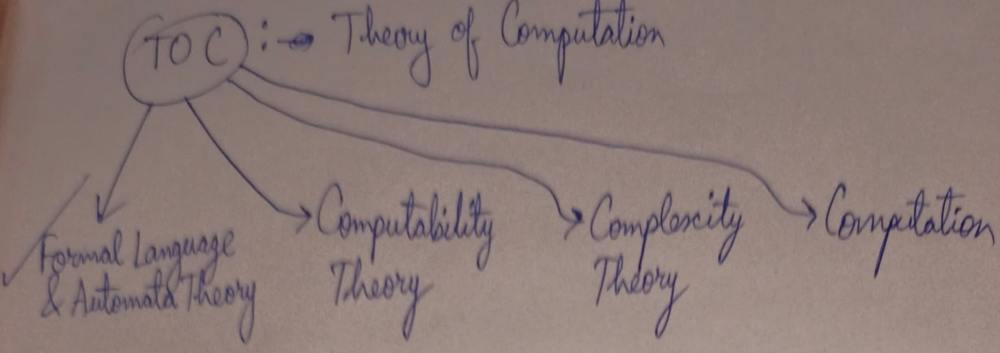
5.3.24

Automata



⁍ Computation is the movement and alteration which occurs during the transition of data or the processing of data based on a set of operations.

⁍ ~~Form~~ Formal Language & Automata Theory :-
It deals with the definition and properties of various mathematical models of computers.
Eg:- ~~Context~~ Finite Automata, Context-free grammar, Turing machine.

⁍ Computability Theory : It deals with what can and what cannot be computed by a particular model. ~~respectively~~

⁍ Complexity Theory : It groups the computable problems based on their efficiency.

Basic Terminologies

1. Symbol/Character : It is the smallest building block/unit which can be any alphabet, letter or a picture. It is infinite.

2. Alphabets (Σ) (sigma): Set of symbols. It is infinite.

Eg:- If $\Sigma = \{a, b\}$, the language consists of only a's and b's.

① 00110001.
$\Rightarrow \Sigma = \{0, 1\}$.

② ABCCDE
$\Rightarrow \Sigma = \{A, B, C, D, E\}$.

Infinite

3. String: ~~A sequence of symbols~~. It is a finite sequence of symbols from some alphabet. It can be denoted by 'w'.

Eg: $\Sigma = \{a, b\}$.
$w = ab, aa, bb, abba, \ldots$

Book: Hopcroft

4. ϵ: ~~The occurrence of~~ null Empty string is the string with zero occurrence of symbols. It is denoted by 'ϵ' (epsilon). Eg:- $a^n b^n$; $n \geq 0$.

$\therefore a^0 b^0 = \epsilon$.

$\Sigma = \{a, b\}$
If no of ~~strings to be~~

V.Imp.

Number of strings of length 2 needs to be generated with $\Sigma = \{a, b\}$ ~~a and b~~: aa, bb, ab, ba (4).

$|\Sigma|$ = No. of symbols in Σ

length $|w|$ = ~~No of~~ n.

Then, No. of strings of length n possible over $\Sigma = |\Sigma|^n$.

P.T.O

5. Language: A collection of strings. It is denoted by 'L'. It has some specific rules.

Eg:- $\Sigma = \{a, b\}$

$L_1 \rightarrow$ Set of all strings of length 2 / 10. (Finite)

$L_2 \rightarrow$ Set of all strings where each string starts with 'a'. (Infinite).

Imp. :→ Language can be finite OR infinite.

Power of Σ

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

$\Sigma^0 = \epsilon \rightarrow$ Set of all strings of length 0.

$\Sigma^1 = a, b \rightarrow$ " " " " 1

$\Sigma^2 = ab, aa, bb, ba. \rightarrow$ " " " " 2

$\Sigma^* =$ set of all possible strings of any length over the alphabet. (i.e. any length).

$\Sigma^+ =$ set of strings of length > 0 [i.e., except ϵ]

Imp. :→ i) $\Sigma^* = \Sigma^0 + \Sigma^1 + \Sigma^2 + \ldots + \Sigma^n$.

:→ $L \subset \Sigma^*$

Closure Properties

1. $L^+ \rightarrow$ indicates the positive closure which represents a set of all strings except the null string or ϵ.

2. $L^* \rightarrow$ Kleene star / Kleene closure. It represents the operands of certain alphabets ~~from~~ for given language where the strings are from 0 to infinite, including the null string (ϵ).

$\therefore \boxed{L^* = L^+ \epsilon}$.

Grammar in TOC

$G = \{V, T, P, S\}$

- G → Grammar
- V → Variable
- T → Terminal
- P → Production rules
- S → Start/Starting Symbol

Variable : Value varies.

Terminal : Value cannot be varied anymore.

Eg :- $S \to aSb/\epsilon$ → Production rules

aSb

$\Rightarrow a\,aSb\,b$

$\Rightarrow aa\,aSbb\,b$

$\Rightarrow aaabbb$ (Putting $S \to \epsilon$) — To complete the string.

Here, $V = S$ (Variable)

$S = S$ (Start Symbol).

$T = a, b$.

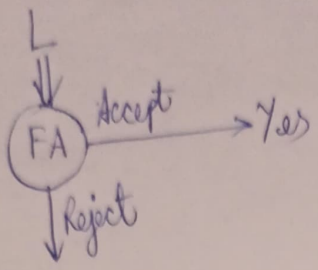
$a^n b^n$ → ~~Languarge~~ Language

⇒ Variables are in capital letters.

⇒ Terminals are in small letters.

7.3.24

Finite Automata

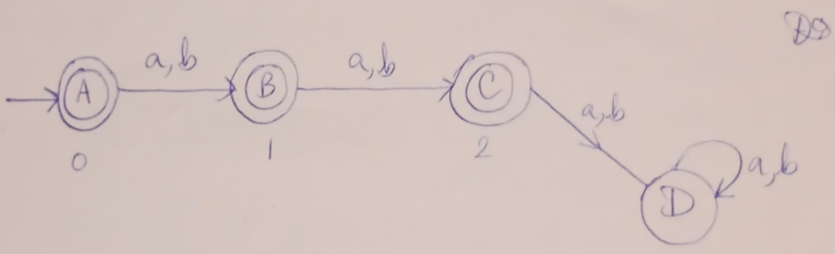


⁂ Finite Automata consists of finite number of states, though the language may be infinite.

○ : Start State ◎ : Final State

Eg:- $a^n b^n$, $n \leq 2$.

$L = \{ \epsilon, a, b, aa, bb, ab, ba \}$.



D: Dead/Trap State

⁂ A string will be accepted if after scanning the entire string, we are in a final state, otherwise it will be rejected.

⁂ For a language to be accepted by a Finite Automata, all the strings in the language must be accepted. Also, the FA must reject every string not belonging to the language.

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

~~Toc~~

- Finite Automata
  - FA with O/P
    - Moore
    - Melay
  - FA without O/P
    - DFA (Deterministic FA)
    - NFA (Non-deterministic FA)
    - ϵ-NFA

DFA : ~~(No. of states)~~

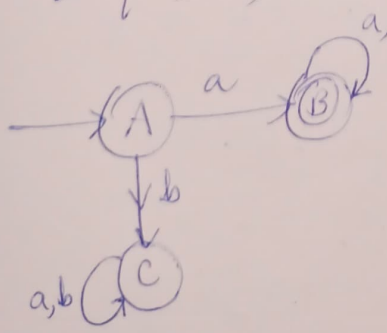
- 5 tuples $(Q, \Sigma, \delta, q_0, F)$
- $Q \rightarrow$ Set of all states
  $\Sigma \rightarrow$ i/p alphabet
  $\delta \rightarrow$ Transition function
  $q_0 \rightarrow$ Starting state
  $F \rightarrow$ Set of final states

Eg:-

L_1 = Set of all strings starting with 'a'.

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

$L = \{a, aa, ab, aba, \dots\}$.



If I/P → abb

$A \xrightarrow{a} B \xrightarrow{b} B$.

P.T.O

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

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

$q_0 = \{A\}$

$F = \{B\}$.

Transition Function (δ):

$$Q \times \Sigma \rightarrow Q$$
$$(A,B,C) \times (a,b) \rightarrow (A,B,C)$$

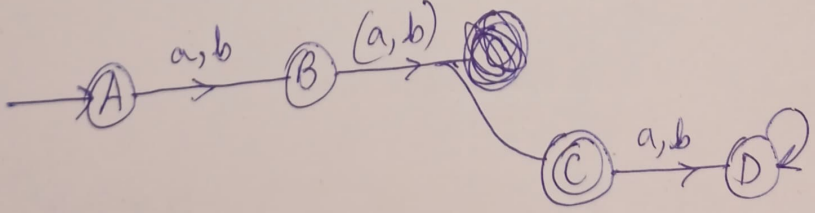
A, a → B
A, b → B C
B, a → B
B, b → B
C, a → C
C, b → C

12.3.24

DFA

1) DFA ⊡ $L = \{\text{Length } 2\}$ $\Sigma = \{a, b\}$.

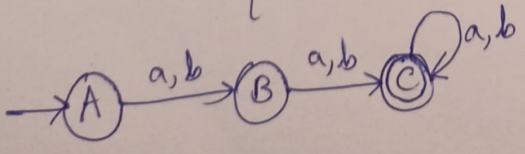
$L = \{ab, ba\}$.



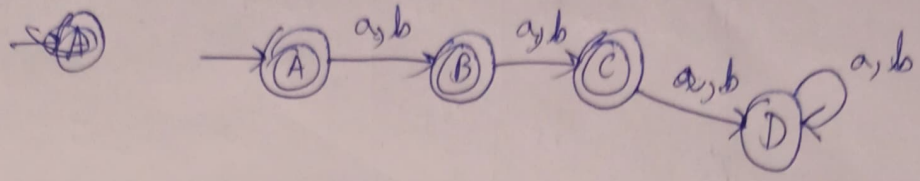
2) $L = \{\text{Length} \geq 2\}$. $\Sigma = \{a, b\}$.

$w \geq 2$.

$L = \{ab, abab, \ldots\}$.



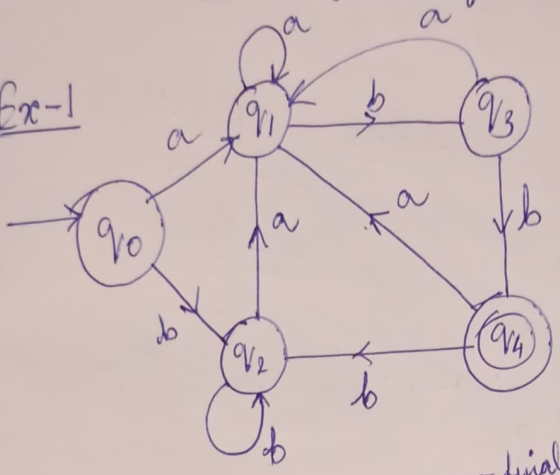
3) $|w| \leq 2$. ~~$L = \{\epsilon, a, b, ab$~~ 0✓ 1✓ 2✓ 3✗



$L = \{\epsilon, a, b, ab, ba, aa, bb\}$.

Minimization of DFA

Ex-1

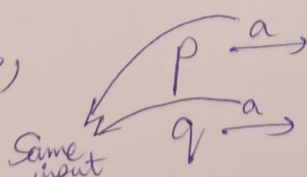


[→ Check whether every non-final state reaches the final state. Here, True.]

[**Equivalence Rule**

(p, q)	
	$\delta(p,q) \in F \Rightarrow \delta(q,w) \in F$
	$\delta(p,q) \notin F \Rightarrow \delta(q,w) \notin F$.

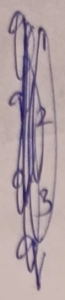
That is,



We can merge these two transitions.]

P.T.O

STATE TRANSITION DIAGRAM



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

Final state

0-equivalence

Non-final states Final States

~~Non-final states~~ $\Rightarrow [q_0\ q_1\ q_2\ q_3] : [q_4]$

1-equivalence

$\Rightarrow$ Check whether any ~~the~~ pair in are equivalent.

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

$[q_0, q_1 \rightarrow$ (a: q_1, q_1; b: q_2, q_3) q_1, q_2, q_3 are ~~eb~~ 0-equivalent

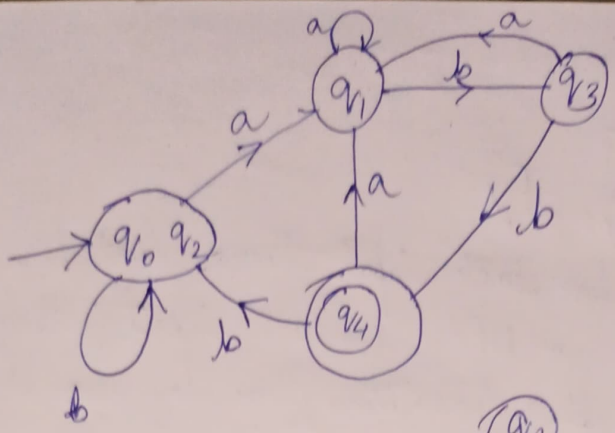
But for $q_3 \rightarrow$ a: q_1; b: $*q_4$ → Not equivalent with Other set $[q_4]$ | $[q_0\ q_1\ q_2\ q_3$

2-equivalence

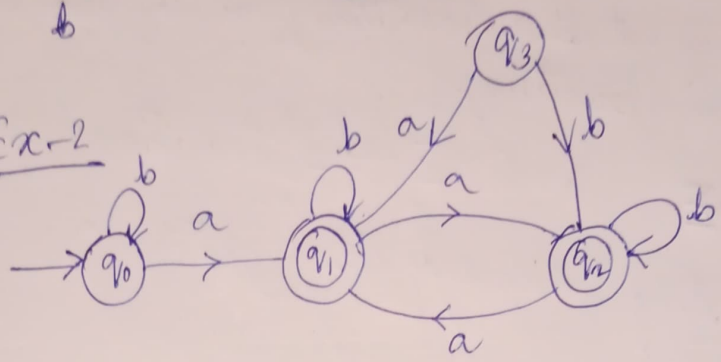
$[q_0\ q_2]\ [q_1]\ [q_3]\ [q_4]$

3-equivalence

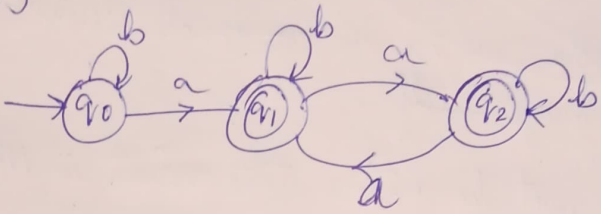
$[q_0\ q_2]\ [q_1]\ [q_3]\ [q_4]$.



Ex-2



Here, q_3 is not reachable. So, remove it.



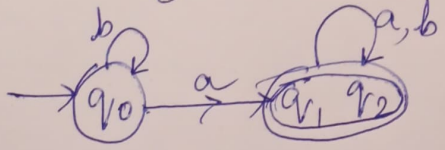
	a	b
$\rightarrow q_0$	$*q_1$	q_0
$*q_1$	$*q_2$	$*q_1$
$*q_2$	$*q_1$	$*q_2$

0-equivalence

$[q_0]\ [q_1\ q_2]$.

1-equivalence

$[q_0]\ [q_1\ q_2]$.



P.T.O

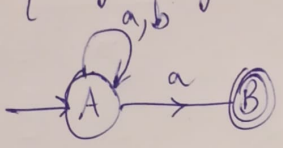
NFA

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

⇒ For DFA, ~~the~~ for each input there must be a transition. But, for NFA, for each input, they may or may not be a transition (there may also be 1 or more than 1 transition).

~~$\delta : Q \times \Sigma$~~ $\quad \delta : Q \times \Sigma \to 2^Q$ transitions

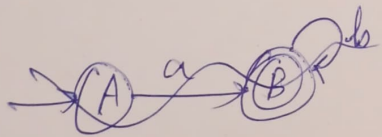
Eg:- $L = \{$every string which ends with a$\}$; $\Sigma = \{a, b\}$.

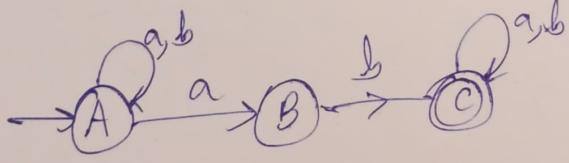


⇒ NFA is very simple, as we can ~~construct~~ define it from the language itself.

⇒ In DFA, there must be a trap state. But in NFA, there is ~~not~~ no trap state.

(1) $L = \{$Contains ab$\}$; $\Sigma = \{a, b\}$.

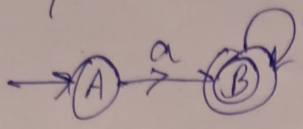




NFA to DFA Conversion

① $L = \{$ Starts with 'a' $\}$; $\Sigma = \{a, b\}$

NFA:



STATE Transition ~~Table~~ Diagram Table

	a	b
→A	*B	ϕ
*B	*B	*B

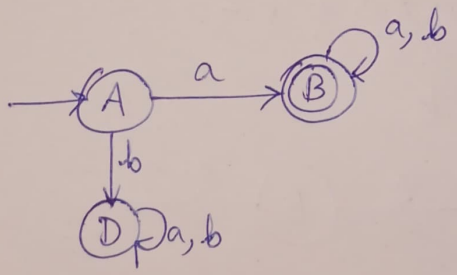
ϕ - Dead Configuration

⇒ Subset Construction Method :-

	a	b
→A	*B	D
*B	*B	*B
D	D	D

Dead state (any name)

DFA:



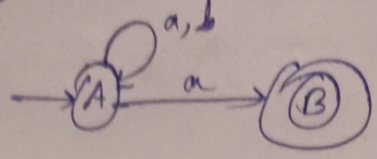
Suggestions :-

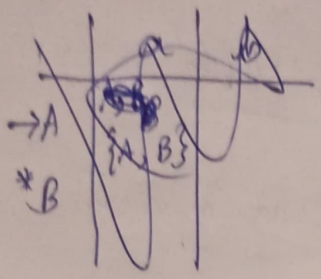
1. What is DFA?
2. DFA VS NFA.
3. ~~Con~~ NFA → DFA.
4. Create DFA and NFA.
5. Minimization of DFA and NFA.

P.T.O

② $L = \{$ String ends with 'a' $\}$; $\Sigma = \{a, b\}$

NFA:



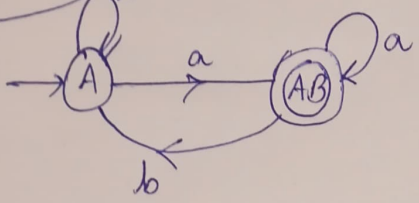


	a	b
→A	{A, B}	{A}
*B	{ϕ}	{ϕ}

Union Union

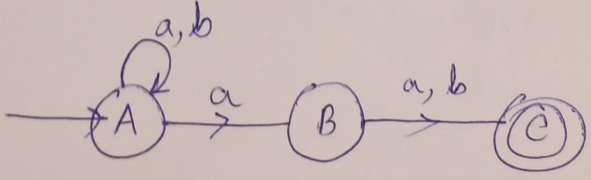
	a	b
→A	[AB]	[A]
*[AB]	[AB]	[A]

DFA:



③ $L = \{$ Second symbol from RHS is 'a' $\}$; $\Sigma = \{a, b\}$. Eg:- abab

NFA:

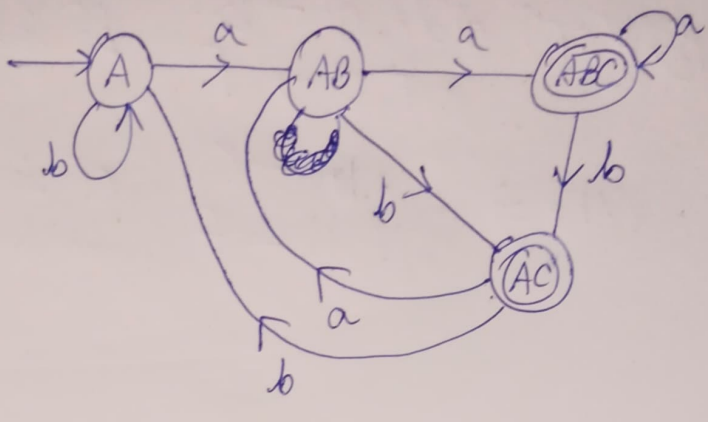


	a	b
→A	{A, B}	{A}
B	{C}	{C}
*C	{ϕ}	{ϕ}

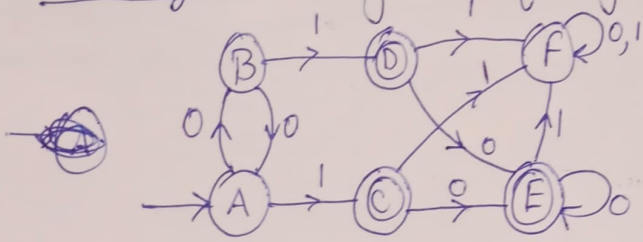
	a	b
→A	[AB]	[A]
[AB]	[ABC]	[AC]
*[ABC]	[ABC]	[AC]
*[AC]	[AB]	[A]

[∵ C is the final state, those states ([ABC] and [AC]) which contain the final state, are final states.]

DFA:



Minimization using Table-filling method / ~~Myhill~~ Myhill-Nerode Theorem



Ans)

	A	B	C	D	E	F
A						
B						
C	✓	✓				
D	✓	✓				
E	✓	✓				
F	✓	✓	✓	✓	✓	

→ (P, Q) map; P ≠ Q.

→ (If P ∈ F and Q ∉ F OR P ∉ F and Q ∈ F), do NOT mark them.

P.T.O

:> Now, check the non-marked transitions

① (A B) ⇒ $\delta(A,0) \to B$ BA is unmarked
$\delta(B,0) \to A$

(AB) ⇒ $\delta(A,1) \to C$ CD is unmarked.
$\delta(B,1) \to D$

② (AF) ⇒ $\delta(A,0) \to B$ BF is unmarked.
$\delta(F,0) \to F$

(AF) ⇒ $\delta(A,1) \to C$ CF is marked, so mark AF.
$\delta(F,1) \to F$