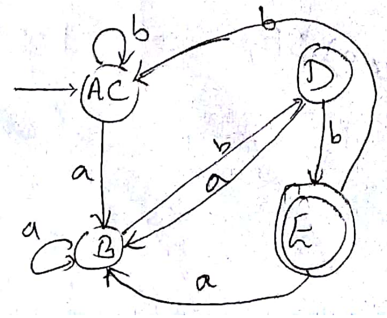
2- equivalence => |w| = 2

[AC] [B] [D] [E]

3- equivalence => |w| = 3

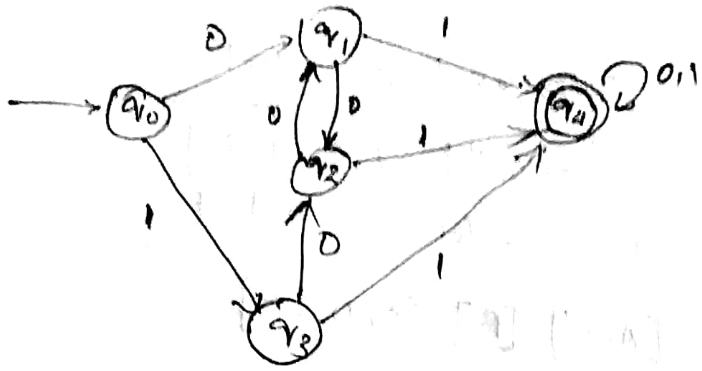
[AC] [B] [D] [E]

As these 2 states are equal ~~equa~~ stop here



3)

2)



~~DFA minimization~~

DFA Transition table:-

	0	1
→ q_0	q_1	q_3
q_1	q_2	q_4
q_2	q_1	q_4
q_3	q_2	q_4
* q_4	q_4	q_4

0-equivalence => $|w| = 0$

$[q_0 q_1 q_2 q_3]$ $[q_4]$

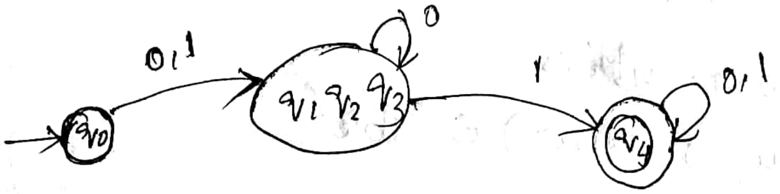
1-equivalence => $|w| = 1$

~~$[q_0]$ $[q_1]$ $[q_2]$ $[q_3]$ $[q_4]$~~

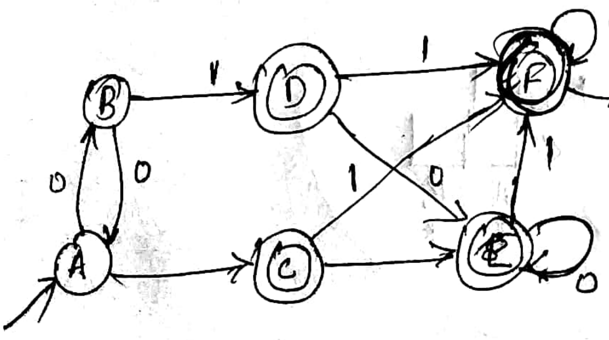
$[q_0]$ $[q_1\ q_2\ q_3]$ $[q_4]$

2-equivalence => $|w| = 2$

$[q_0]$ $[q_1 q_2 q_3]$ $[q_4]$



3)



→ not a final state

DFA Transition table:-

	0	1
→ A	B	C
B	A	D
* C	E	F
* D	E	F
* E	E	F
F	F	F

0-equivalence => |w| = 0

[A, B, F] [C, D, E]

1-equivalence => |w| = 1

[A, B] [F] [C, D, E]

2-equivalence => |w| = 2

[A,B] [F] [C, D E]

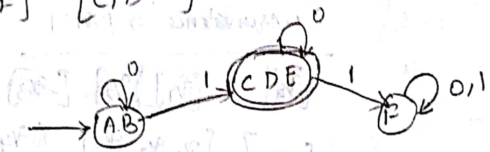


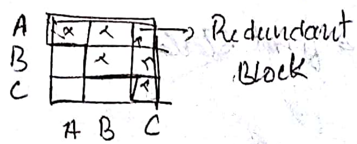
Table Filling Algorithm:-

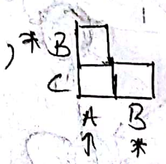
Prepare a n×n table where n → no. of states

Select a pair (a,b) where a ∈ F and b ∉ F

Then fill the corr. entries

as (a,b) ∈ Table => (b,a) ∈ Table.



2) 

Likewise for n states

In step 2 identify the unmarked entries

~~(A,B)~~ (F,A) (B,F) (C,E) (C,D) (D,E)

Now δ(F,A), 0 = F B ~~unmark~~ δ((F,A), 1) = FC

For the previous question DFA

Step 1

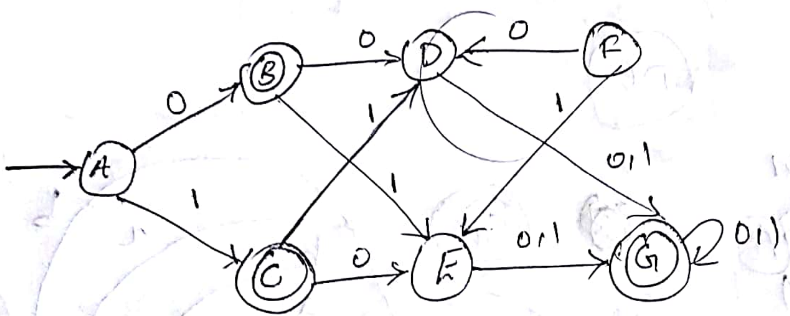
	A	B	C	D	E
B					
* C	X	X			
* D	X	X			
* E	X	X			
F	XX	XX	X	X	X

check whether FB or FG is marked or not. if any one is marked then FA can be marked.

$\delta((C,D),0) = EE$, $\delta((C,D),1) = FF$ } not possible

$\delta((C,E),0) = (E\,E)$ } not possible
$\delta(C,E,1) = (F\,F)$

1)



DFA Transition table:-

	0	1
→ A	B	C
* B	D	E
* C	E	D
D	G	G
E	G	G
* G	G	G

0-equivalence => $|w| = 0$

[A D E] [B C G]

1-equivalence => $|w| = 1$

[A D E] [B C] [G]

{BCG} {ADE}

(G,B) (D,A) (E,A) (E,D) (G...

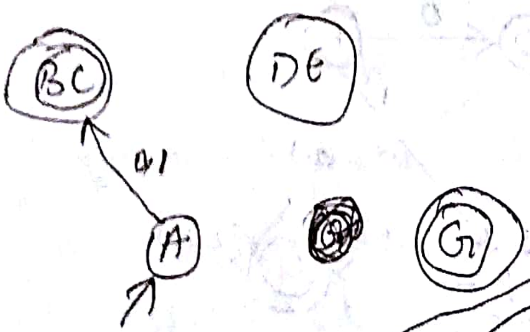
X X X X

	A	B	C	D	E
* B	x				
* C	x				
D	xxx	x	x		
E	xxx	x	x		
* G	x	xx	xx	x	x

3rd step :- AD AE BC ED

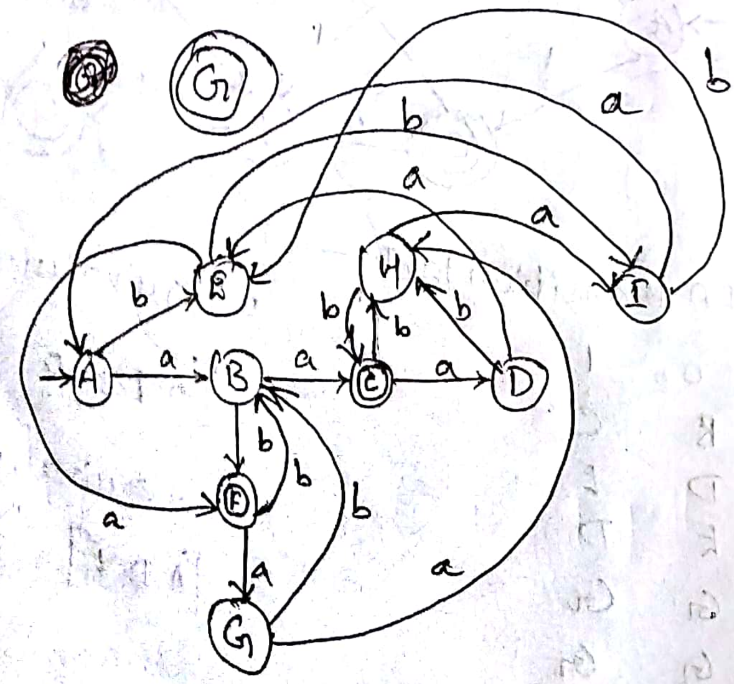
BG, BG BG CG DE ED GG GG

~~4th step :- BC → ED~~



DFA minimization :-

δ	a	b
→ A	B	E
B	C	F
* C	D	H
D	E	H
E	F	I
* F	G	B
G	H	B
H	I	C
* I	A	E



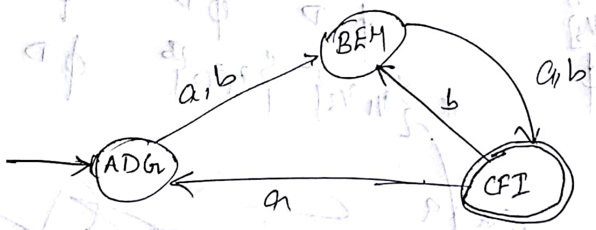
~~$Q = q_0 \ldots$~~

$[C, F, I]$ $[A, B, D, E, G, H]$

B	xx							
* C	x	x						
D		xx	x					
E	xx		x	xx				
* F	x	x		x	x			
G		xx	x		xx	x		
H	xx		x	xx		x	xx	
* I	x	x		x	x		x	x
	A	B	C	D	E	F	G	H

~~(A,B)~~
(A,D) ~~(B,D)~~
~~(A,E)~~ (B,E) ~~(D,E)~~
~~(C,F)~~
(A,G) ~~(B,G)~~ (D,G) ~~(E,G)~~
~~(A,H)~~ (B,H) ~~(D,H)~~ (E,H) ~~(G,H)~~
(C,I) (F,I)

CF, FI
EF HI, EH, HB



The unmarked pairs are

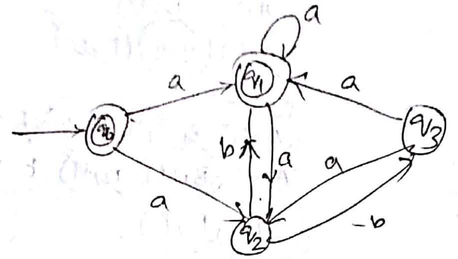
AD AG
BE BH
CF CI
DG, EH, FI

A D G
B E H
C F I

ADG
BEH
CFI

Tutorial 2

1) NFA ⟶ DFA

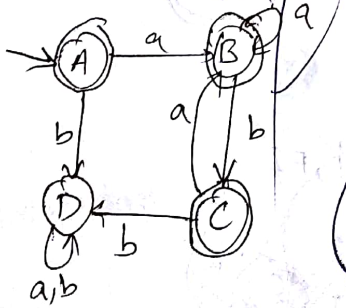


NFA Transition Table:-

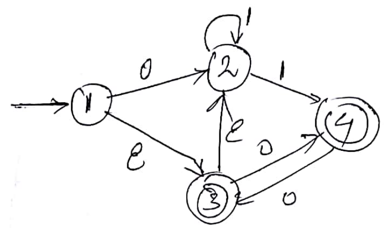
	a	b
*→ q_0	$\{q_1, q_2\}$	ϕ
* q_1	$\{q_1, q_2\}$	ϕ
q_2	ϕ	$\{q_1, q_3\}$
q_3	$\{q_1, q_2\}$	ϕ

DFA Transition Table

	a	b
*→ [A] q_0	$\{q_1 q_2\}$ B	ϕ D
* $\{q_1 q_2\}$ B	$\{q_1 q_2\}$ B	$\{q_1 q_3\}$ C
ϕ D	ϕ D	ϕ D
* $\{q_1 q_3\}$ C	$\{q_1 q_2\}$ B	ϕ D



Put a comma in NFA table but don't put a comma in DFA.



	0	1	ε
1	2	∅	3
2	∅	(2,4)	∅
3	4	∅	2
4	3	∅	∅

$\varepsilon_P(1) = \{3,2,1\}$ $\varepsilon_P(4) = \{4\}$

$\varepsilon_P(2) = \{2\}$ $\varepsilon_P\{3\} = 3,2\}$

	0	1
1	2,4	2,4
2	∅	2,4
3	4	2,4
4	3	∅