Theory of computing

* Theory of computing মূলত algorithm compiler নিয়ে কাজ করে, কীভাবে কম্পিউটার compute করে কতটুকু দ্রুত কাজ করে এবং কী পরিমাণ জায়গা লাগে তা আমরা বুঝতে পারবো এই TOC ব্যবহার করে।

Binary String end 0 → AC ✓

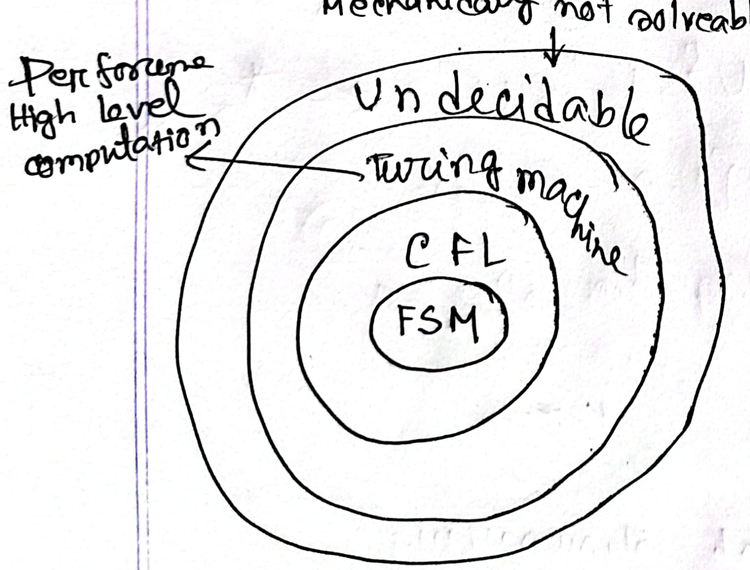
1101 0110 =

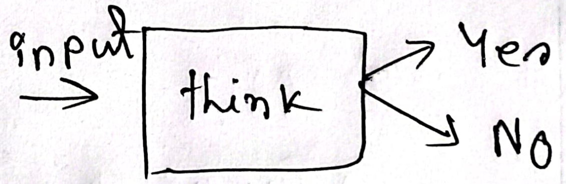
✓ compilere ও একইভাবে কাজ করে

if 0 = ✓

else ✗

Compilers





FSM – finite State Machine
- simplest model
- limited memory.

CFL → context Free Language

Not P. language
Set of string.

① FSM → Finite State Machine.

↓

বুঝতে হলে, কিছু basic word বুঝতে হবে like,

Symbol → a, b, c, 0, 1, 2, 3. . . . . .

Alphabet → denoted by Σ - collection of symbols → {a, b}, {d, e, f, g}
{0, 1, 2, 3}...

String - sequence of symbols → a, b, 0, 1, aa, bb, ab, 01. . .

Language - Set of String

Ex- $\Sigma = \{0, 1\}$ → Alphabets

⇒ language L1 = Set of all string of length 2
= {00, 01, 10, 11}

⇒ L_2 = Set of all string of length 3.
= {000, 001, 010, 011, 100, 101, 110, 111}

⇒ L_3 = Set of all string that begin with 0.
= {0, 00, 01, 000, 001, 010, 011, 0000...}

L_1 and L_2 are finite set because finite number of element

L_3 is infinite set

* power of Σ : $\Sigma = \{0, 1\}$

Σ^0 = set of all string of length 0 : $\Sigma^0 = \{\epsilon\}$

Σ^1 = Set " " " " 1 : $\Sigma^1 = \{0, 1\}$ cardi - 2

Σ^2 = " " " " " 2 : $\Sigma^2 = \{00, 01, 10, 11\}$

cardinality - 4

" " " " n .

Σ^n = "

cardinality : number of elements in a set.

for Σ^n = 2^n
Cardinality,

$\Sigma^* \rightarrow \Sigma^0 \cup \Sigma^1 \cup \Sigma^2 \cup \Sigma^3$

$\rightarrow \{\epsilon\} \cup \{0, 1\} \cup \{00, 01, 10, 11\} \cdots$

= set of all possible string of all lengths over $\{0, 1\}$

$\longrightarrow$ infinite

prerequisite done Now

$\downarrow$

simplest model of computati'o

limited Memory

Finite State Machine

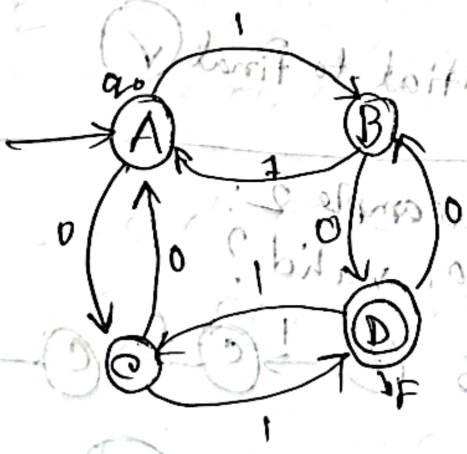
Finite Automata

- FA with output
  - Moore Machine
  - Mealy Machine
- FA without output
  - DFA — Diterministic
  - NFA — Non DFA
  - ε-NFA

প্রথমে দেখবো,

DFA → Deterministic Finite Automata

DFA Diagrame



DFA Defined by using 5 things $(Q, \Sigma, q_0, F, \delta)$

Q = Set of all states

Σ = inputs

q_0 = Start state / initial state

F = Set of Final State

δ = transition Function from $Q \times \Sigma \rightarrow Q$

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

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

$q_0 = A$

$F = \{D\}$

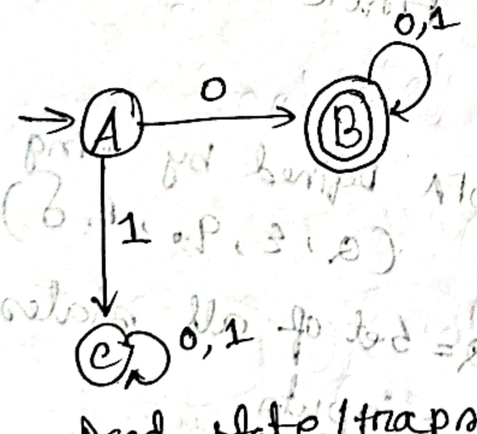
$\delta = Q \times \Sigma$

	0	1
A	C	B
B	D	A
C	A	D
D	B	C

Now all DFA basic done; Example – 1:

L_1 = Set of all string that start with '0'

= { 0, 00, 01, 000, 010, 011, 0000, ... }



Dead state / trap state

কারন এখান থেকে আর: AC আউটপুট পাবো না

check ok?

Ex – 001 valid?

Initial State $A \xrightarrow{0} B \xrightarrow{0} B \xrightarrow{1} B$ → final State

initial to final ✓

Example 2:

→ 101 valid?

I.S $A \xrightarrow{1} C \xrightarrow{0} C \xrightarrow{1} C$ → Not final State

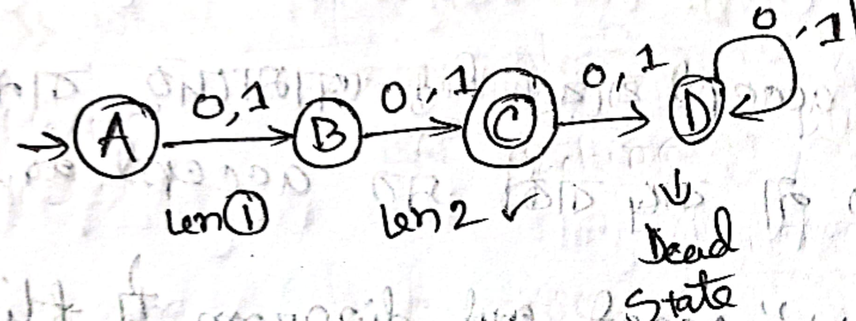
So Ans ✗

Example - 2

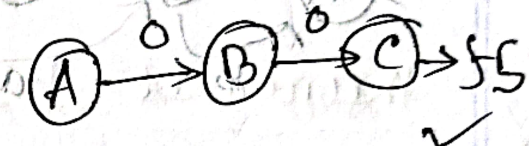
construct a DFA that accepts ~~set of~~ all string over $\{0, 1\}$ of length 2

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

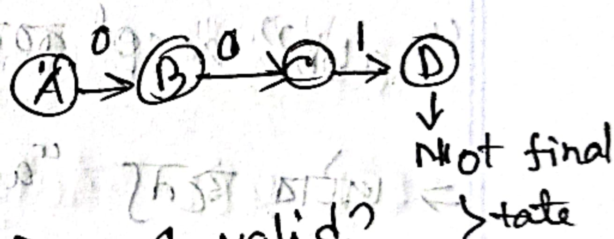
$L = \{00, 01, 10, 11\}$



Ex → 00 valid?



Ex → 001 valid?



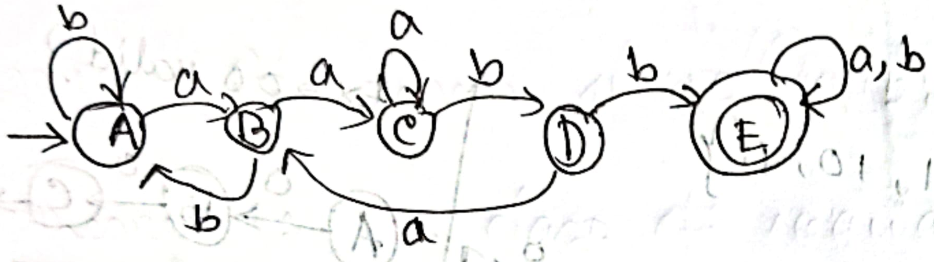
Ex - 1 valid?

So Not valid

Example 3: construct DFA that accepts any string over {a, b} that does not contain the string "aabb" in it.

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

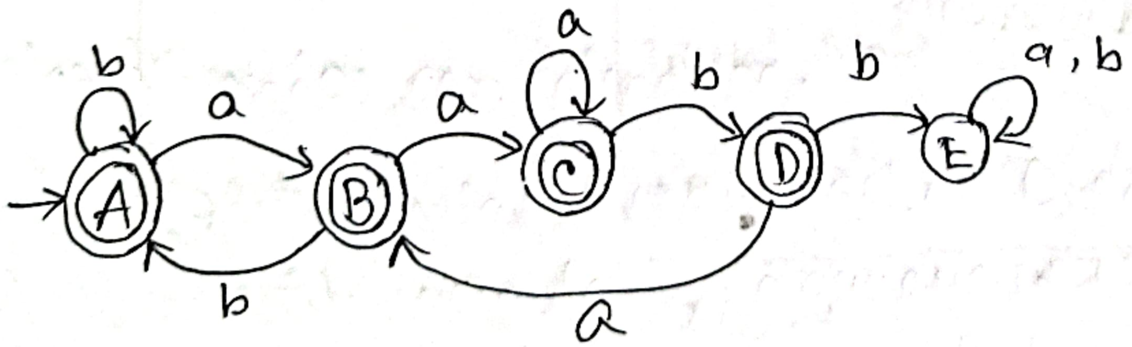
এই প্রব্লেমটার জন্য আমরা প্রথমে "aabb" accept করে এমন DFA Design করবো। তারপর সেটা উল্টাই দিবো।



এই DFA "aabb" কে accept করে, কিন্তু আমাদের বানাতে হবে "aabb" accept করবে না বাকি সব accept করে এমন।

⇒ এটার জন্য "aabb" accept diagram টা flip করে দিবো

Final state টাকে Non-dead state বানাই দিবো বাকি গুলোকে final state বানাই দিবো।



Regular Language

যেকোনো
* একটা Language কে regular তখনই বলবে যখন FSM (Finite State Machine) এটাকে recognize করতে পারে।

⇒ তাহলে Not regular Language কখন হবে?
The languages,
» which are not recognized by FSM
» which require memory
— আমরা আগেই বলেছি FSM এর Memory খুবই limited
— এটি কোনো String store এবং count করতে পারে না

Operation on regular Language

UNION → $A \cup B = \{x \mid x \in A \text{ or } x \in B\}$ মানে A,B এর সব নিবে।

Concatenation – $A \circ B = \{xy \mid x \in A \text{ and } y \in B\}$ A থেকে x নিবে B " y " then জোড়া করবে $\boxed{\circ \rightarrow \text{চিহ্ন}}$।

STAR – $A^* = \{x_1, x_2, x_3 \dots x_k \mid k \geq 0 \text{ and each } x_i \in A\}$
এটার মানে A থেকে ইচ্ছামতো Symbol নিয়ে State বানাতে পারবে

Ex → $A = \{pq, r\}$, $B = \{t, uv\}$

$A \cup B = \{pq, r, t, uv\}$

ϵ = ফাকা string

$A \circ B = \{pqt, pquv, rt, ruv\}$

$A^* = \{\epsilon, pq, r, pqr, rpq, pqpq, rr, pqpqpq, rrr \cdots\cdots\}$

এখানে ২টি Theorem বা সূত্র আছে:

<u>Theorem 1:</u> The class of regular Language is closed under UNION.

যদি A এবং B দুটি ~~রেগুলার~~ regular language হয় তবে তাদের U → (A∪B) ও regular language হবে।

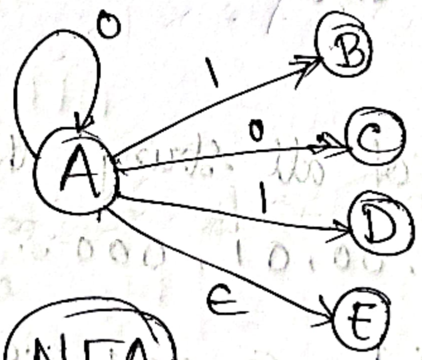
<u>Theorem 2:</u> The class of regular Language is closed under concatenation.

Same → ২টি regular language A B এর concatination AoB ও একটি regular language হবে।

NFA — Non-Deterministic Finite Automata

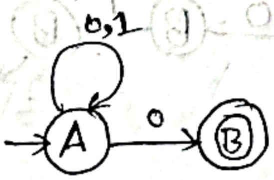
* DFA আমরা জানি আমাদের current state দিলে আমরা next state জানি 0 হলে কই যাবে আর 1 হলে কই যাবে, state গুলো নির্দিষ্ট হয়। state forward হয় random choice করে যায় না।

* NFA তার বিপরীত। যেমন-
  - → multiple next state
  - → next state may be chosen random
  - → All " " " " parallel



ε-ফাঁকা string

NFA



define as $(\underbrace{Q, \Sigma, q_0, F}_{\text{same as DFA}}, \underbrace{\delta}_{\text{different}})$

$Q = \{A, B\}$

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

$q_0 = A$

$F = B$

$\delta = Q \times \Sigma \rightarrow 2^Q / 2^n$

$A \times 0 - A$

$A \times 0 - B$

$A \times 1 = A$

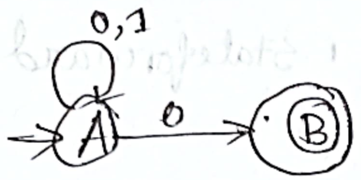
$B \times 0 = \phi$

$B \times 1 \rightarrow \phi$

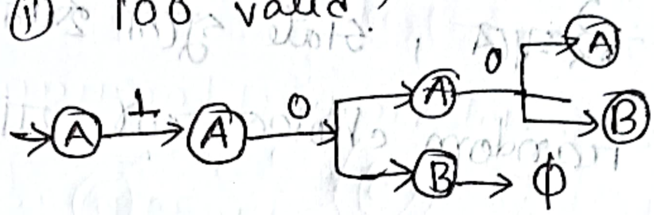
$A \xrightarrow{1} A, AB, B, \phi - 2^2 \rightarrow 4$

Example - 1 (NFA)

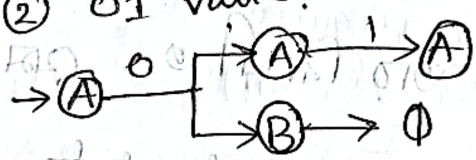
L = Set of all string that end with 0.



① 100 valid?



② 01 valid?



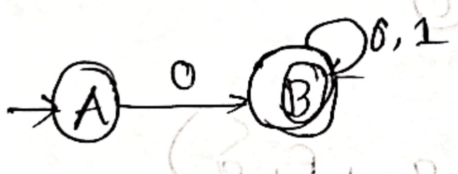
$\times$

» এখান থেকে বুঝলাম যদি NFA Machine টা রান হয় এবং শেষে end state এর যেকোনো একটি যদি final state হয় তাহলেও string accept করবে। otherwise করবে না।

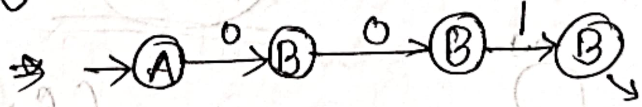
EX - 2

L = Set of all string that start with 0.

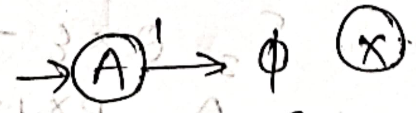
= {0, 00, 01, 000 .... }



① 001 accept?



② 101?



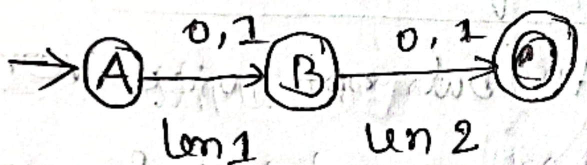
NFA তে আমি কিন্তু input 1 হলে কী হবে বলি নাই Basically DFA

তে তাও বলা লাগলেও NFA তে তা লাগেনা। Final state পর্যন্ত পৌঁছালেই হয়। এরপর আর চিন্তা করতে হবে না।

→ construct a NFA that accepts set of all strings over $\{0,1\}$ of length 2.

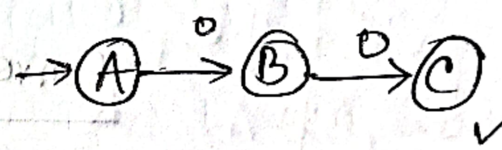
$\Sigma = \{0,1\}$

$L = \{00, 01, 10, 11\}$

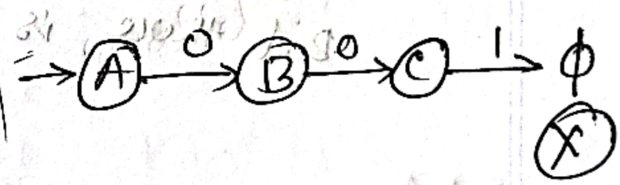


≫ len 3 হলে ϕ থাকবে আর দেখাতে হবে না NFA তে

① 00 ?

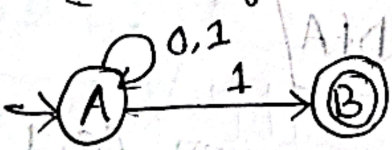


② 001 valid? ⊗

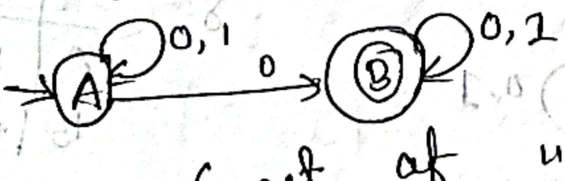


Example-3

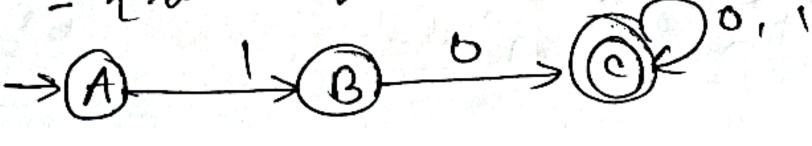
3.1 ⇒ L_1 = {set of all strings that ends with '1'}



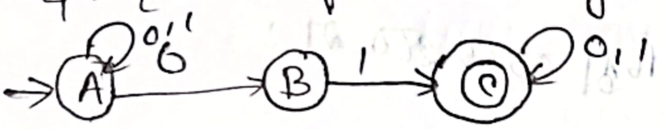
3.2 ⇒ L_2 = {set of all " " contain '0'}



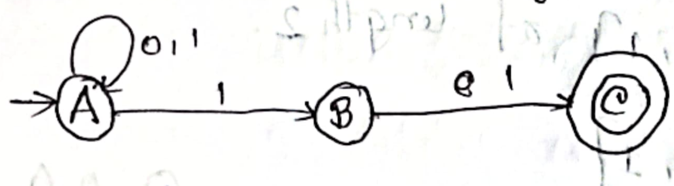
3.3 ⇒ L_3 = {set of " " that starts with '10'}



3.4 ⇒ L_4 = {set of all strings that contain '01'}



3.5 ⇒ L_5 = {set of all string that end with '11'}



Conversion of NFA to DFA

- Every DFA is an NFA but not vice versa.

  But there is an equivalent DFA for every NFA

DFA

$$\delta = Q \times \Sigma \rightarrow Q$$

NFA

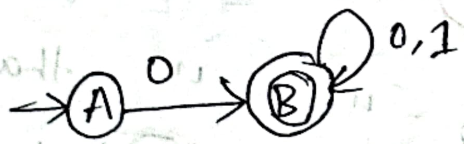
$$\delta = Q \times \Sigma \rightarrow 2^Q$$

$$\boxed{NFA \cong DFA}$$

L = {set of all string over (0,1) that starts with '0'}

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

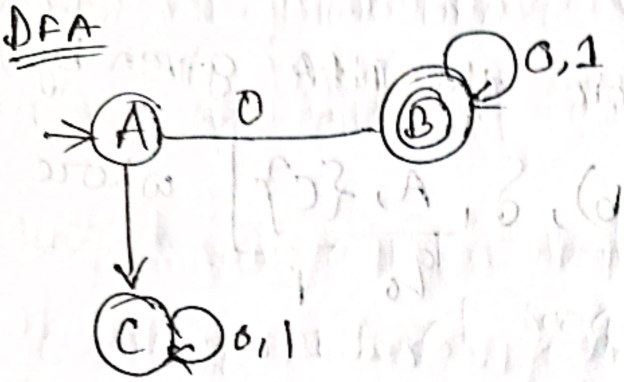
NFA



δ:

	0	1
A	B	ϕ
B	B	B

DFA



	0	1
A	B	C
B	B	B
C	C	C

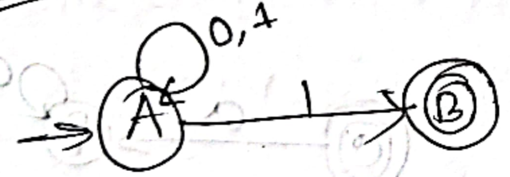
C - dead state

Example -1

L = {set of all string over (0,1) that end with '1'}

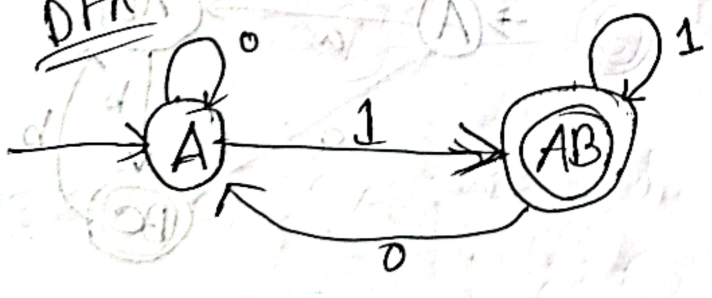
$\Sigma = 0,1$

NFA



	0	1
A	{A}	{A, B}
B	ϕ	ϕ

DFA



	0	1
A	{A}	{AB}
AB	{A}	{AB}

AB single State

* এই NFA to DFA তে process কে subset construction method এ বলে

$\boxed{AB = A \cup B}$

Example-2

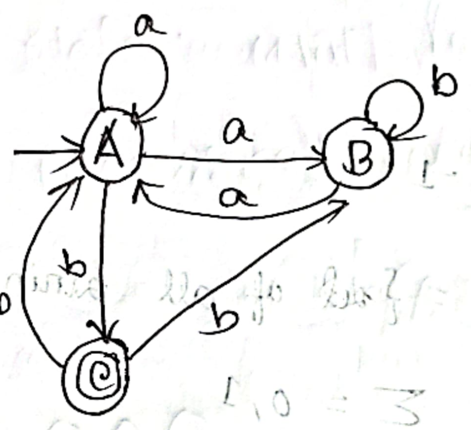
Find the equivalent DFA for the NFA given by

$M = [\underbrace{\{A, B, C\}}_{Q}, \underbrace{(a, b)}_{\Sigma}, \delta, \underbrace{A}_{q_0}, \underbrace{\{C\}}_{F}]$ where δ is

given by;

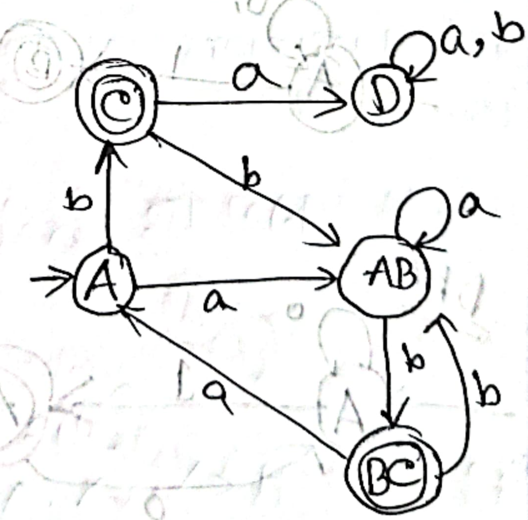
NFA

	a	b
→A	A,B	C
B	A	B
Ⓒ	–	A,B



DFA

	a	b
→A	AB	C
AB	AB	BC
(BC)	A	AB
Ⓒ	D	AB
D	D	D

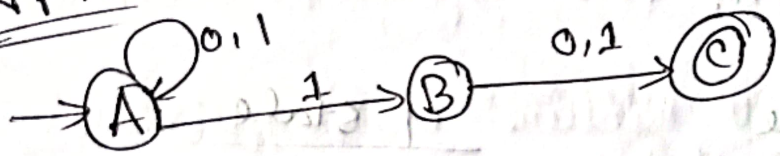


D - Dead State

Example-3

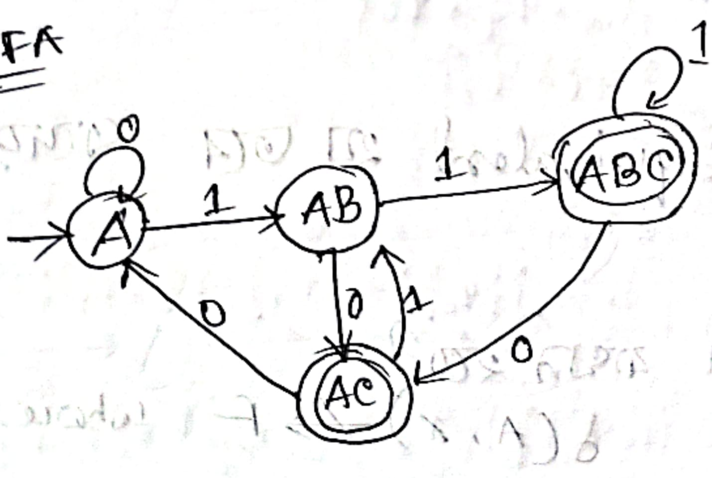
Design an NFA for a language that accepts all strings over {0, 1} in which the second last symbol is always '1'. Then convert it to its equivalent DFA.

NFA



	0	1
→A	A	A, B
B	C	C
(C)	ϕ	ϕ

DFA



	0	1
→A	A	AB
AB	AC	ABC
(AC)	A	AB
(ABC)	AC	ABC

Minimization of DFA

- minimal version of an DFA
- minimum number of state possible

সহজ কথায় বললে ধর তোমার কোনো friend আর তুমি ২টো DFA design করলা। তোমার friend এর লাগলো 4 টা state আর তোমার 5 state। কোনটা নিবা 4 টা না?

✓ এই state কমিয়ে আনার process টাই minimization of DFA

5 state   4 state

$\underbrace{O \quad O}_{\text{Equivalent}}, O \quad O \quad O$

ধরে নাও O এক একটা state. এগুলো minimize করার কিছু শর্ত আছে।

২টো state অর্থাৎ যদি equivalent হয় তবে এদের একটা করে দিতে পারবো।

এখন প্রশ্ন equivalent কখন হবে?

$\delta(A, x) \to F$ and $\delta(B, x) \to F$

OR

$\delta(A, x) \not\to F$ $\delta(B, x) \not\to F$ where x = input string

যদি A তে x input দিলে F এ যায়
এবং B " " " " " "

অথবা না যায় both এখন হবে equivalent

equivalent এর কিছু ~~টেক~~ type আছে।

length of x string

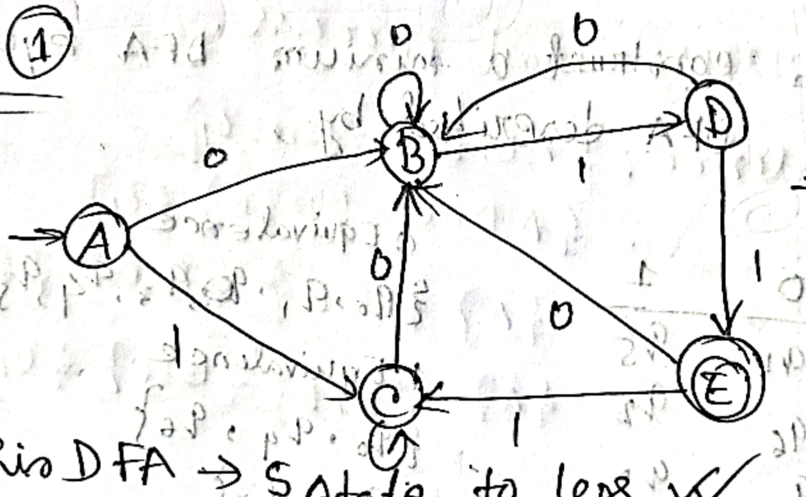
if, $|x| = 0$ - - - - 0 equivalent

$|x| = 1$ - ——— 1 "

$|x| = 2$ - - — 2 "

$|x| = n$ - - - — n "

Example 1



transation table

	0	1
→A	B	C
B	B	D
C	B	C
D	B	E
E	B	C

make this DFA → 5 state to less ✓

0 Equivalent = {A, B, C, D} {E} → final state আলাদা করি গুলো ২ ভাগে

1 " = {A, B, C} {D} {E}

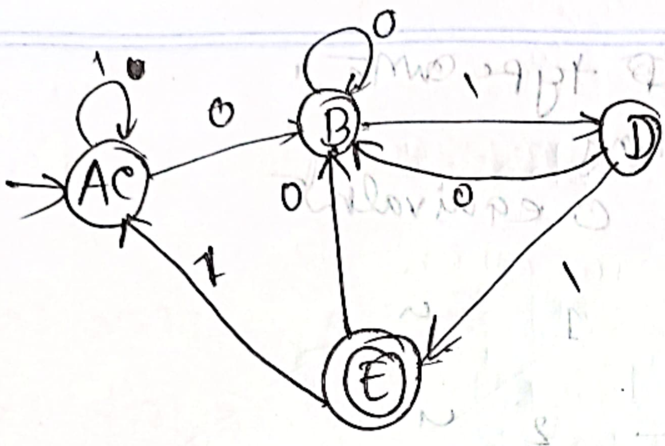
2 " = {A, C} {B} {D} {E}

3 " = {A, C} {B} {D} {E}

AB ✓
AC ✓
CD ✗

* যে ২টাকে eq- চিন্তা করতেছি তাদের state গুলো যদি same থাকে বা ঐ state গুলো যদি same set এ থাকে তবে তাকে eq- হিসেবে নিতে পারবো

২টা consiquetive same আসলে আমরা break করবো এবং AC কে 1টা state বানাইতে পারবো



Given δ:

	0	1
→A	B	C
B	B	D
C	B	C
D	B	E
(E)	B	C

So this DFA is 4 state, previous one and this one give same output..

Example-2 construct a minimum DFA equivalent to the DFA described by,

	0	1
→ q_0	q_1	q_5
q_1	q_6	q_2
(q_2)	q_0	q_2
q_3	q_2	q_6
q_4	q_7	q_5
q_5	q_2	q_6
q_6	q_6	q_4
q_7	q_6	q_2

0 Equivalence

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

1 Equivalence

$\{q_0, q_4, q_6\}$

$\{q_1, q_7\}$

$\{q_3, q_5\}$ $\{q_2\}$

2 Equivalence

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

3 Equivalence

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

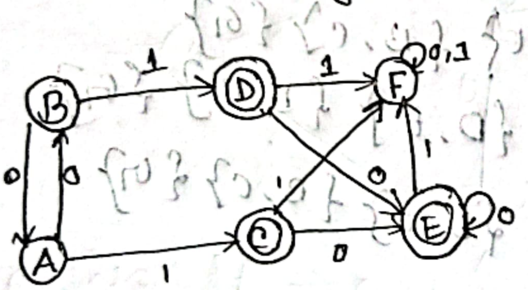
	0	1
$\rightarrow \{q_0, q_4\}$	$\{q_1, q_7\}$	$\{q_3, q_5\}$
$\{q_6\}$	$\{q_6\}$	$\{q_0, q_4\}$
$\{q_1, q_7\}$	$\{q_6\}$	$\{q_2\}$
$\{q_3, q_5\}$	$\{q_2\}$	$\{q_6\}$
$\{q_2\}$	$\{q_0, q_4\}$	$\{q_2\}$

$\Rightarrow$ 8 States to 5 state

Example-3 DFA-Minimization...

When there are more than one Final states involved

Minimize the following DFA:

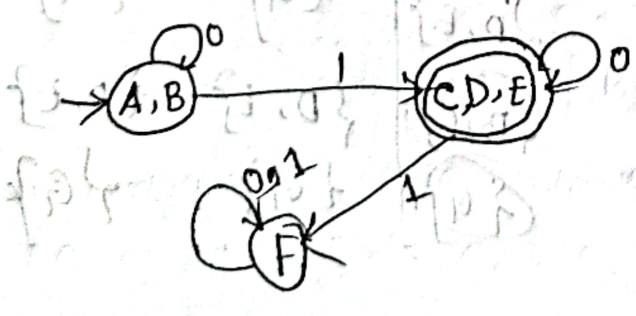


	0	1
$\rightarrow$ A	B	C
B	A	D
C	E	F
D	E	F
E	E	F
F	F	F

0 equivalence : {A, B, F} {C, D, E}

1 " : {A, B} {F} {C, D, E}

2 " : {A, B} {F} {C, D, E}

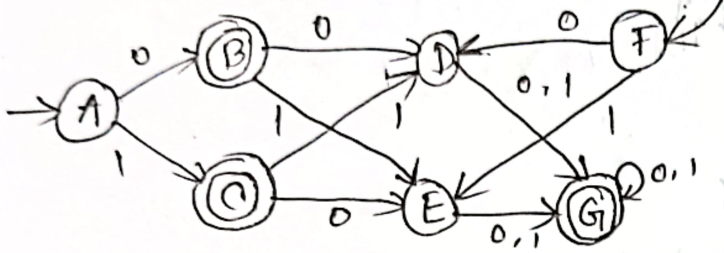


	0	1
$\rightarrow$ {A, B}	{A, B}	{C, D, E}
{F}	{F}	{F}
{C, D, E}	{C, D, E}	{F}

initial state থেকে যে state এ কোনোভাবেই যাওয়া যায় না।

Example - 4

When there are Unreachable states involved.



	0	1
→A	B	C
(B)	D	E
(C)	E	D
D	G	G
E	G	G
(G)	G	G

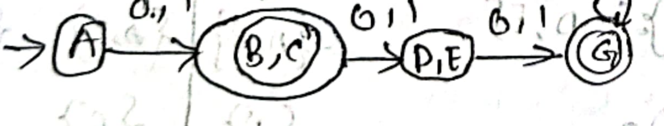
এই অবস্থায় আমরা U.S → F কে বাদ দিয়ে transition table δ করতে, তারপর normal process.

0 Equivalence : {A, D, E} {B, C, G}

1 : {A, D, E} {B, C} {G}

2 : {A} {D, E} {B, C} {G}

3 : {A} {D, E} {B, C} {G}



	0	1
→{A}	{B, C}	{B, C}
{D, E}	{G}	{G}
({B, C})	{D, E}	{D, E}
({G})	{G}	{G}

Minimization of DFA - Table Filling Method
(Myhill - Nerode Theorem)

Steps:

1) Draw a table for all pairs of states (P, Q)
2) Mark all pair where $P \in F$ and $Q \notin F$. এর মানে pair এর মধ্যে ১টা final এবং অন্যটা non final হতে হবে।
3) If there are any unmarked pairs (P, Q) such that $[\delta(P, x), \delta(Q, x)]$ is marked, then mark [P, Q]

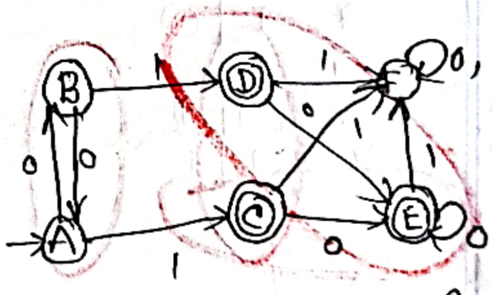
$\boxed{x = \text{input } 0/1}$

Repeat this until no more markings can be made.

→ যদি x=0 বা 1 হয়ে (P,x) এবং (Q,x) এর অন্য ২টা state এ যাবে ঐ state টা Marked হলে এটাও Mark করতে হবে।

4) combined all the unmarked pair and make them a single state in the minimized DFA.

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



$(D, C) - \delta(D, 0) = E \quad \delta(D, 1) = F$
$\quad - \delta(C, 0) = E \quad \delta(C, 1) = F$ Not mark

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

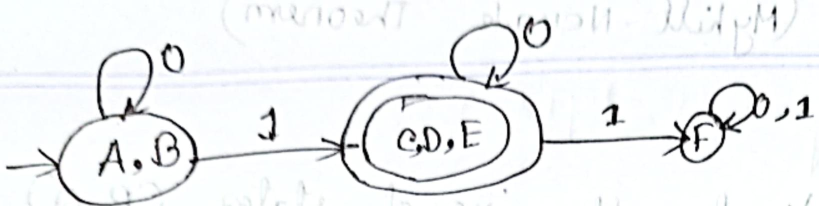
$(E, D) - \delta(E, 0) = E \quad \delta(E, 1) = F$
$\quad \delta(D, 0) = E \quad \delta(D, 1) = F$

$(B, A) - \delta(B, 0) = A \quad \delta(B, 1) = D$
$\quad \delta(A, 0) = B \quad \delta(A, 1) = C$

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

$(F, B) = \delta(F, 0) = F$
$\quad \delta(B, 0) = A$

$\boxed{(A, B) \ (D, C) \ (F, C) \ (E, D)}$



✓ Finite Automata with outputs