NFA

⊛ Non-deterministic Finite Automata

→ যেকোনো if input এর জন্য multiple transition অর্থাৎ একাধিক State এ যেতে পারে।

→ ∈ - epsilon এর জন্য transition হতে পারে। যা DFA তে possible নয়।

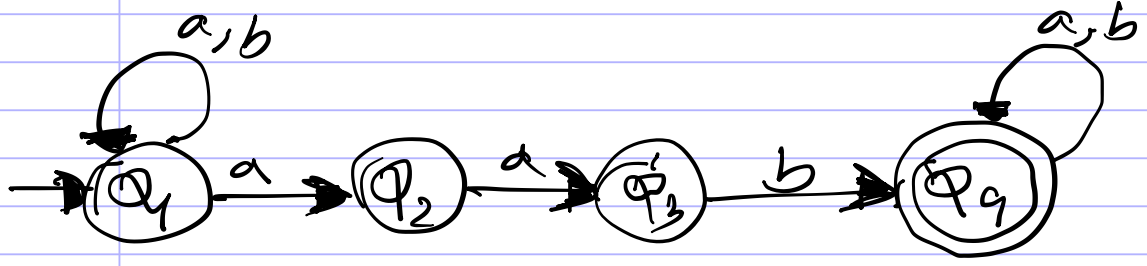
→ যদি কোনো input এর জন্য transition (next state এ না যায়) না থাকে তাহলে সেটি Accepted.

⊛ সকল DFA ∈ NFA কিন্তু সকল NFA DFA নয়।

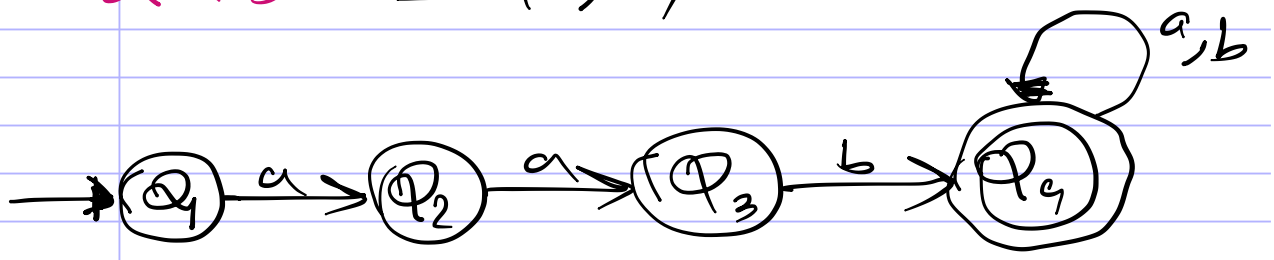
[NFA তে only necessary input এবং এর জন্য transition গুলো দেখানোই হবে।]

4- General type

(*) Design NFA that contains "aab".
$\Sigma = \{a, b\}$.

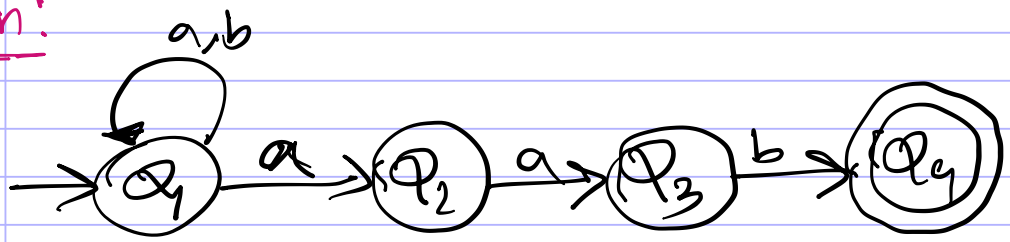


(*) Design NFA that starts with "aab". $\Sigma = \{a, b\}$

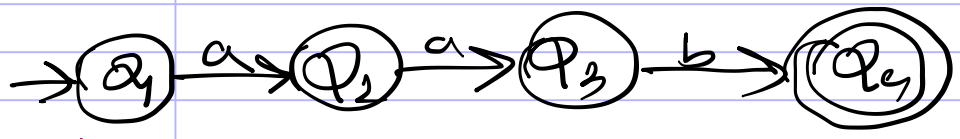


(*) Design NFA that Ends with "aab"
$\Sigma = \{a, b\}$.

Soln:

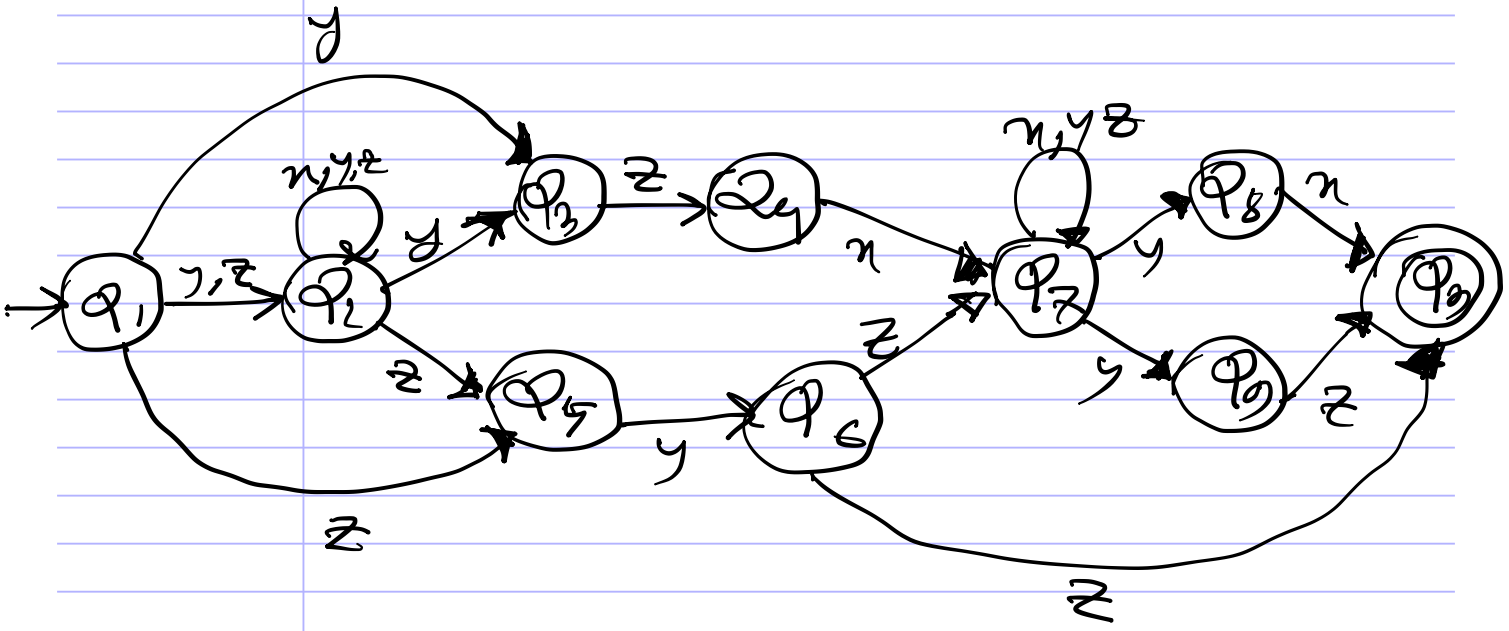


✳ Exactly "aab" . $\Sigma = \{a, b\}$



Fall-24 (2-a)

✳ Doesnot start with "x", contains "yzx" or "zyz" and ends with "yz" or "yx".



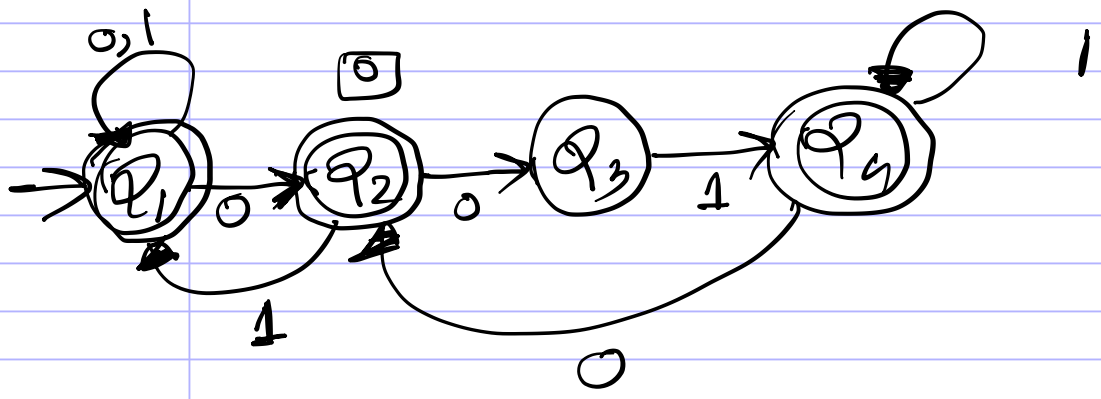
Start: y/z

contains: yzx / zyz

end: yx / yz

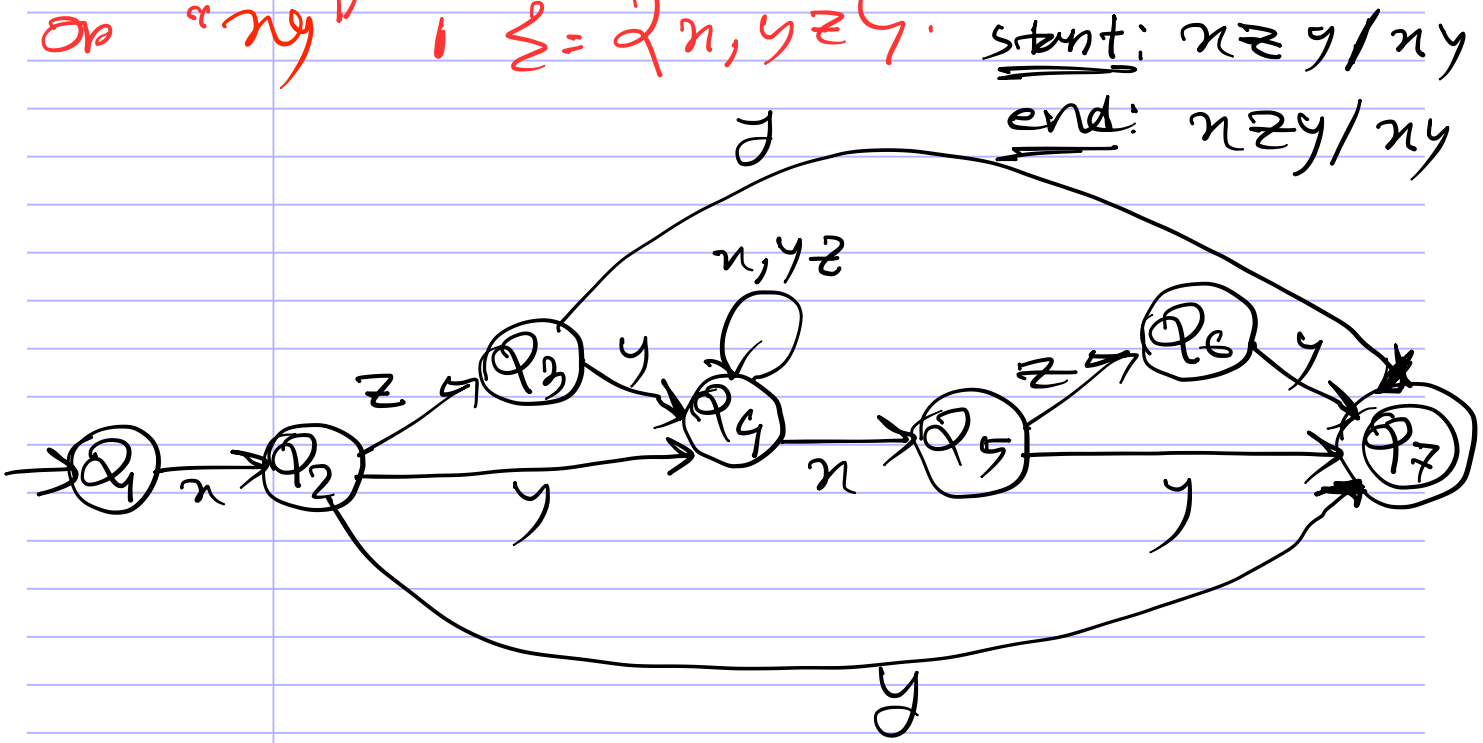
Fall-24 = (2-C)

(*) Every "00" in w is followed by atleast one 1 | Σ = {0,1}



Summer-23

2|c (*) Starts and ends with "xzy" or "xy" | Σ = {x, y, z}.



A

⊛ Fall 23 (2-b)

Start and ends with differend symbols with a total length of atleast 2 | $\Sigma = \{0,1\}$.

Ans:

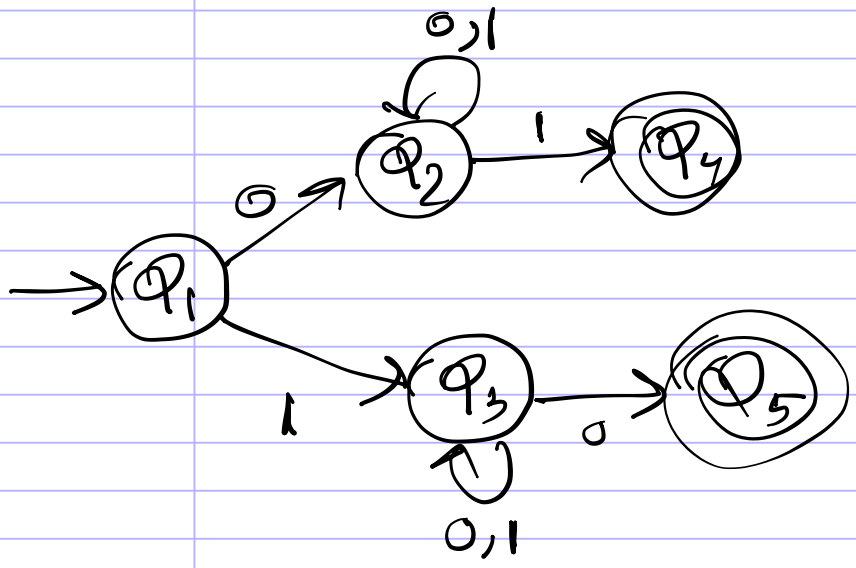
AV

0 1

1 0

NFA

1, 0, 11, 00



Spring-24 (2-a)

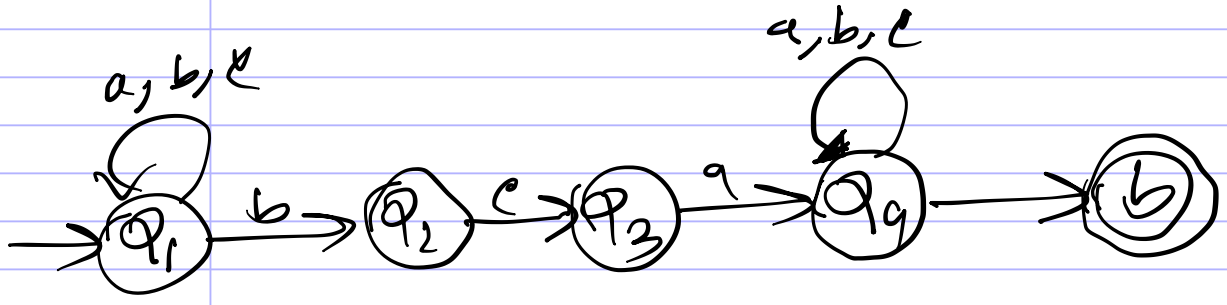
(*) Ends with "b" and contain "bca"

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

Soln:

Contain : bca

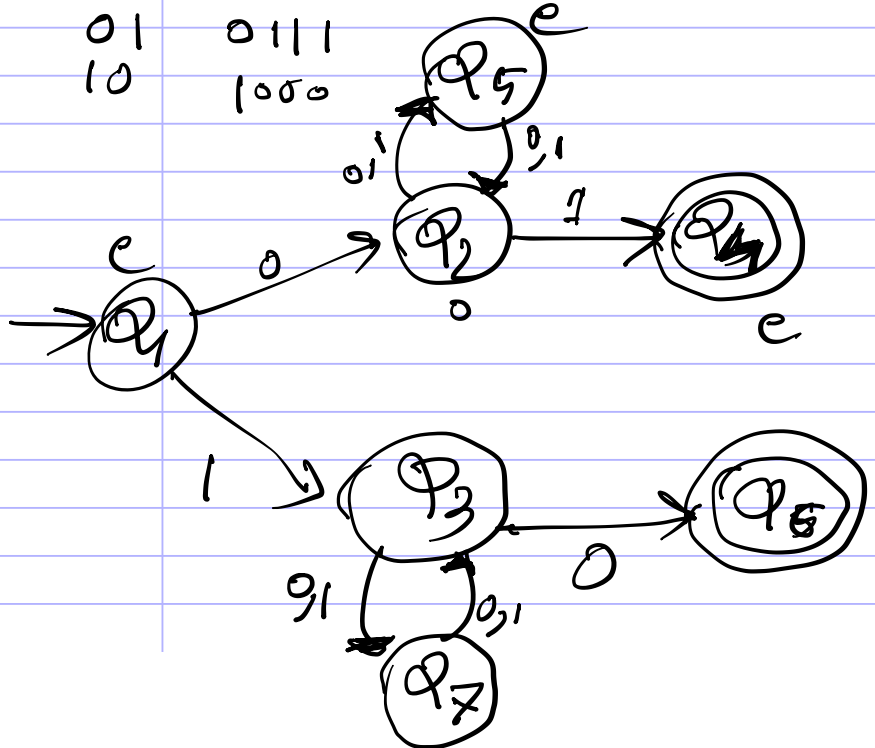
ends : b



Spring 24 (2-b)

(*) Starts and ends with different symbols when the total length is is a multiple of 2 , $\Sigma = \{0, 1\}$.

AV: 2, 4, 6, 8, ...



NFA - Tree

NFA - Tree বানানোর Rules :-

(I) প্রথমে একটি NFA Diagram এবং একটি string দেয়া থাকবে।

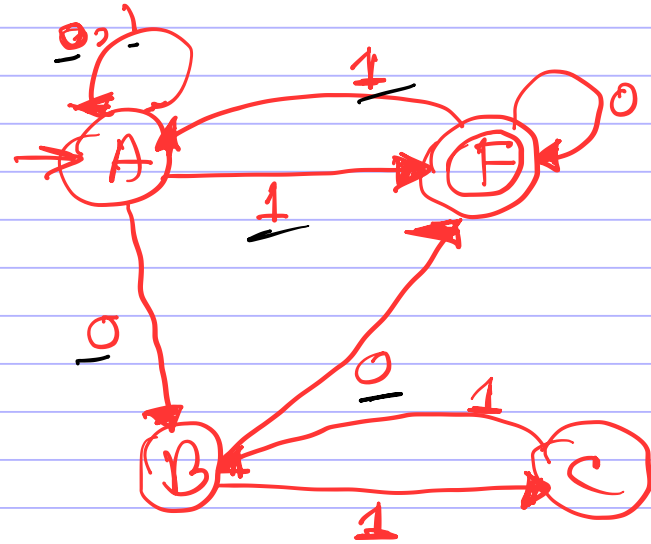
(II) Diagram থেকে Transition table বানাব।

(III) Question এ যেই string দিবে left to right one by one alphabet এর জন্য transition states গুলো দিয়ে tree বানাবো।

(IV) যদি last layer minimum একটিও Accepted state থাকে তবে string টি Accepted

→ নিচে example এর মাধ্যমে explain করা হলো :-

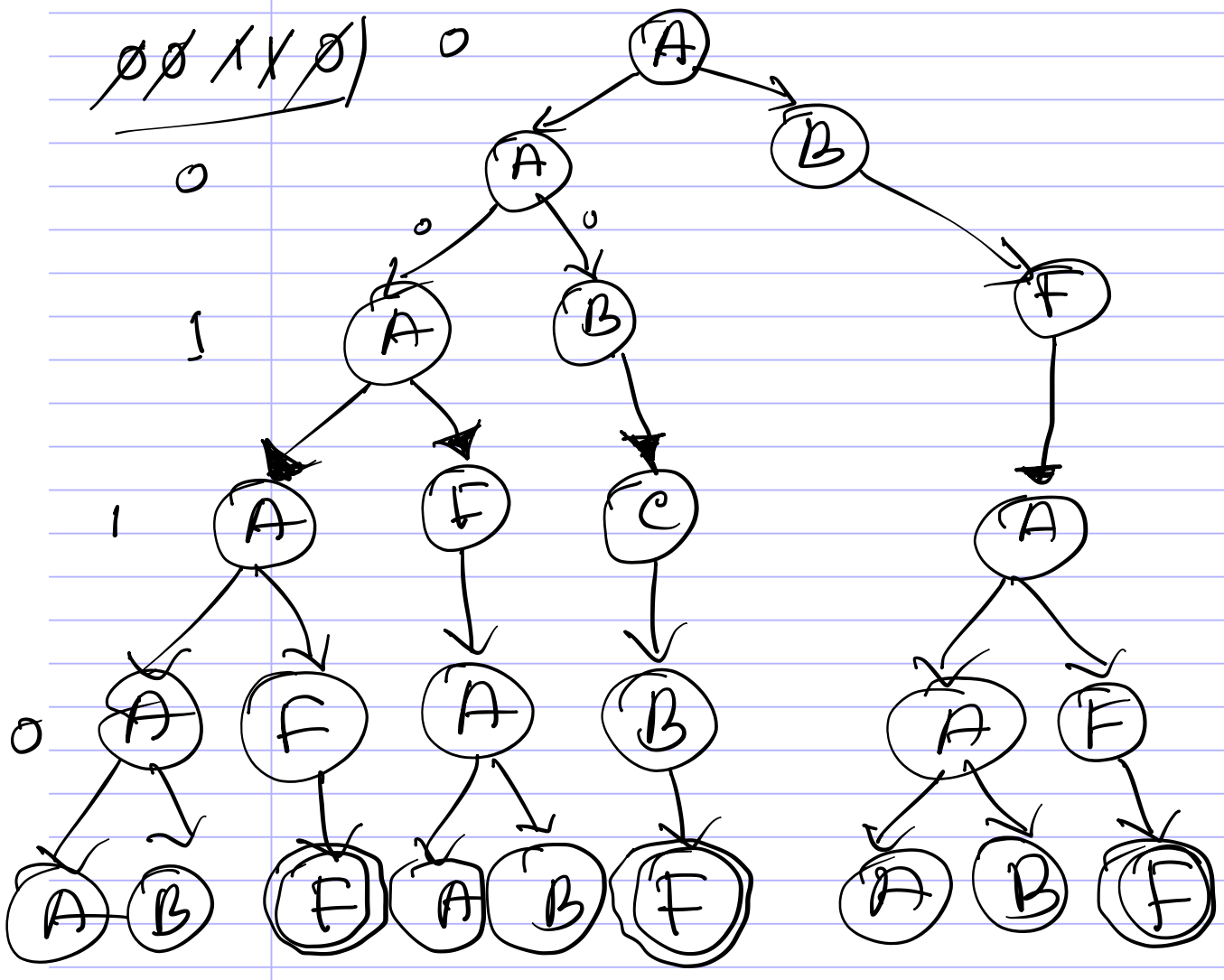
Ex:



Draw NFA Tree For String "00110".

Solution:

States \ Input	0	1
→ A	A, B	A, F
B	F	C
C	ϕ	B
* F	F	A



∴ So the string is Accepted

Ans

Conversion

2 ধরনের conversion হয়ে থাকে :-

(i) NFA to DFA conversion

(ii) ∈-Epsilon NFA to DFA

NFA to DFA Conversion

Steps:

(1) Question এ একটা NFA Diagram থাকবে। সেটাকে DFA তে convert করতে হবে।

(2) NFA diagram এর Transition table বানাব।

(3) NFA transition table থেকে DFA transition table বানাব। বানানোর Rules:-

→ প্রথমে NFA table এর start এর transition গুলোকে DFA table এর মতন করি।

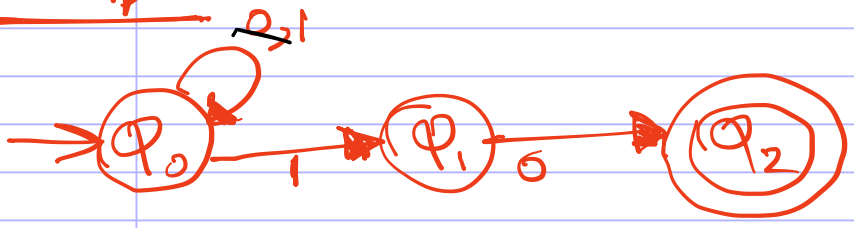
→ then table এর Right side এ new state গুলোকে Left side এ আনি।

→ new states গুলোর transition বের করব।

→ এভাবে continue করব until right side এ new কোনো state না আসে।

→ Then DFA table থেকে DFA Draw করব।

Example:



Soln:

① NFA table:

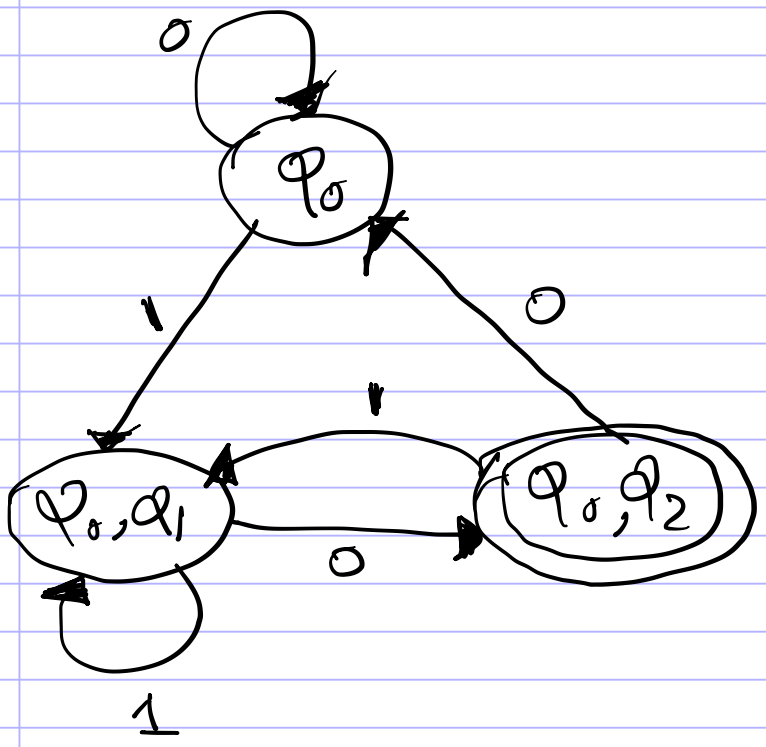
States \ I/P	0	1
→ q_0	q_0	q_0, q_1
q_1	q_2	ϕ
* q_2	ϕ	ϕ

0	q_0	q_1
	↓	↓
	q_0	q_2

① → q_0, q_1
q_0, q_1

② DFA table:

F/D / States	0	1
→ $\{q_0\}$	$\{q_0\}$	$\{q_0, q_1\}$
$\{q_0, q_1\}$	$\{q_0, q_2\}$	$\{q_0, q_1\}$
* $\{q_0, q_2\}$	$\{q_0\}$	$\{q_0, q_1\}$



Ans

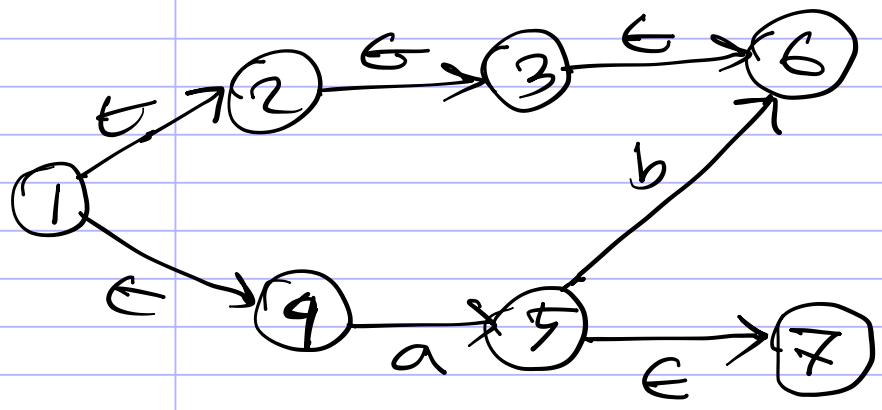
ϵ-Epsilon NFA to DFA

(*) Epsilon NFA to DFA convert এর জন্য প্রথমে E close making শিখব।

Eclose: কোনো একটা state যে নিজে এবং এর যে সকল predecessor যা ϵ-Epsilon transition হয়ে আসে তারা সবাই মিলে একসাথে E-close.

→নিচে - Eclose table বানানোর মাধ্যমে আরো clearly বিষয়টি understand করি।

Ex:



Eclose table

States	Eclose
1	1, 2, 4, 3, 6
2	2, 3, 6
3	3, 6
4	4
5	5, 7
6	6
7	7

ϵ-NFA to DFA conversion

→ এখানে ϵ-NFA to DFA conversion Process NFA to DFA এর মতোই But এখানে NFA transition সুলোর Eclose সুলো নিতে হবে।

Steps:

① প্রথমে NFA transition table এর জন্য ϵ-close Table বানাবো।

② DFA table বানাব।

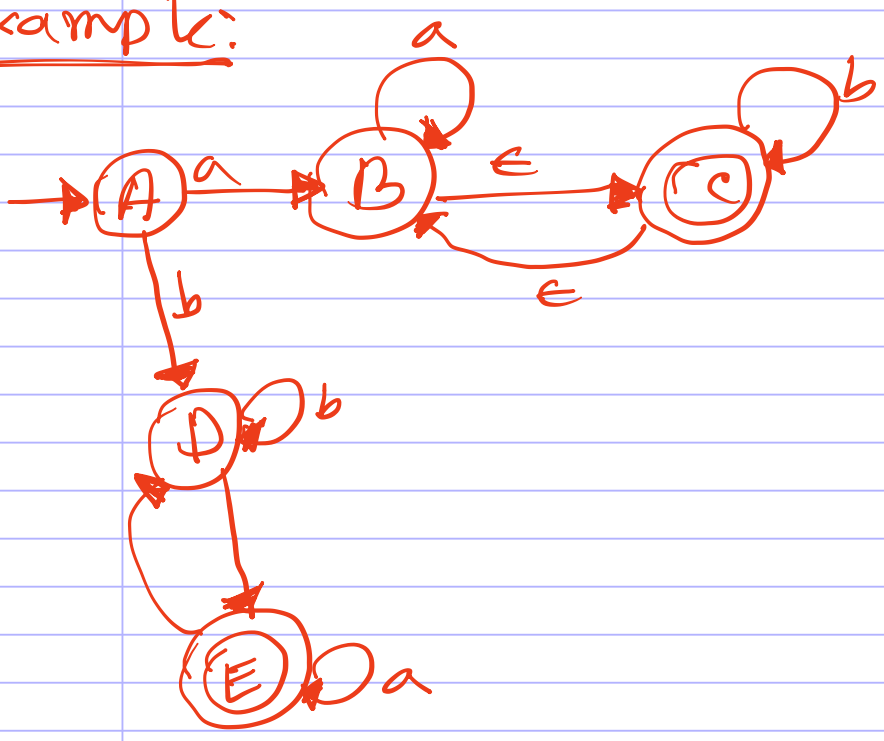
→ DFA table এর starting state হবে Eclose table এর starting starting Eclose টি।

→ Then প্রথমে NFA Transition
তে বসাব, তারপর তার Eclose নিয়ে
union DFA state হিসাবে নিব।

→ Right side এ New state
left side এ যা যা তারে continue
করব যতক্ষন না Rightside এ new
state আসা বন্ধ হয়।

→ Then DFA Table থেকে
DFA Draw করব।

Example:



Convert this ϵ-NFA to DFA.

Solution:

① NFA Table:

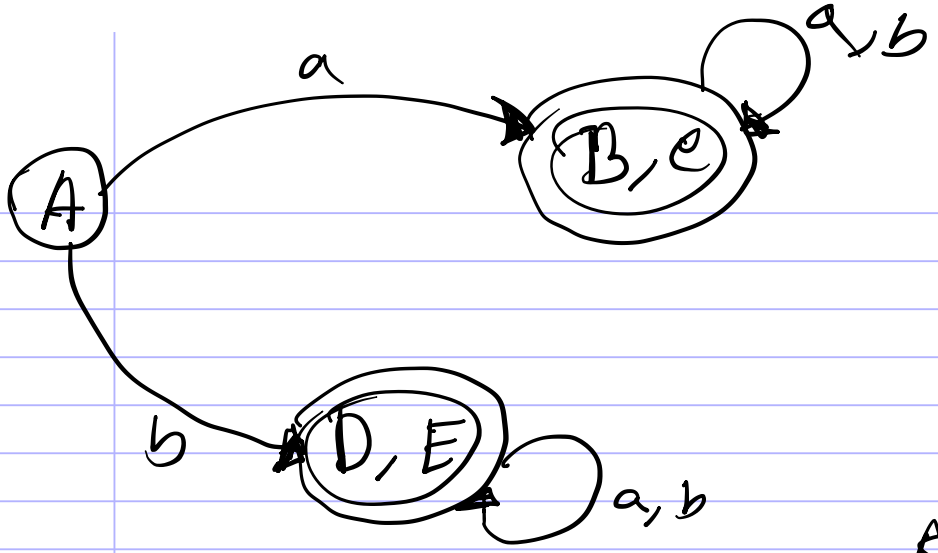
I/O / States	a	b
→ A	B	D
B	B	ϕ
* C	ϕ	C
D	ϕ	D
* E	E	ϕ

② E-closure Table

States	E closure
→ A	A
B	B, C
* C	C, B
D	D, E
* E	E, D

③ DFA Table

I/O / States	a	b
→ {A}	{B, C}	{D, E}
* {B, C}	{B, C}	{C, B}
* {D, E}	{D, E}	{E, D}



Ans

ϵ-NFA Tree

(*) Diagram হতে 2ট Table বানাবো :- (1) NFA Table (2) EClose Table

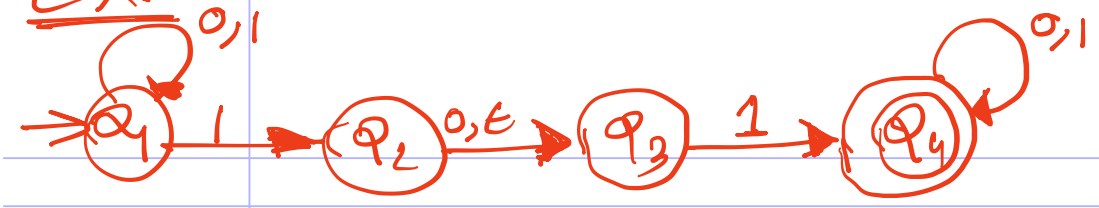
Steps:

(1) প্রথমে Diagram থেকে NFA Table এবং Eclose Table বানাবো।

(2) Then string এর left to right bit by bit করে এবং NFA transition অনুসারে E close গুলো নিয়ে Tree বানাবো।

(3) যদি sate ও Final state থাকে এবং string ফি Accepted.

Ex:



check the string "010110" is Accepted or not by Drawing NFA tree.

Ans:

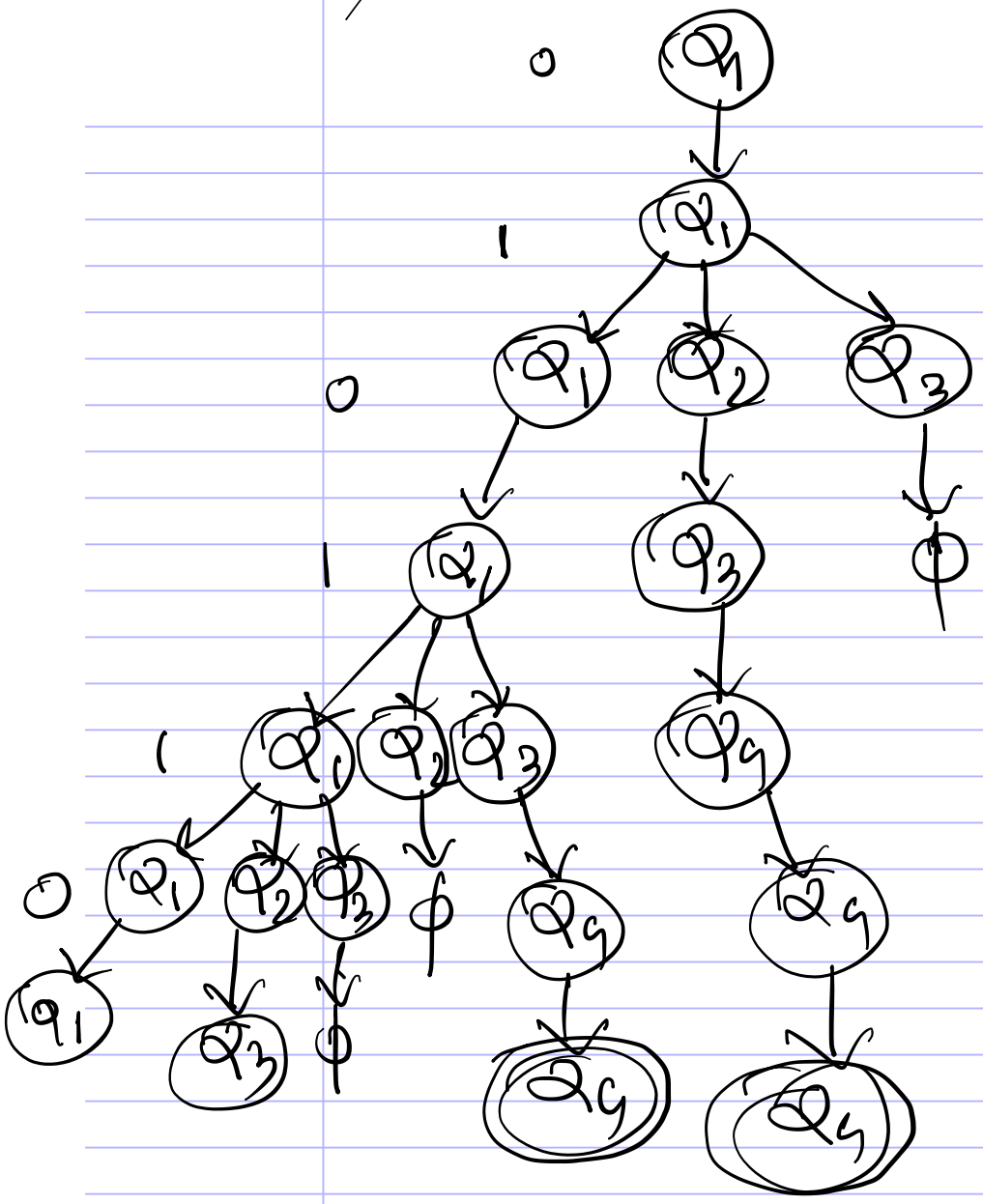
① NFA Table

States \ I/P	0	1
→ q_1	q_1	q_1, q_2
q_2	q_2	ϕ
q_3	ϕ	q_4
* q_4	q_4	q_4

② E-close Table

States	Eclose
→ q_1	q_1
q_2	q_2, q_3
q_3	q_3
* q_4	q_4

$\phi/\phi/\phi$



* The string is Accepted

(Ans)

Regular Expression

(*) All the language that are accepted by a finite Automata are regular languages.

Regular expressions are a method to represent regular languages.

(1) ϵ, ϕ - এই 2 টা হলো Regular Expression Regex.

(2) $L(E+F) = L(E) + L(F)$

(3) $L(E.F) = L(E) . L(F)$

Regex Quantifiers:

* = Zero or more

\+ = One or more

? = Zero or One

|, + = or / Either

[3] = Exact Number

[low, high] = Range of numbers

[1, 2, 3] = only Numbers

Order of Precedence

(*) closure - highest

. (And / Concatetion)

+ (union) = lowest

(*) Contains a single 1. $\Sigma = \{0, 1\}$

Ans: $0^* 1 0^*$

(*) Has atleast one 1.

Ans: $\Sigma^* 1 \Sigma^*$

(*) Contains the string "001" as a substring.

Ans: $\Sigma^* 001 \Sigma^*$

(*) Every 0 in ω is followed by atleast one 1.

Ans: $1^* (01^+)^*$

* w is a string of Even length.

$$(\Sigma\Sigma)^*$$

* w is a multiple of three.

$$(\Sigma\Sigma\Sigma)^*$$

(*) $\{01, 10\}$

$= 01 \cup 10$

(*) Starts and end with the same symbol.

$= 0\Sigma^*0 \cup 1\Sigma^*1 \cup 0 \cup 1$