Malakar Sir

'A' Day
cycle-02

10.10.18

Last Class Continuity:

if $x \geq 4$, then $2^x \geq x^2$

$$\frac{2^{x+1}}{2^x} = \frac{2^x \cdot 2}{2^x}$$

L.H.S = 2 > R.H.S = 1.53...

যতবার x এর value বাড়াব L.H.S 2 করে বাড়াব

R.H.S = 1.53 করে বাড়াব.

So, conclusion is true for the hypothesis being True.

Inductive Proof:

if $n \geq 0$,

$$\sum_{i=0}^{n} i^2 = \frac{n(n+1)(2n+1)}{6}$$

$n = 0$,

L.H.S 0 = 0 R.H.S

Inductive Step :

$$\sum_{i=0}^{n+1} i^2 = \frac{(n+1)(n+2)(2(n+1)+1)}{6}$$

$$= \frac{(n+1)(n+2)(2n+3)}{6}$$

$$\sum_{i=0}^{n} i^2 + (n+1)^2 = \frac{(n+1)(n+2)(2n+3)}{6}$$

$$\Rightarrow \frac{(n+1)(n+2)(2n+1)}{6} + (n+1)^2 = \frac{(n+1)(n+2)(2n+3)}{6}$$

.
.
.
.
.
.

এটা ঠিক থাকলেই Proof টা ঠিক আছে;

Proof by Contradiction:

if H and then not C then total is False

True False False

(#) Square of even number is even.

First even ধরি,

$(2n+1)^2$

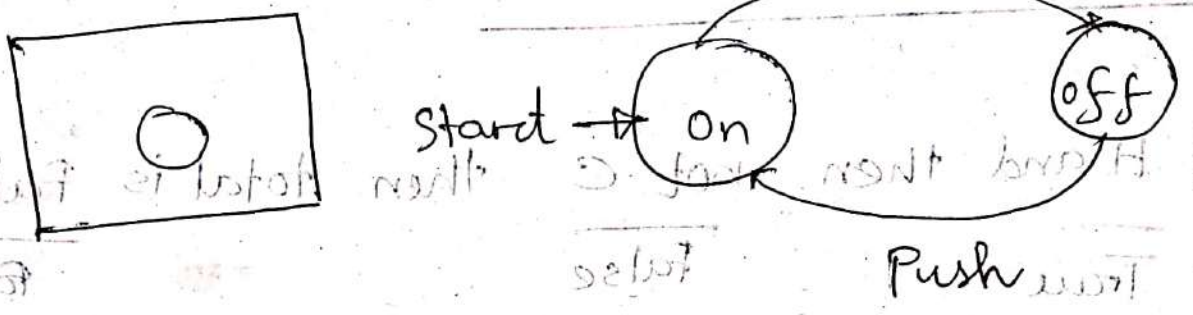
$= 4n^2 + 4n + 1$

$= 2n(2n+2) + 1$

$= 2k + 1$

So, odd Number's square is odd.

So, even number's square is even.



1. State → ON, OFF

2. Transformation of Status by input (Finite Automata)

1. Alphabets

Set of Symbols (set of Alphabets)

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

sigma

2. String:

00, 11, 10, 101, 1010, ...

empty String (ϵ) যদি কিছু 0 Symbol থাকে,

↓

0 length (কোনো Symbol নাই)

$|011| = 3$ यहाँ length of String

Language

$\Sigma^* = \{\epsilon, 0, 1, 00, 01, 10, 11, \ldots\}$

$\Sigma^1 = \{0, 1\}$ * Language में meaningful Strings (यहाँ)

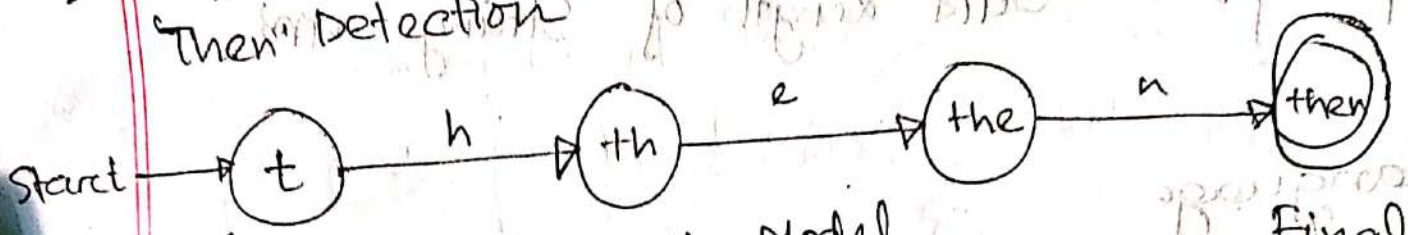
$\Sigma^2 = \{10, 11, 00\}$

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

$L \in \Sigma^*$ / $L \subseteq \Sigma^*$

Sequence Reading

"Then" Detection



Finite Model

Final State Double Circle এ প্রকাশ করা হয়

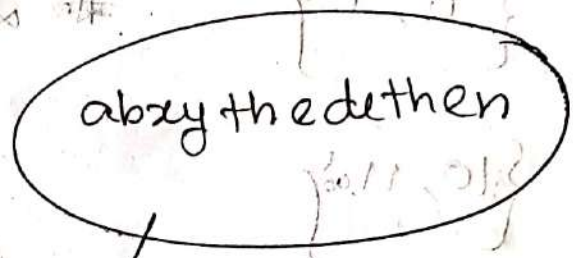
এরকম State থাকতে পারে

t

th

the

then



এগুলো হল Finite Automata যে, যে input sequence থেকে নির্দিষ্ট sequence Read করতে পারবে।

1. Set of Status, Q
2. Set of Inputs
3. Start Node, q_0 ($q_0 \in Q$ set এর Member)
4. Set of Final States (অনেক Final State থাকতে পারে)
5. Transition function (δ)

(#) Transition Function হচ্ছে এমন যার মাধ্যমে Next State এ যেতে পারব কিনা। Depends on current State Next State input

To read 'thats'

যদি -
input sequence → that hat hats

(#) 't' এর ক্ষেত্রে 't' কিভাবে বুঝব?

এর জন্য দুই type automata আছে।

⇒ Deterministic Finite Automata
(একটা state থাকবে current state)
⇒ Non-deterministic Finite Automata
(multiple State current state
থাকতে পারে)

Shakhar Sir

'B' Day
Cycle-02

13/10/18

Q = Set of states

Σ = Set of input symbols

δ = Transition function

q_0 = Starting State

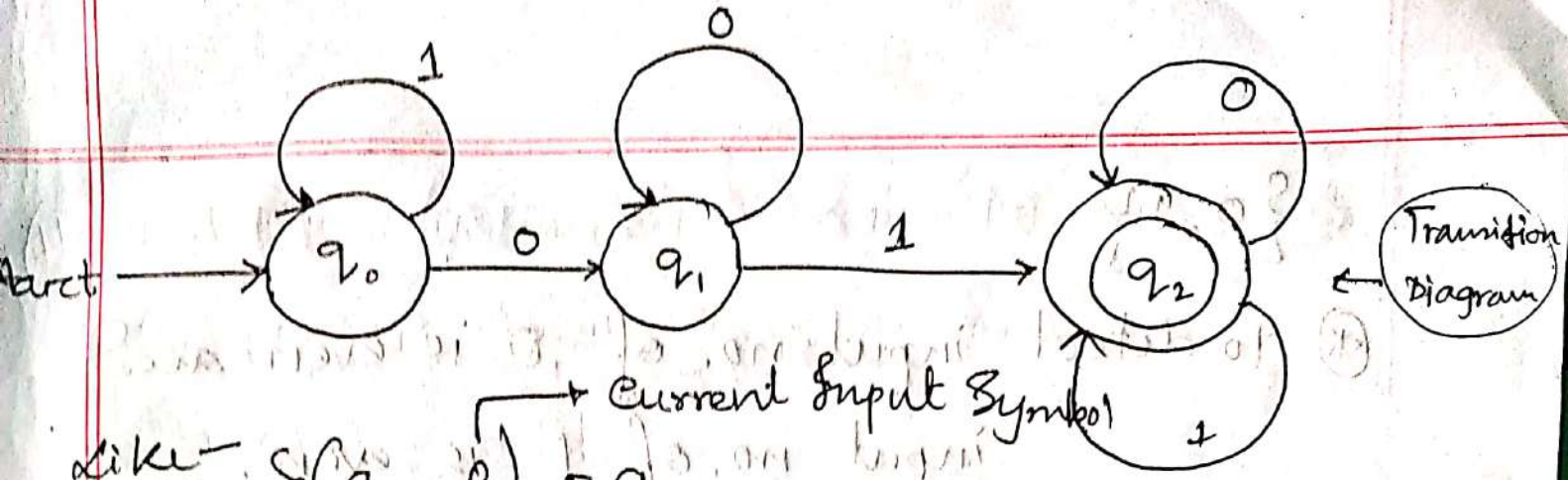
F = Final States

01

Suchas- x01y

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

1. 01 Never seen
2. 0 seen, 1 unseen
3. 01 seen



Like $\delta(q_0, 0) = q_1$

↓ Current State

$\delta(q_0, 1) = q_0$

Let, the name of finite automata be A.

$$A = (\{q_0, q_1, q_2\}, \{0, 1\}, \delta, q_0, \{q_2\})$$

state \ input	1	0
→ q_0	q_0	q_1
q_1	q_2	q_1
* q_2	q_2	q_2

Final node (*)

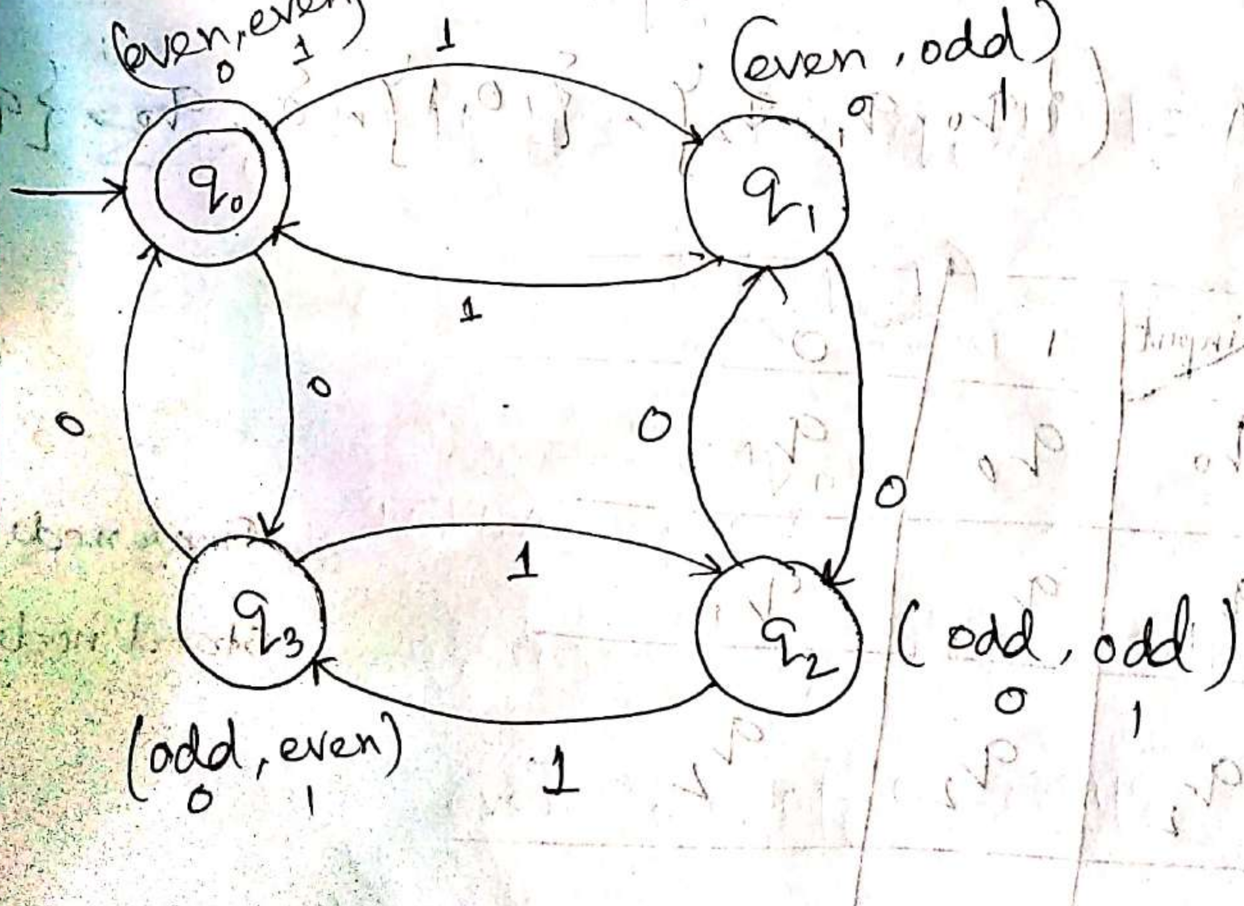
Start node (→)

{0, 1}

(*) To ditect input no. of 0 is even and input no. of 1 is even.

1. 0 even 1 even
2. 0 even 1 odd
3. 0 odd 1 even
4. 0 odd 1 odd

{ input starting node No. {0,0} ধরা হল that means 0 no. of 0 and 0 no. of 1. So both are even.



1 1 0 1 1 0 (~~0~~ n)

starting state ($= q_0$)

$$\hat{\delta}(q_0, \epsilon) = q_0$$

$$\hat{\delta}(q_0, 1) = \delta(\hat{\delta}(q_0, \epsilon), 1)$$

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

$$= q_1$$

$$\hat{\delta}(q_0, 11) = \delta(\hat{\delta}(q_0, 1), 1)$$

$$= \delta(q_1, 1)$$

$$= q_0$$

$$\hat{\delta}(q_0, 110) = \delta(\hat{\delta}(q_0, 11), 0)$$

$$= \delta(q_0, 0)$$

$$= q_3$$

$$\hat{\delta}(q_0, 1101) = \delta(\hat{\delta}(q_0, 110), 1)$$

$$= \delta(q_2, 1)$$

$$= q_2$$

$$\hat{\delta}(q_0, 1101\,0) = \delta(\hat{\delta}(q_0, 1101), 0)$$

$$= \delta(q_2, 0)$$

$$= q_1$$

$$\hat{\delta}(q_0, 11010\,1) = \delta(\hat{\delta}(q_0, 11010), 1)$$

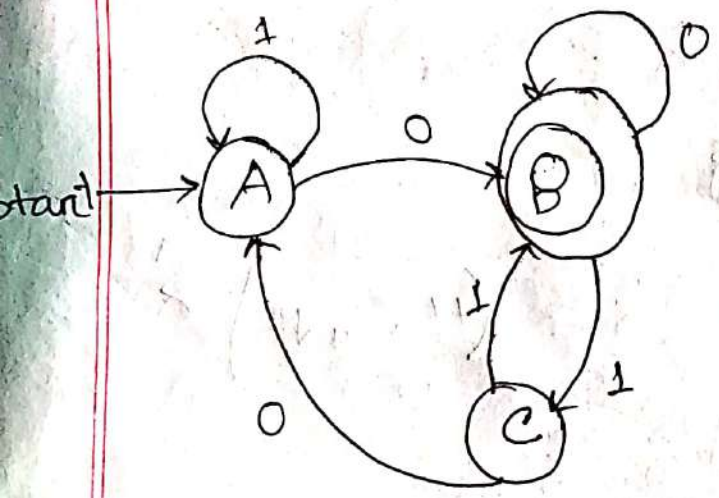
$$= \delta(q_1, 1)$$

$$= q_0$$

~~$\hat{\delta}(q_0, 110101010) = \delta(\hat{\delta}$~~

So, Final node যেহেতু q_0 আসছে So, even no. of 1 & even no. of 0 আছে।

	0	1
→ A	B	A
* B	B	C
C	A	B



[দুই প্রকার Deterministic Finite Automata]

Shankhar Sir LAB-01

'C' Day

Cycle-02

14/10/18

Q.1. Assume,

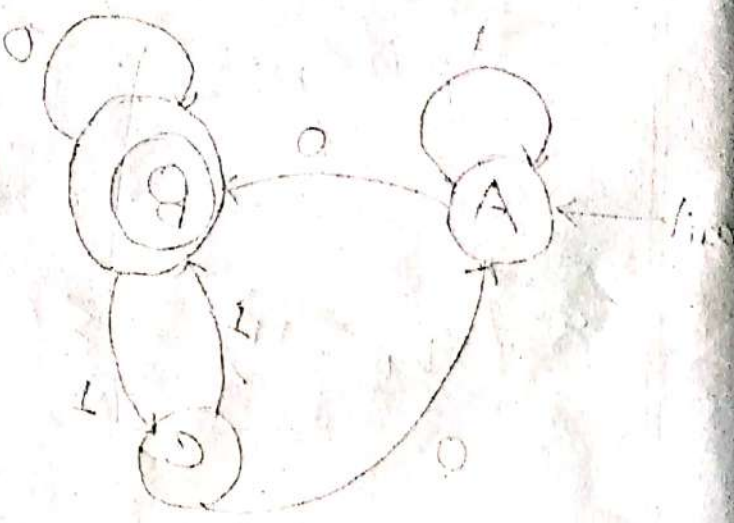
$Q = \{q_0, q_1, q_2\}$

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

$q_0 = q_0$

$f = \{q_2\}$

	0	1
q_0	q_0	q_1
q_1	q_2	q_1 *
q_2	q_1	q_2



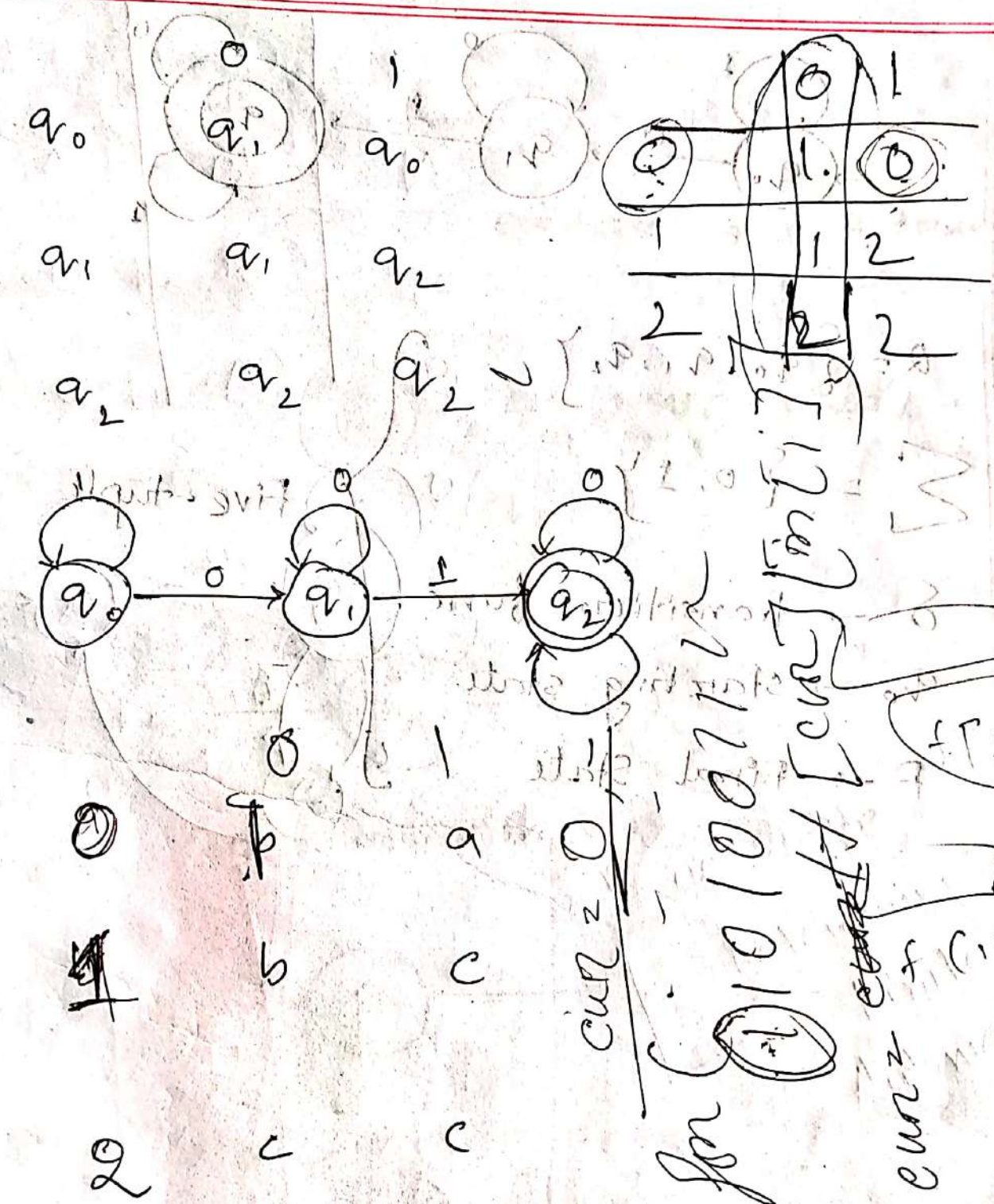
Input:

Transition Table

Output:

(I) Compute δ

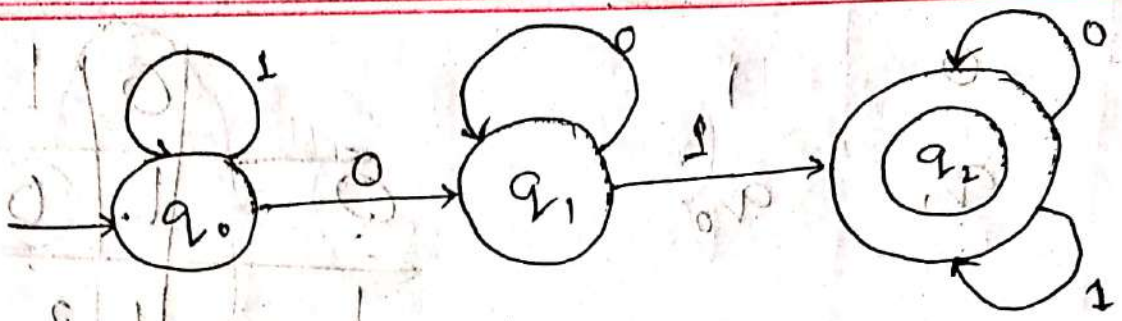
(II) for any input sequence compute state using δ.



'E' Day

Cycle-02

28.10.18



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

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

$\delta \rightarrow$ transition func.

$q_0 \rightarrow$ starting state

$F \rightarrow$ Final State

} Five-tuple

Non-deterministic Finite Automata

* Det. এর ক্ষেত্রে ১ একটা মাত্র state এ transit করতে পারে।

* Not det. কোনো নাও নিতে পারে / multiple state এ নিতে পারে।

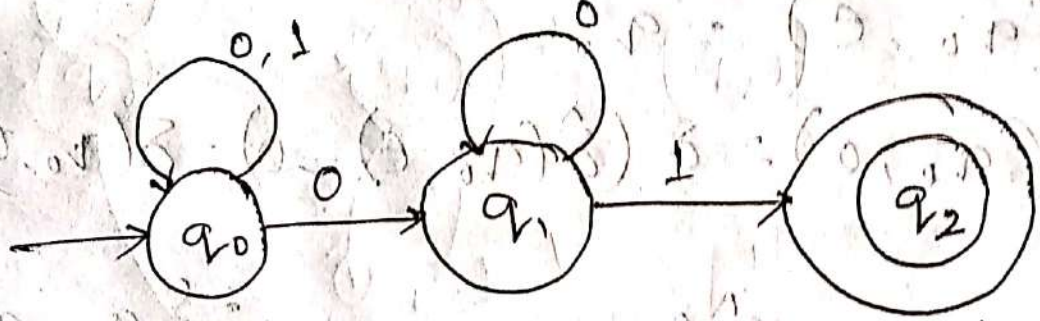
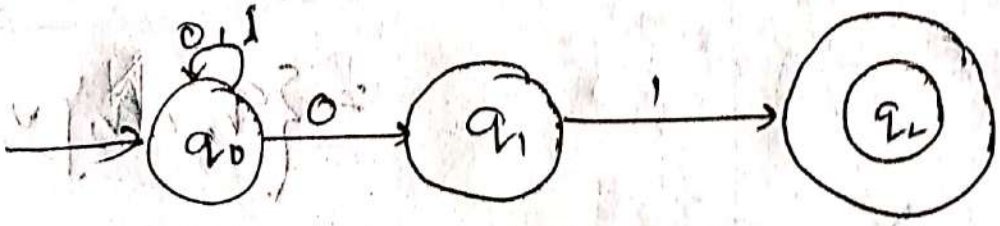


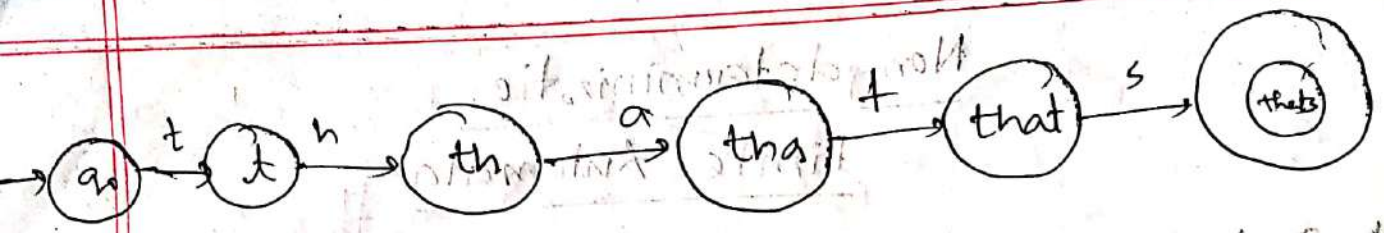
Fig: 01 (combination আছে কী না?



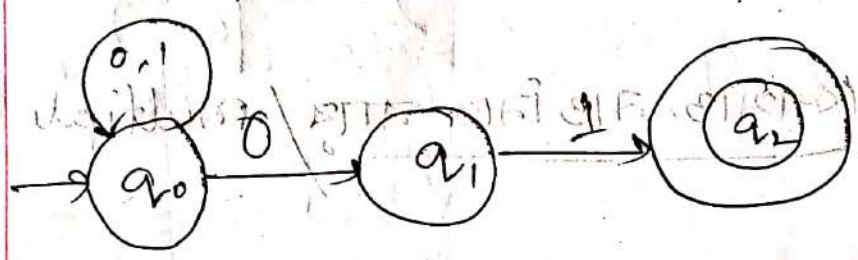
01 $q_0 \to q_0 \to q_0 \xrightarrow{1} q_0 \to q_0$

$q_0 \to q_1 \to q_2$ (stuck)

$q_0 \to q_1$ (stuck)



thathats



input
00101

$\hat{\delta}(q_0, \epsilon) = q_0$

$\hat{\delta}(q_0, 0) = \delta(\hat{\delta}(q_0, \epsilon), 0) = \delta(q_0, 0)$

$= \{q_0, q_1\}$

$\hat{\delta}(q_0, 00) = \delta(\hat{\delta}(q_0, 0), 0) = \delta(q_0, 0) \cup \delta(q_1, 0)$

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

$\hat{\delta}(q_0, 001) = \{q_0, q_1\}$

$= \delta(\hat{\delta}(q_0, 00), 1)$

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

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

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

$= \{q_0, q_2\}$

$$\hat{\delta}(q_0, 0010) = \delta\left(\hat{\delta}(q_0, 001), 0\right)$$

$$= \delta(\{q_0, q_2\}, 0)$$

$$= \delta(q_0, 0) \cup \delta(q_2, 0)$$

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

$$= \{q_0, q_1\}$$

$$\hat{\delta}(q_0, 00101) = \delta(\hat{\delta}(q_0, 0010), 1)$$

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

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

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

$$= \{q_0, q_2\}$$

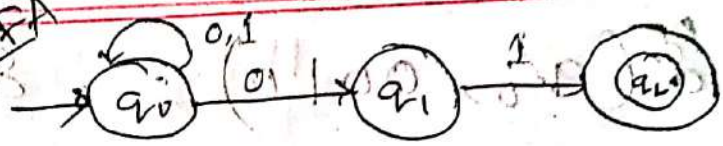
'A' Day

Cycle-03

29.10.18

Transition Table:

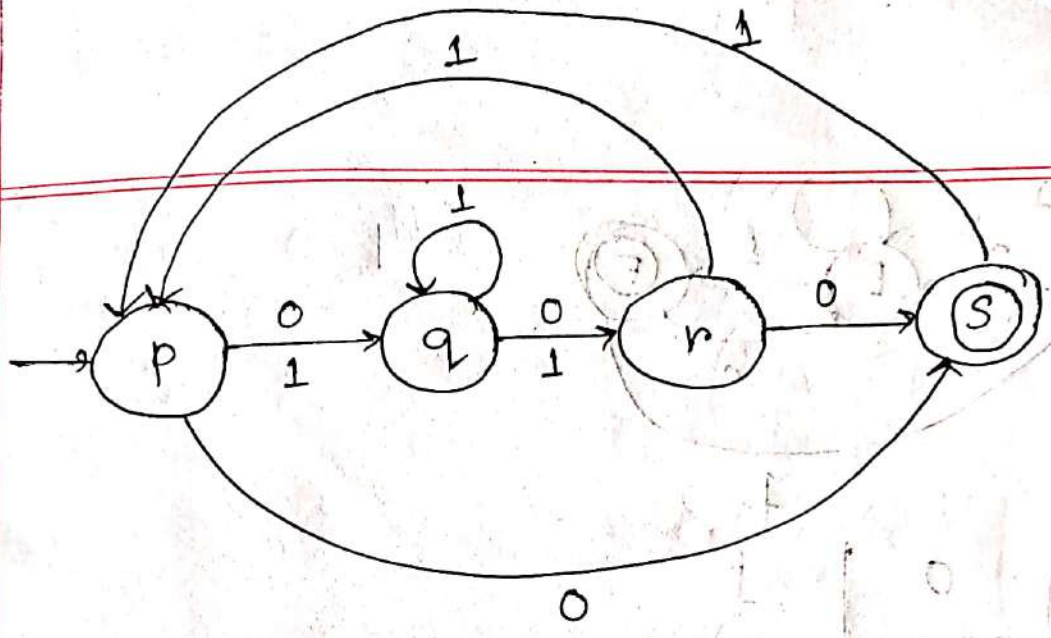
NFA



Input / States	0	1
→ q_0	$\{q_0, q_1\}$	q_0
q_1	ϕ	q_2
* q_2	ϕ	ϕ

(#) Draw the graph of given table -

I/P / States	0	1
→ p	{q, s}	{q}
q	{r}	{q, r}
r	{s}	{p}
* s	ϕ	{p}

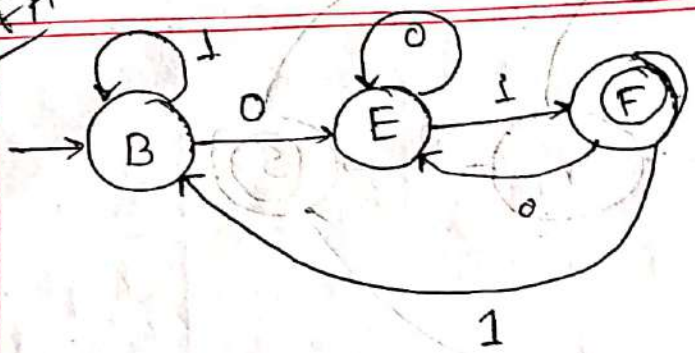


(*) 2^n সংখ্যক combination হবে যাবে

n = no. of states . For this- such as -

		σ	
ϕ	A	ϕ A	ϕ A
→ q_0	B	$\{q_0, q_1\}$ E	$\{q_0\}$ B
q_1	C	ϕ A	$\{q_2\}$ D
* q_2	D	ϕ A	ϕ A
$\{q_0, q_1\}$	E	$\{q_0, q_1\}$ E	$\{q_0, q_2\}$ F
* $\{q_0, q_2\}$	F	$\{q_0, q_1\}$ E	$\{q_0\}$ B
* $\{q_1, q_2\}$	G	ϕ A	$\{q_2\}$ D
* $\{q_0, q_1, q_2\}$	H	$\{q_0, q_1\}$ E	$\{q_0, q_2\}$ F

DFA



	0	1
→ q_0	$\{q_0, q_1\}$	q_0
q_1	ϕ	q_2
* q_2	ϕ	ϕ

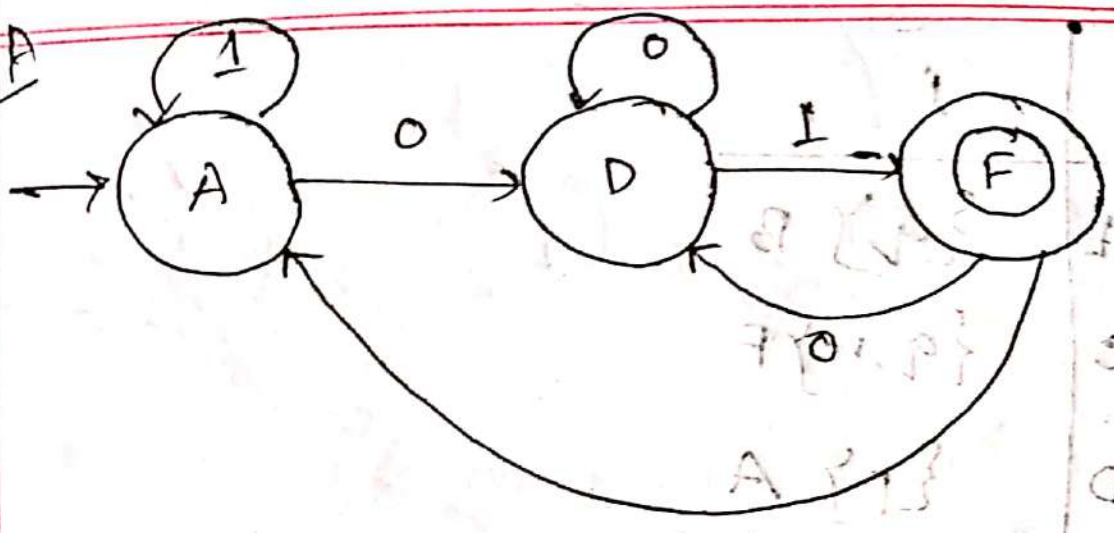
	0	1
→ A	D	A
B	E	C
* C	E	E
D	D	F
E	E	E
* F	D	A

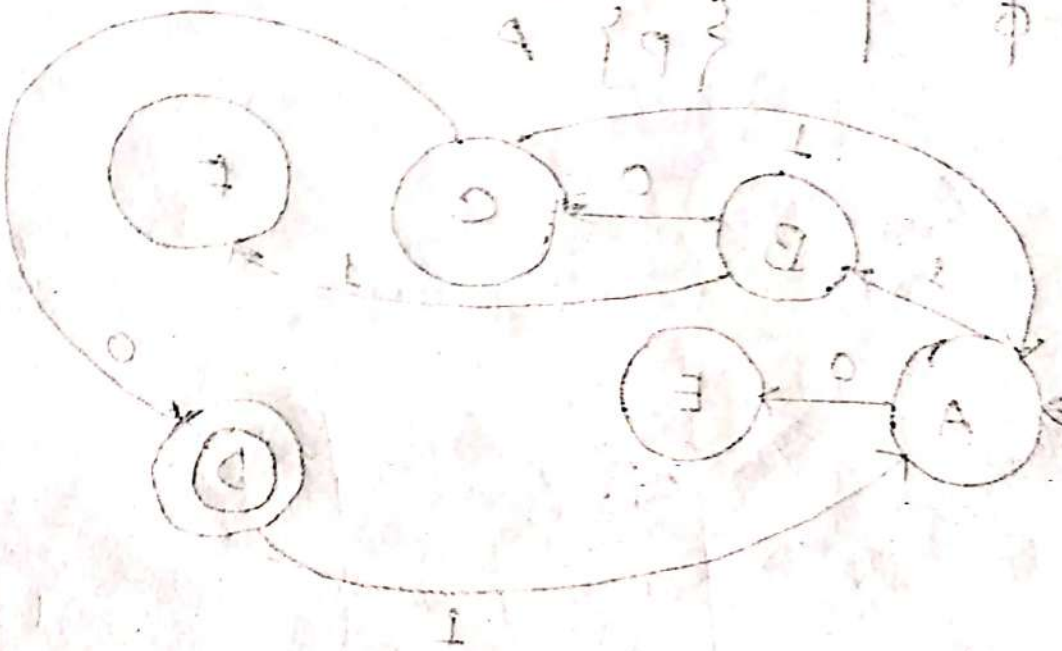
$\{q_0, q_2\} = F$

$\phi = E$

$\{q_0, q_1\} = D$

DFA





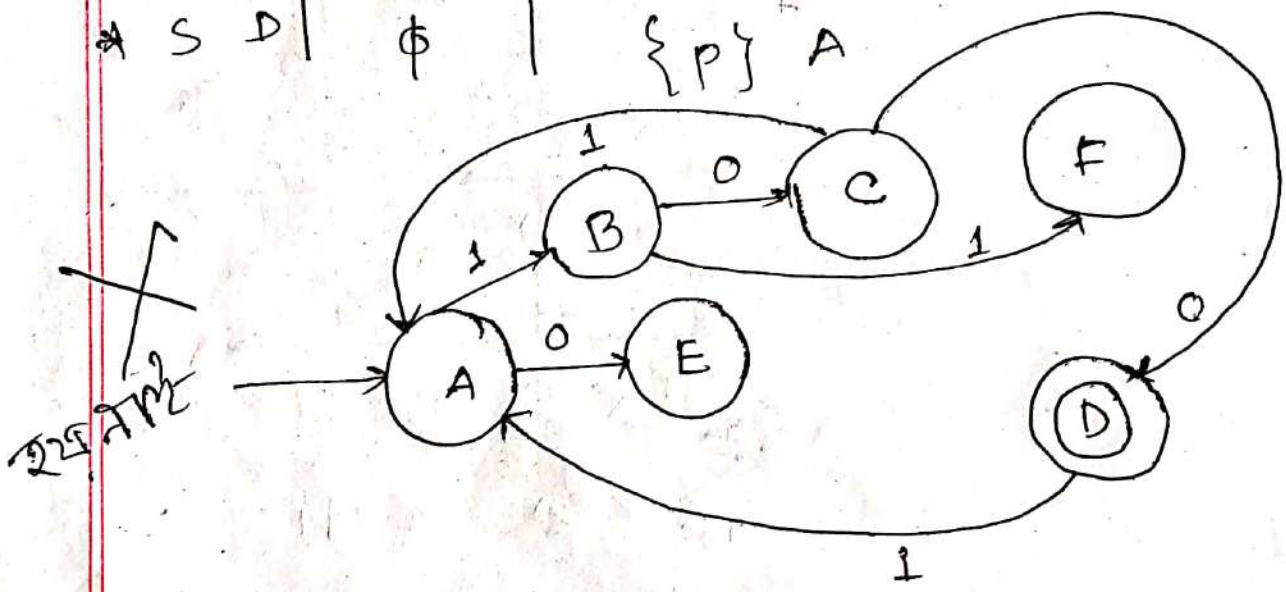
Scanned by CamScanner

'B' Day

30.10.18

cycle-03

		0	1
→	p A	{q, s} E	{q} B
*	q B	{r} C	{q, r} F
	r C	{s} D	{p} A
*	s D	ϕ	{p} A



		0	1
→	A	E	B
	B	C	F
	C	D	A
*	D	ϕ	A

		0	1
→	p A	{q, s} E	{q}
*	q B	{r}	{q, r} F
	r C	{s}	{p}
*	s D	ϕ 4	{p}

	0	1
→ A	E	B
B		
C		
D		
E	C	{P, q, r} H
F		
4		
H		

Full করব বাসায়

ϵ-NFA

ϵ - empty string (কিছুবারে খালি না)

ϕ - nothing (কিছুবারে খালি)

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

↓

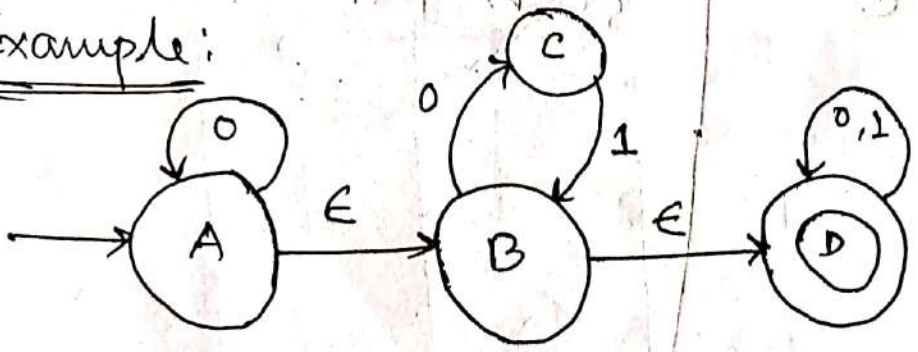
(current state, input)

For DFA $Q \times \Sigma \rightarrow$ next state

NFA $Q \times \Sigma \rightarrow 2^Q$

ϵ-NFA $Q \times \Sigma \cup \{\epsilon\} \rightarrow 2^Q$

Example:



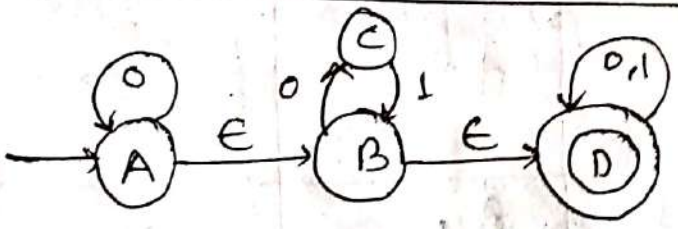
epsilon closure:

Eclose (A) = {A, B, D}

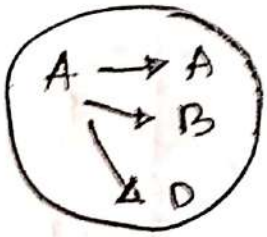
(*) সকল DFA ≅ NFA কিন্তু সকল NFA, DFA নয়।

(*) ∈-NFA → NFA convert করতে ∈ বাদ দিতে হবে।

Conversion of ∈-NFA → NFA:



	0	1
A	A	ϕ
B	C	ϕ
C	ϕ	B
D	D	D



Epsilon বিহীন

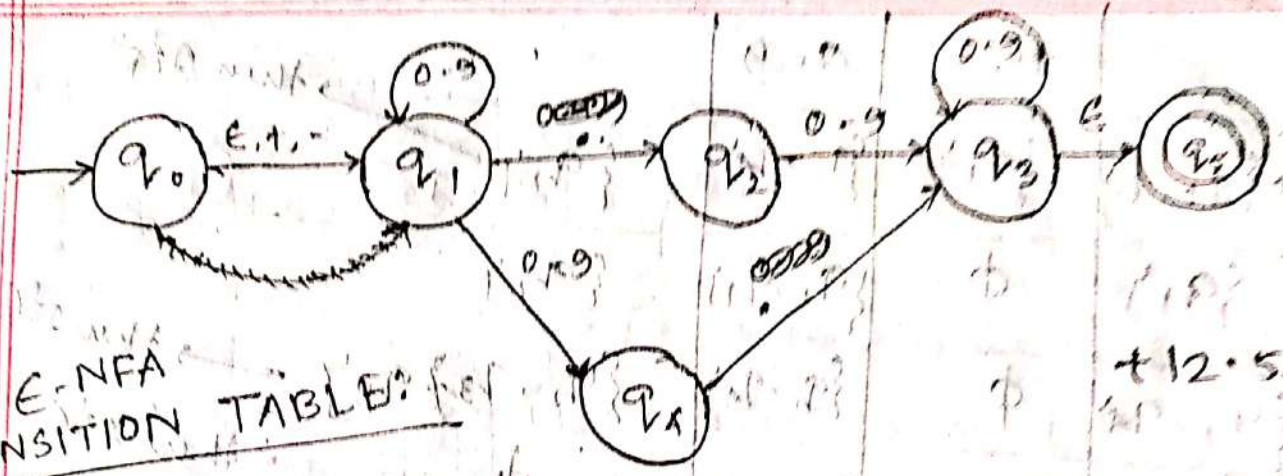
	0	1
A →	A	A,B,D
B	C,D	
D	D	

Shekhar Sir

'E' Day

Cycle-03

1/11/18



+12.50

ϵ-NFA TRANSITION TABLE:

	+, −	0-9	.	ϵ
→ q_0	$\{q_1\}$	ϕ	ϕ	$\{q_1\}$
q_1	ϕ	$\{q_1, q_4\}$	$\{q_2\}$	ϕ
q_2	ϕ	$\{q_3\}$	ϕ	ϕ
q_3	ϕ	$\{q_3\}$	ϕ	$\{q_5\}$
q_4	ϕ	ϕ	$\{q_3\}$	ϕ
* q_5	ϕ	ϕ	ϕ	ϕ

$$\Sigma = \{\epsilon, ., +/-, 0-9\}$$

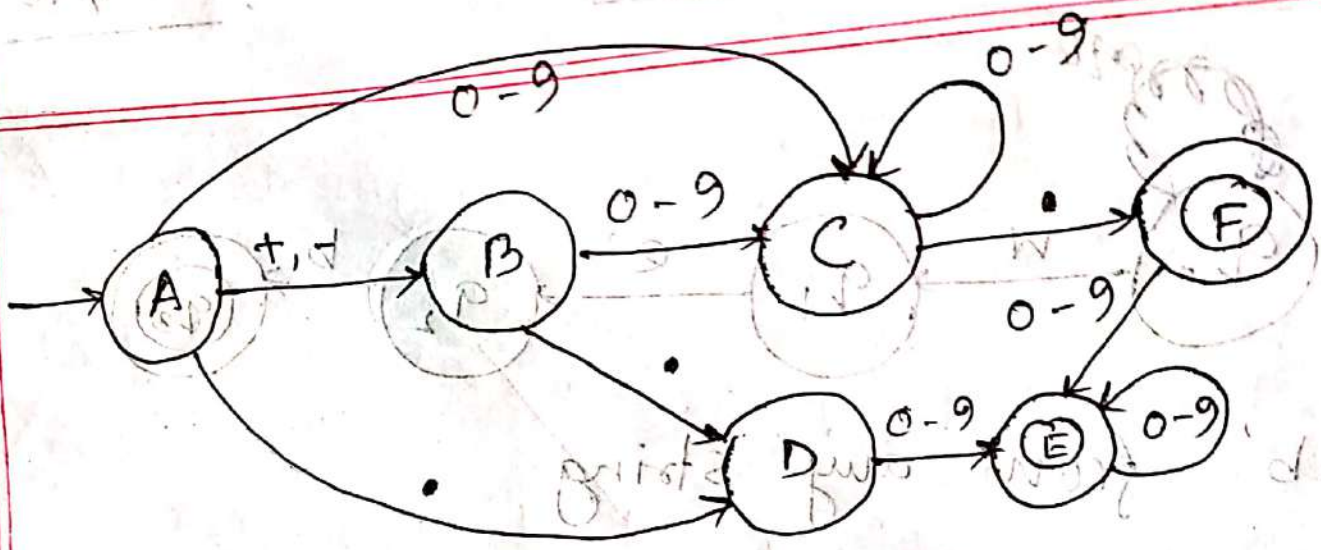
ε-NFA to DFA:

	+, −	0-9	.
→ A {q0, q1}	{q1}	{q1, q4}	{q2}
B {q1}	ϕ	{q1, q4}	{q2}
C {q1, q4}	ϕ	{q1, q4}	{q2, q3, q5}
D {q2}	ϕ	{q3, q5}	ϕ
*F {q2, q3, q5}	ϕ	{q3, q5}	ϕ
*E {q3, q5}	ϕ	{q3, q5}	ϕ
ϕ	ϕ	ϕ	ϕ

Now Renaming:

	+, −	0-9	.
→ A	B	C	D
B	ϕ	C	D
C	ϕ	C	F
D	ϕ	E	ϕ
*E	ϕ	E	ϕ
*F	ϕ	E	ϕ

Ending State যে যে অবস্থান সেগুলো



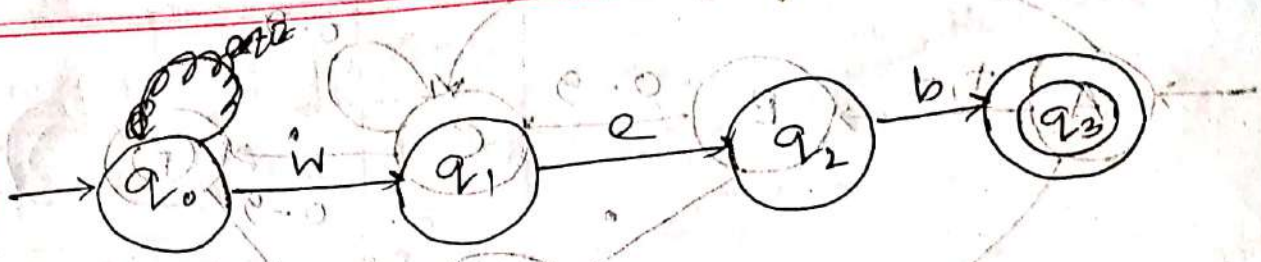
কোনো state থেকে কোনো input ছাড়া যেসব state এ যাওয়া যায়, তাকে হচ্ছে ε closure

যেমন: q_0 to q_1, - q_3 to q_5

Shekhar Sir

'A' Day
cycle-04

5/11/18



(*) 'web' from any string.

Proof by Contradiction: (1st class revision)

If H, then C

If H and not C lead falsehood

(*) If n^2 is even, then n is even,

let,
n is not even

So, $n = (2k+1)$

Now, $n^2 = (2k)^2 + 2.2k.1 + 1$

$= 4k^2 + 4k + 1$

$= 2(2k^2 + 2k) + 1$

"If n^2 is even, n is odd" is always false

So,
from contradiction we can say;

'If n^2 is even, n is even'

CT-01 (Syllabus)

Chap- 1

Art 1·1, 1·2, 1·5

Chap-2

Art 2·2, 2·2·1, 2·2·2, 2·2·3, 2·2·4 Example 2·4

Art 2·3, 2·3·1, 2·3·2, 2·3·3, 2·3·4, 2·3·5,

2·4·2, 2·5, 2·5·1 - 2·5·5

Shekhar Sir

'B' Day

Cycle-04

6/11/18

Regular Expression

Languages

$L = \{001, 01, 111\}$

$M = \{\epsilon, 01\}$

1) Union operator $L \cup M = \{\epsilon, 001, 01, 111\}$

Intersection $L \cap M = \{01\}$

2) Concatenation $L \cdot M = \{001, 01, 111, 00101, 01011101\}$

3) Closure operator

Let,

$L = \{0, 11\}$

$L^0 = \{\epsilon\}$

$L^1 = \{0, 11\}$

$L^2 = \{00, 011, 110, 1111\}$

$L^3 = \{000, 0110, 0011, \ldots\}$

$L^* = L^0 \cup L^1 \cup L^2 \cup L^3 \cup \ldots \ldots \cup L^n$

Basis

$\epsilon \quad \phi$

যে কোনো symbol হল Regular Expression

$L(\epsilon) = \{\epsilon\}$

$L(\phi) = \phi$

$L(0) = \{0\}$

$L(0^*) = \{\epsilon, 0, 00, 000, \ldots\}$

0 বার, 1 বার, 2 বার, 3 বার

0 বার নিচ্ছে

* L এর ক্ষেত্রে যা লিখবে সেটা Bold/Italic করে দিতে হবে।

গুরুত্বপূর্ণ নিয়ম -

$L \cup M = L + M$ ⟶ Union

$L.M = LM$ ⟶ Concaderation

$a^*e = \{\epsilon, a, aa, aaa, \ldots\}\{e\}$

$= \{e, ae, aae, aaae, \ldots\}$

$(01)^* = \{\epsilon, 01, 0101, 010101, \ldots\}$

⊛ Coding এর জন্য Regular Expression Website -

www.regular-expressions.info

Sample of JavaScript Code:

```
var x = /a*e/
x.test("..... ae .....")
```

⟶ True/False Return করবে।

^ae$ ⟶ এবার কেবল exact আসবে

↓

এবারও exact আসবে

a = 1
y = 5

y - a + 1

* এর স্থানে যেকোনো অংখ্যুক্তার যেকোনো Symbol হতে পারে.

LAB-02

'c' Day

Cycle-04

10/1/18

aaaae

```
for (i = 0 ; i < x; i++)
    j = 0
{ if (i == 0)
  { if (regx[0] == input[0])
    { continue;
    } }
else if (regx[i] == "*")
{ while (j == a[i-1])
  { j++;
  }
}
```

```
else if (regx[i] != "*"
        && i != 0)
{
  if (regx[i] == input
                [j]
  { j++;
    continue; }
  }
  else
  { break; } }
```

Shakhar Sir

'E' Day
Cycle-01

12/11/18

Regular Expression:

1) Union
2) Concatenation
3) Closure/Star/ Kleene closure

$L = \{001, 11, 01\}$ → Language

$M = \{\epsilon, 01\}$

$L \cup M = \{001, 11, 01, \epsilon\}$

$L \cdot M = \{001, 11, 01, 00101, 1101, 0101\}$

L^* $\quad L^0 = \{\epsilon\}$

$L^1 = \{001, 11, 01\}$

$L^2 = \{00\,1001, 00\,11, \ldots\}$

$\rightarrow L^* = L^0 \cup L^1 \cup L^2 \cup L^3 \cup \ldots$

এখন Algebraic form এ লিখব-

LUM হবে L+M

Like 0 + 1 is $\{0\} \cup \{1\}$

L.M হবে LM

যেমন: কোথাও 01 থাকলে $\{0\}.\{1\}$ ← Concatenation

0^* থাকলে ⟶ $= \{\epsilon, 0, 00, 000, \ldots\}$

$0^0 = \{\epsilon\}$ -আর 0^1 থাকলে ⟶ $= \{1, 01, 001, 0001, \ldots\}$

$0^1 = \{0\}$

Precedence

Presidence of these operations:

Suppose - $0^* 1 + 1^* 0$ আছে so, কার কাজ আগে করব, তাই Presidence

1) Star (Kleen Closure) ← সবার আগে
2) Dot (Catenation)
3) Plus (Union) ← সবার শেষে

Like-

$0^* 1 + 1^* 0$

$\{\epsilon, 0, 00, \ldots\} 1 + \{\epsilon, 1, 11, \ldots\} 0$

$= \{1, 01, 001, \ldots\} + \{0, 10, 110, \ldots\}$

যদি কোনো প্রশ্নে অনেক Option এর মাঝে একটা লাগে -

$(01)^* + (10)^* + 1(01)^* + 0(10)^*$

0 দিয়ে শুরু 1 দিয়ে শেষ	1 দিয়ে শুরু 0 দিয়ে শেষ	1 দিয়ে শুরু 1 দিয়ে শেষ	0 দিয়ে শুরু 0 দিয়ে শেষ

$\{\epsilon, 01, 0101, \ldots\} + \{\epsilon, 10, 1010, \ldots\}$

$+ \{01, 101, 10101, \ldots\} + \{0, 010, 01010, \ldots\}$

$(01)^* 0$

'B' Day

14/11/18

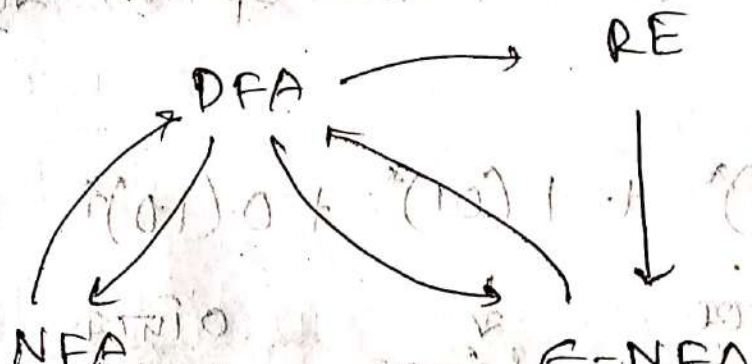
Cycle-05

$(1+0)^* 1$

$(\{1\} \cup \{0\})^* 1$

$= (\{\epsilon, 1, 11, 111, 1111, \ldots\} \cup \{\epsilon, 0, 00, \ldots\})^* 1$

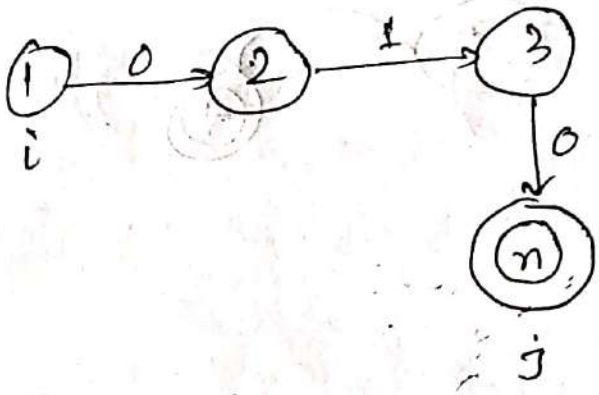
$= \{1, 11, 111, 1111, 11111, \ldots\} \cup \{1, 01, 001, 0001, \ldots\}$



Let,

$L(A)$ = Language of DFA

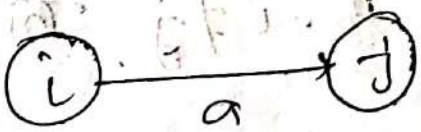
$L(R)$ = ... RE



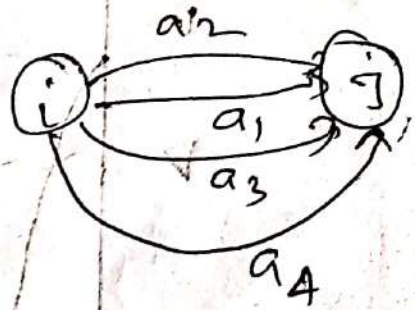
Rij

ধরলাম $n = k$, যদি i থেকে k তে যেতে হয় তাহলে max. All State লাগতেছে. আর সর্বোচ্চ কোনো step না নিলে i থেকে j তে যেতে হয় $i \to j$ হচ্ছে or i আর j same হবে. সেখানে কোনো দাম (Path) নাই,

1. No path
$R_{ij}^{(0)} = \phi$. (Directly কোনো Path নাই, So যেতে পারতেছে না)



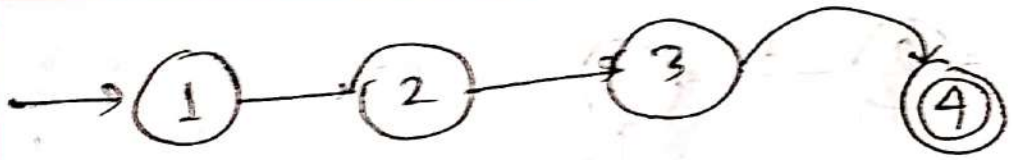
2. Only path → মানে হচ্ছে শুধু যদি একটা Path থাকে
$R_{ij}^{(a)} = a$



3. a_1 or a_2 or a_3 or a_4

That means.

$R_{ij} = a_1 + a_2 + a_3 + a_4$



$k=4 \rightarrow R_{ij}^{(3)}$

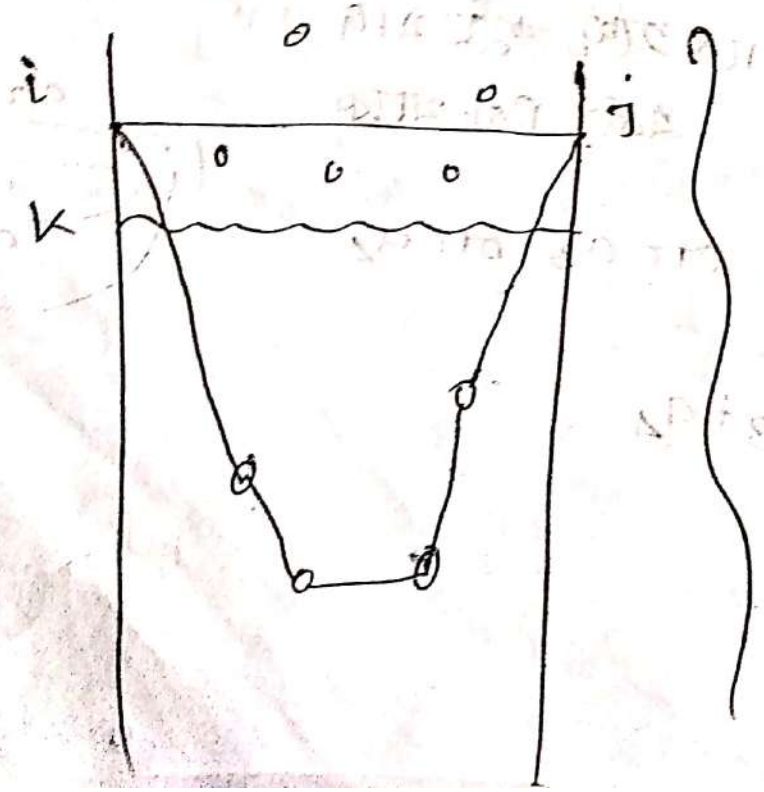
$k=3 \rightarrow R_{ij}^{(2)}$

$k=2 \rightarrow R_{ij}^{(1)}$

$k=1 \rightarrow R_{ij}^{(0)}$

Directly যাওয়া যাবে-

1. Direct path
2. $i=j$ হলে
3. $i \neq j$ হলে আগের দুইবার উঁচু Condition



k এর চেয়ে বেশি Value Stati Use করতে পারবা না।

যেকোনো সংখ্যকবার Repeat হতে পারে...

1) Direct $R_{ij}^{(k)} = R_{ij}^{(k-1)}$

2) Many

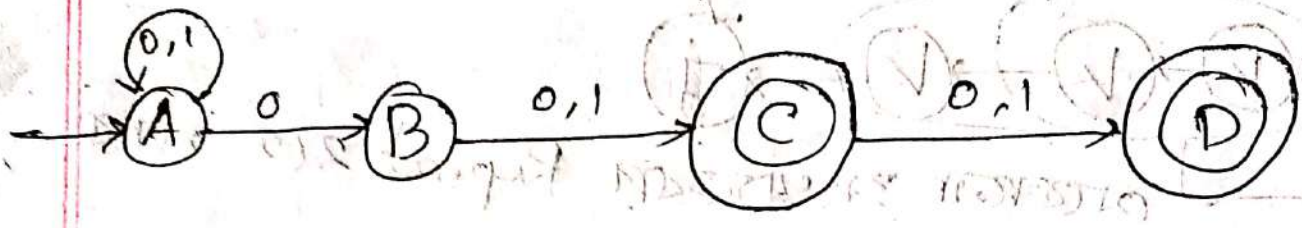
$R_{ij} = R_{ik}^{(k-1)} \cdot R_{kk}^{(k-1)} \cdot R_{kkj}^{(k-1)} \cdot R_{kj}^{(k-1)}$

$R_{ik}^{(k-1)} \left(R_{kk}^{(k-1)}\right)^{*} \cdot R_{kj}^{(k-1)}$

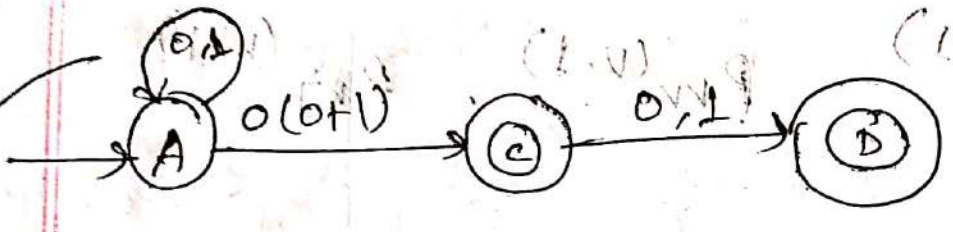
So,

$$R_{ij}^{(k)} = R_{ij}^{(k-1)} + R_{ik}^{(k-1)} \cdot \left(R_{kk}^{(k-1)}\right)^{*} \cdot R_{kj}^{(k-1)}$$

Article theorem :- 3.4



$0 \cdot (0+1)$



$(0+1)^*$ $0(0+1) \cdot (0+1)$ D পর্যন্ত

আবার,

$(0+1)^*$ $0(0+1)$ C পর্যন্ত

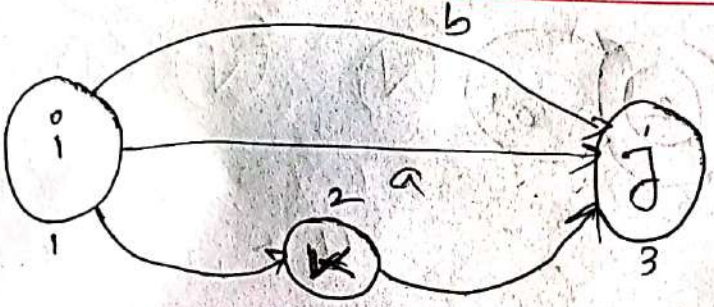
Thus, total Result

$= (0+1)^* \cdot 0 \cdot (0+1) + (0+1)^* 0 (0+1) \cdot (0+1)$

Shakhar sir

'B' Day
cycle-06

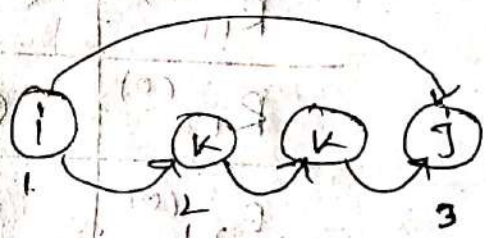
24/11/18



$R_{ij} = \phi$ (কোনো Path নাই)

$R_{ij} = a$ (একটা Path আছে)

$R_{ij} = a+b$ (Multiple Path)



যদি, i to j direct না গিয়ে k হয়ে যায়?

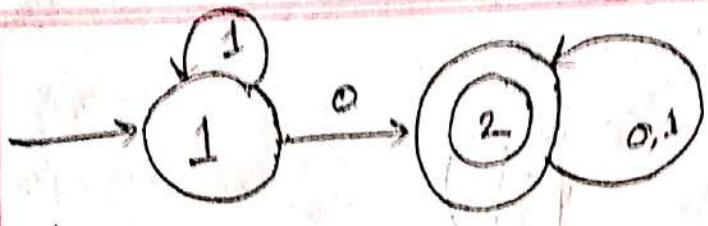
$$R_{ij}^{(k)} = R_{ij} + R_{ik}(R_{kk})^* R_{kj}$$

Actually -

$$R_{ij}^{(k)} = R_{ij}^{(k-1)} + R_{ik}^{(k-1)} \left(R_{kk}^{(k-1)}\right)^* R_{kj}^{(k-1)}$$

Why (k-1) why not k?

⇒ কারণ k কে Pass করতে পারব না।



Let, $k=0$

Possibility –

$R_{11}^{(0)}$	$\epsilon+1$
$R_{12}^{(0)}$	0
$R_{21}^{(0)}$	ϕ
$R_{22}^{(0)}$	$0+1+\epsilon$

For $k=1$ From formula $R_{ij}^{(1)} = R_{ij}^{(0)} + R_{ik}^{(0)} \left(R_{kk}^{(0)}\right)^* R_{kj}^{(0)}$

$R_{11}^{(1)}$	$(\epsilon+1)+(\epsilon+1)(\epsilon+1)^*(\epsilon+1)$ $= 1^*$
$R_{12}^{(1)}$	$0+(\epsilon+1)(\epsilon+1)^*.0$ $= 1^*0$
$R_{21}^{(1)}$	$\phi+\phi.(\epsilon+1)^*.(\epsilon+1)$ $=\phi$
$R_{22}^{(1)}$	$(\epsilon+0+1)+\phi.(\epsilon+1)^*.0$ $=(\epsilon+0+1)+\phi=(\epsilon+0+1)$

Note:

কোনো রিজুলার সাথে ϕ গুণ হলে Result হবে ϕ.

like,

$$R.\phi = \phi$$

But,

$$R+\phi = R$$

Example - 3.5 (HW) → (MUST)

For $k=2$,

$$R_{ij}^{(2)} = R_{ij}^{(1)} + R_{ik}^{(1)} \left(R_{kk}^{(1)}\right)^{*} \cdot R_{kj}^{(1)}$$

$R_{11}^{(2)}$	$1^{*} + (1^{*}0) \cdot (\epsilon+0+1)^{*} \cdot \phi = 1^{*}$
$R_{12}^{(2)}$	$1^{*}0 + (1^{*}0) \cdot (\epsilon+0+1)^{*} \cdot (\epsilon+0+1)$
$R_{21}^{(2)}$	$\phi + (\epsilon+0+1) \cdot (\epsilon+0+1)^{*} \cdot \phi = \phi$
$R_{22}^{(2)}$	$(\epsilon+0+1) + (\epsilon+0+1) \cdot (\epsilon+0+1)^{*} \cdot (\epsilon+0+1)$

$R_{11}^{(2)}$ → $i=1, j=1$

$i=1, k=2$

$k=2, k=2$

$k=2, j=1$

$R_{21}^{(2)}$ → $i=2, j=1$

$i=2, k=2$

$k=2, k=2$

$k=2, j=1$

$R_{12}^{(2)}$ → $i=1, j=2$

$i=1, k=2$

$k=2, k=2$

$k=2, j=2$

$R_{22}^{(2)}$ → $i=2, j=2$

$i=2, k=2$

$k=2, k=2$

$k=2, j=2$

Shakhar (VM) — LAB-03 — 25/11/18

e, 1, 0

1, 0

1, 0, 11	e,
L = 10, 11, 100	
M = 11, 100	
L: M = 1011, 10100, 11111	11100, 10011, 100100

Enter elements of L:

e

1

0

110

-1 → break

Union
* Concatenation
* Closure

Enter elements of M:

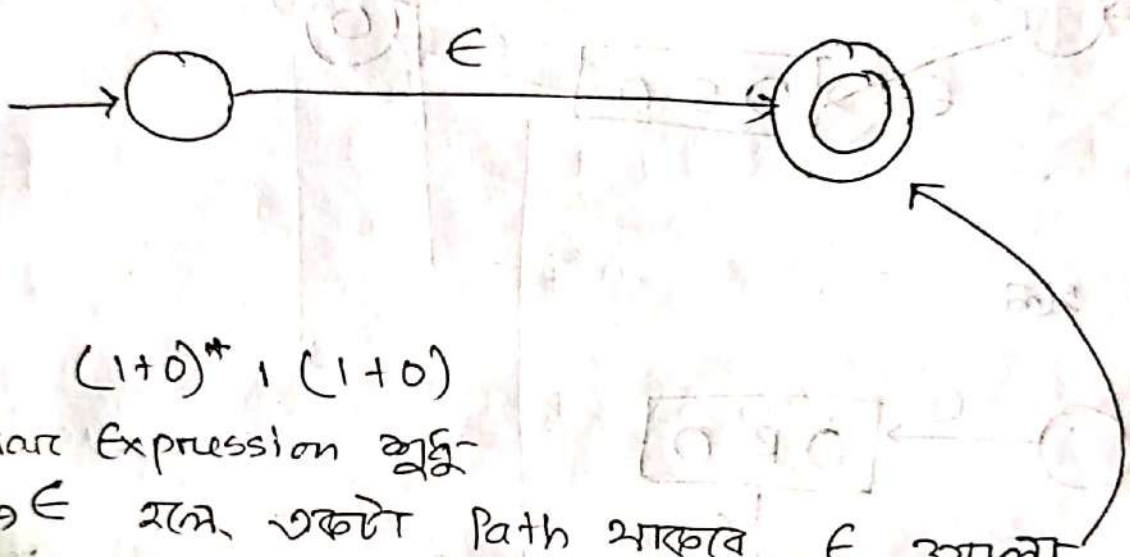
-1 → break

Makhan Sir

'E' Day
Cycle-06

27/11/18

Regular Expression to ∈-NFA:



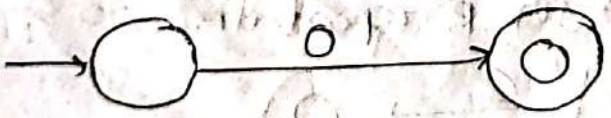
$(1+0)^* 1 (1+0)$

Regular Expression মুছে-

↳ ∈ হলে একটা Path থাকবে ∈ ওয়ালা

ϕ হলে কোনো Path থাকবে না.

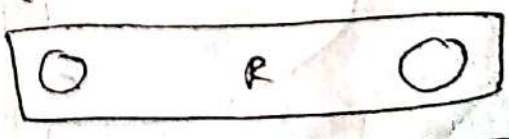
0 হলে-

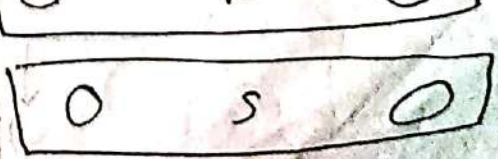


↳ দুটো symbol R ও S আছে -

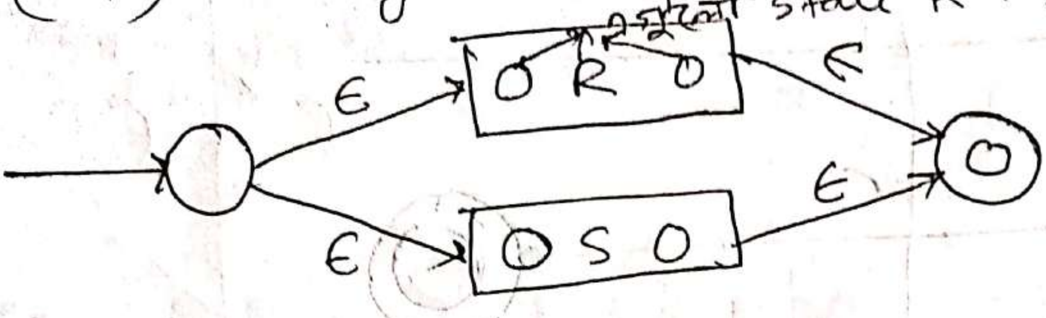
$R = (0+1)^*$

$S = 1^*$

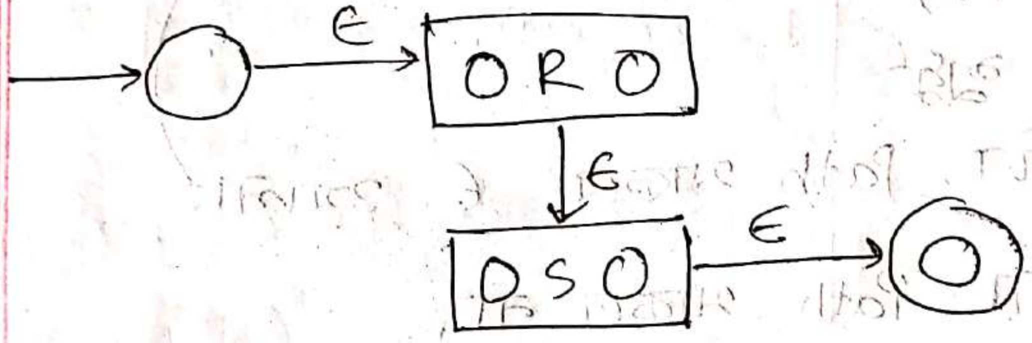




(R+S) যদি Regular expression হয়-

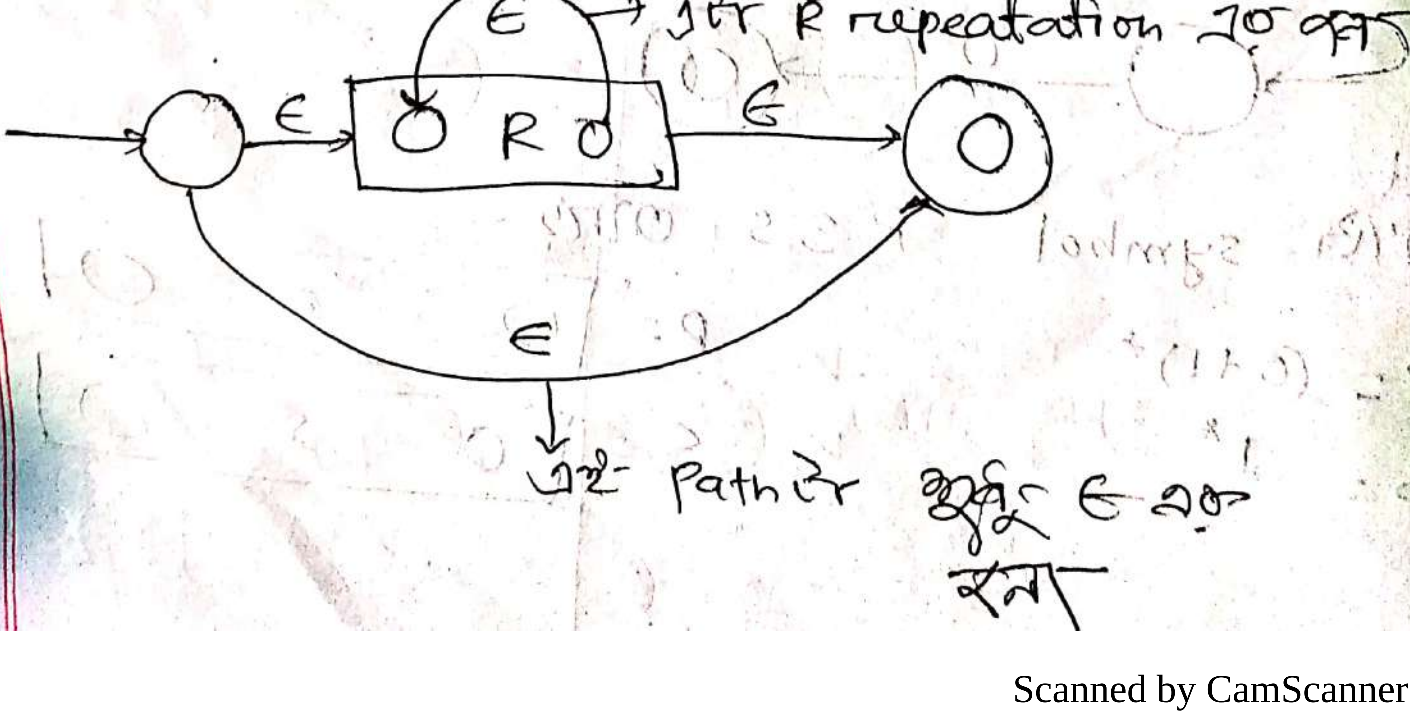


(RS) হলে

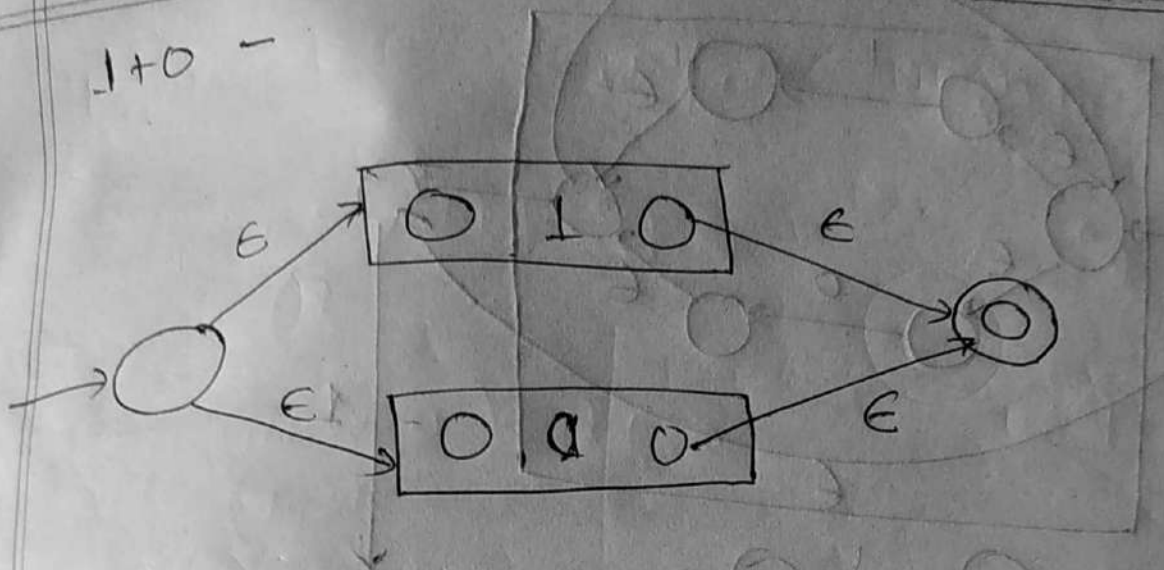


R^* হলে -

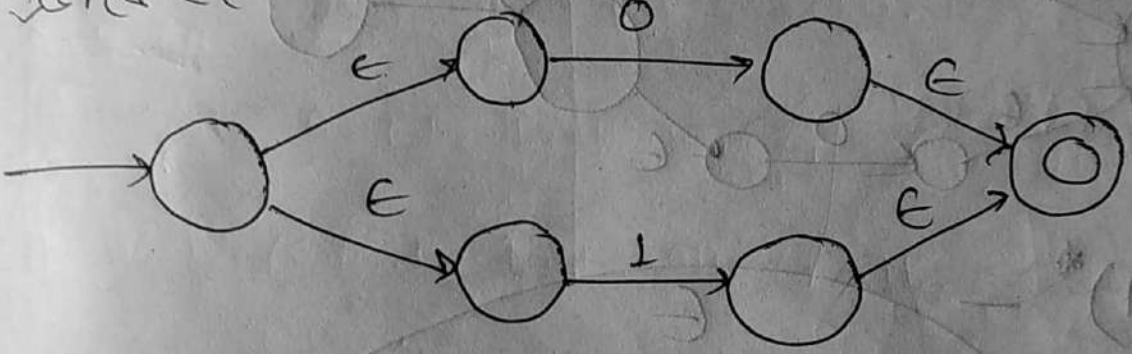
$R^* = \{ \epsilon, R, RR, RRR, \ldots \}$



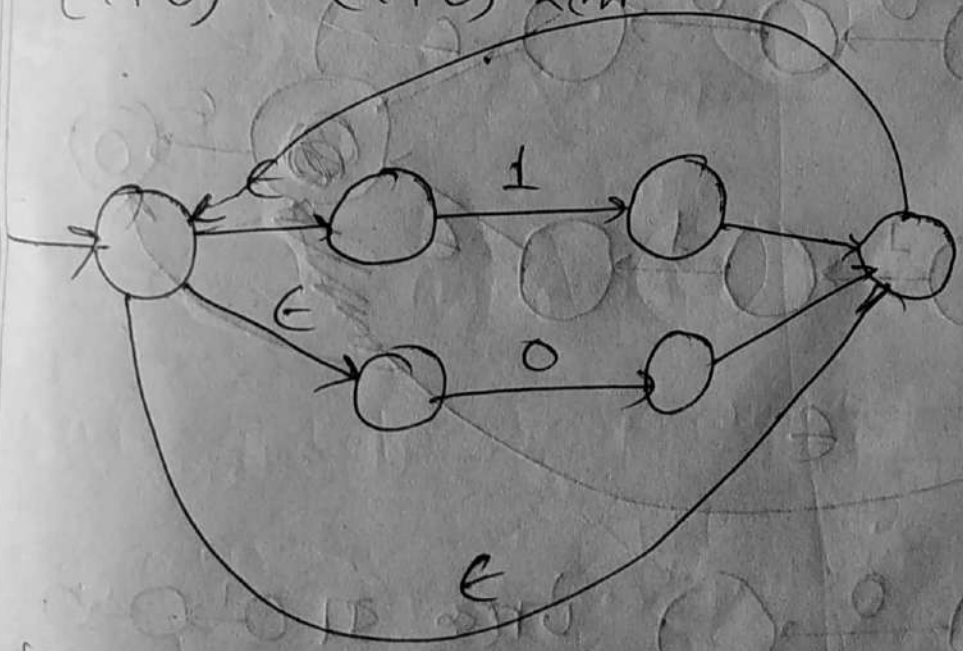
1+0 –



এভাবে হবে –

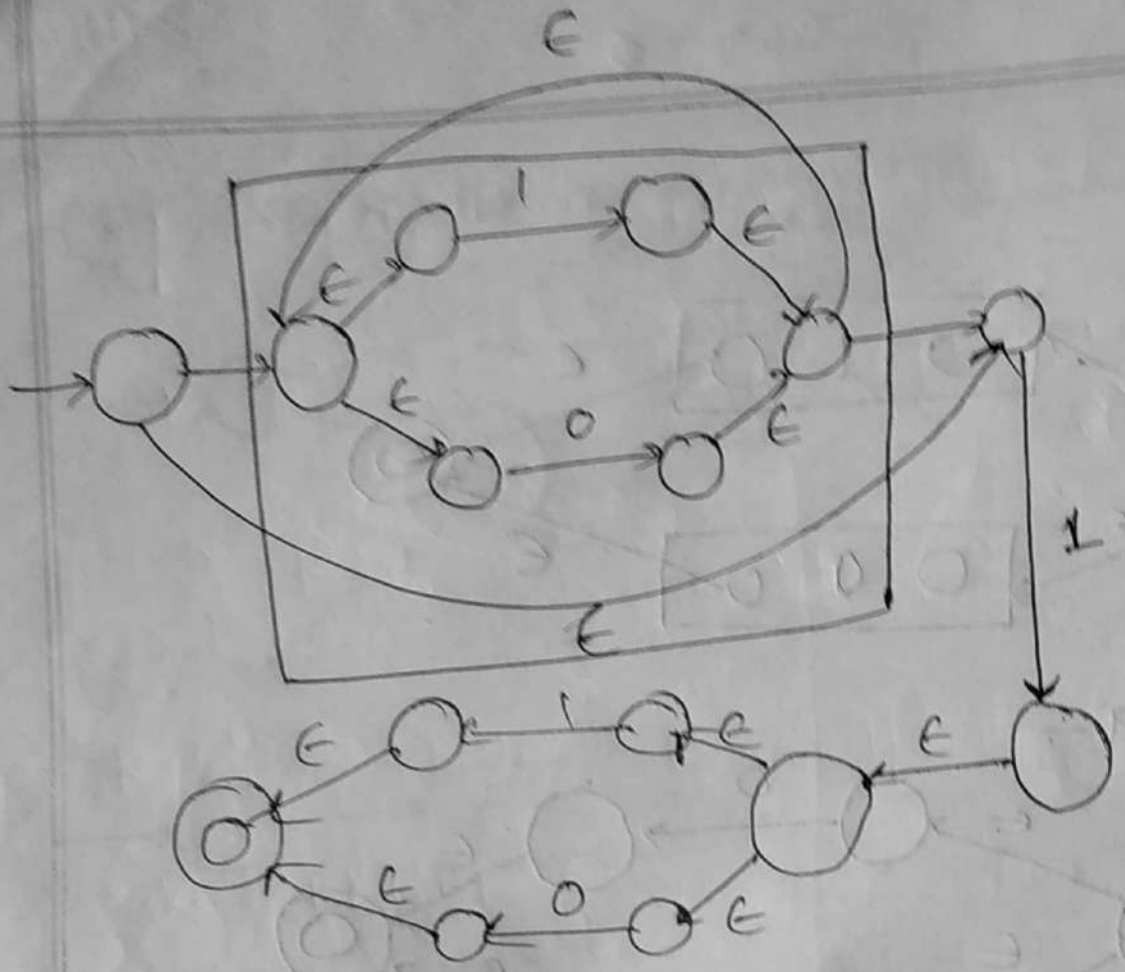


$(1+0)^*$ 1 $(1+0)$ হবে

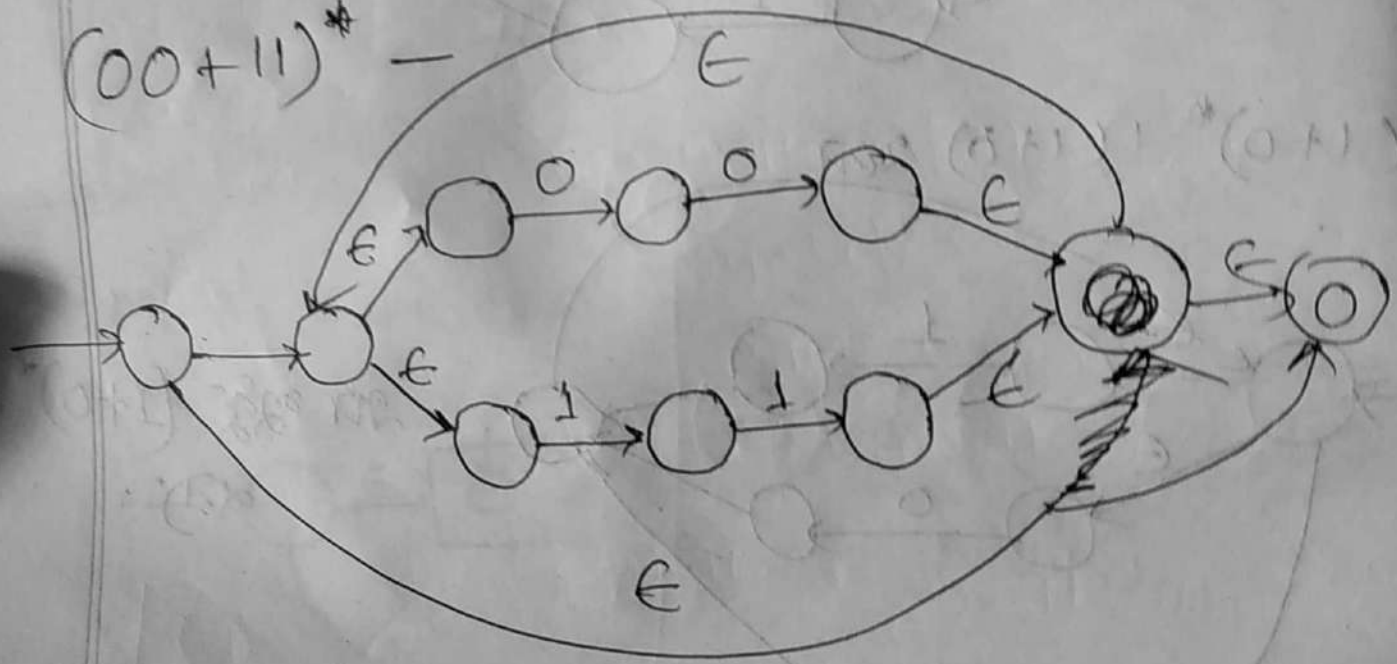


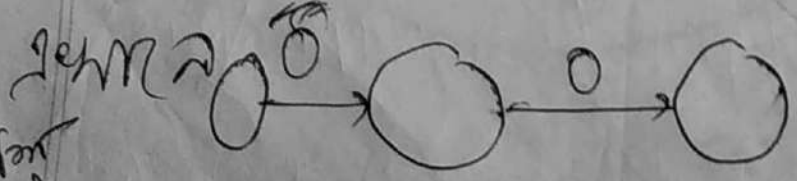
এবার আমরা $(1+0)^*$ এর পর.

এভাবে Full.

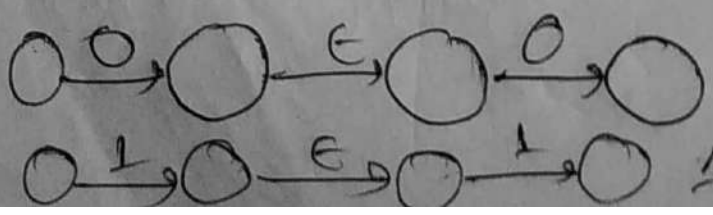


$(00+11)^*$ —





টাকে বা $0 \to 0 \to 0$



এভাবেও বা

এভাবেও (এখন যাবে,

'B' Day
Cycle-07

8/12/18

1. Union
2. Concatenation
3. Enclosure

(•) → dot
→ any operator

$a_1 + a_2 + a_3 + \ldots + a_n$

$[a_1 a_2 a_3 \ldots a_n]$ ← এটা Unix এ

$[a_1 - a_n]$ → এভাবেও করা যায় like $[a-z]$

Example: এটা Variable declaration valid হয়েছে কিনা

$[\_ A-Z\ a-z][\_ A-Z\ a-z\ 0-9]^*$ দেখার জন্য

Lexical Analyzer

At unix o.s.

$[0-9]$ → $[:digit:]$

$[A-Z a-z]$ → $[:alpha:]$

| $a_1 + a_2$ $a_1 | a_2$

+ R^+ RR^* (min. 1 বার থাকবে R)

? $R^?$ $\in + R$ (0 অথবা 1 বার থাকবে)

$R\{5\}$ মানে হচ্ছে R, 5 বার আছে.

like - RRRRR

a. c → a b c → মাঝখানে যে কোনো Symbol/Letter থাকতে পারে

a * c → মাঝখানে যে কোনো Symbol/Letter যে কোনো সংখ্যকবার থাকতে পারে

CT-2 Syllabus - Chapter - 3 এর Article 3.3 (ক) পর্যন্ত,

akhare
Sir

'E' Day

11.12.18

Cycle-07

Chap-3 (CT-02 syllabus)

Ant 3.1.1
3.1.2 → Proof
Example - 3.2
3.1.3
3.2.1

Example 3.5
Ex 3.6
3.2.3
3.3

Theorem 3.4 (যেকোনো একটা যেটা বইয়ে আছে আছে)

Associative & Commutative Law

1+2 = 2+1 ⟶ commutative

(1+2)+3 = 1+(2+3) ⟶ Associative

For RE,

→ L+M = M+L

→ (L+M)+N = L+(M+N)

} For uniqn

(LM)N = L(MN) ⟶ Concatenation এর ক্ষেত্রে Associative law
এর Commutative law নাই

Identity: In math $5 \times 1 = 5$ Like this in RE—

$L \cdot \epsilon = L$

Annihilators

$L \odot M = M$

যেকোনো Operator হতে পারে, তাহলে M হবে Annihilator

$L \cdot \phi = \phi$ → এখানে ϕ Annihilator

(যা দিয়ে গুন করলেও বা Operate করলেও শেষ পর্যন্ত শূন্য থাকে সবসময়)

Distributive Law:

$L(M+N) = LM + LN$

$(L+M)N = LN + MN$

$= MN + LN$ [applying Commutative]

$(L^*)^* = L^*$

$L^+ = L \cdot L^*$ (২টা বা তার বেশি)

$L? = \epsilon + L$

$\phi^* = \epsilon$

$\epsilon^* = \epsilon$

$(L+M)^* = \{\{\quad\} \cup \{\}\}^*$

$= (L^* M^*)^*$ [Derive করে দাব এটা]

$(1+0)^* 1 = (1^* 0^*)^* 1$ ⟶ Proof দিবে এই type দুইটা side equal কিনা এটা, এত সহজ দিবে না। আগে- কত Difficulties ছিল।

Day

cycle-08

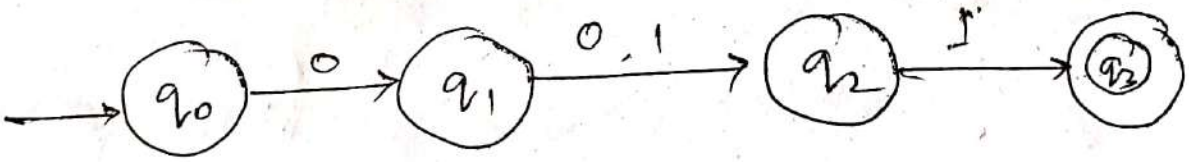
LAB-02

Input

1. No. of inputs
2. Transition table
3. Starting state
4. Final state

Output:

Regular Expression between start & final.



State	0	1
q_0	q_1	ϕ
q_1	q_2	q_2
q_2	ϕ	q_3
q_3	ϕ	ϕ

Khare Sir

'E' Day

Cycle-08

13.1.19

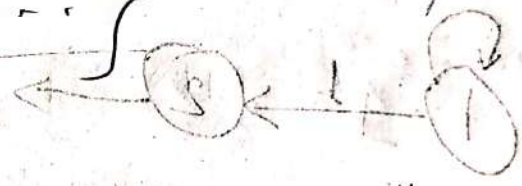
Pumping Lemma

ধরি, n state ওয়ালা একটা DFA আছে যেখানে

'n' state দ্বারা বিভিন্ন Pattern তৈরি হচ্ছে,

like -

$L = \{0, 01, 0101, \dots\}$



Let,

w একটা input comb. like $w = "0101110"$

যদি

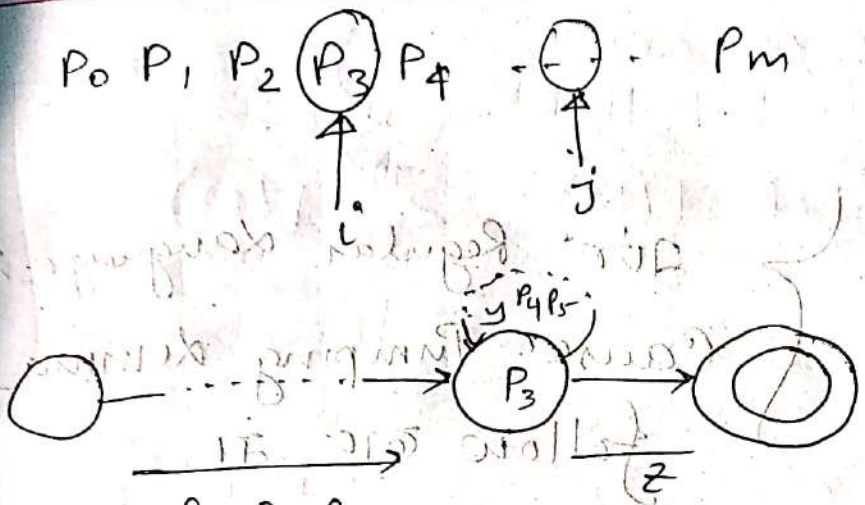
$|w| > n$ হয়

$0, 1, 2, 3, 4, \dots m$

$0 \le i < j \le m$

$P_0, P_1, P_2, \dots, P_m$ (0-m) state এর কম

$P_i = P_j$ হবে।



$|w| > n$ হবে

$w = xyz$ হিসেবে লিখতে পারি

Here,

① $y \neq \epsilon$

② $|xy| < n$

এখন xyz কে $xy^k z$ হিসেবে লেখা যাবে, Cause y এ একে অধিকবার ঘুরে আসলেও y ই পাওয়া যাবে, এখানে $K \geq 0$

So, কোনো DFA যদি এই Property (এগুলো যা লিখলাম) follow করে তাহলে ঐ language কে Regular Language বলা যাবে নতুবা না।

$0^n 1^n$

01
0011
000111
00001111

এটা Regular Language না,
Cause Pumping Lemma
follow করে না,

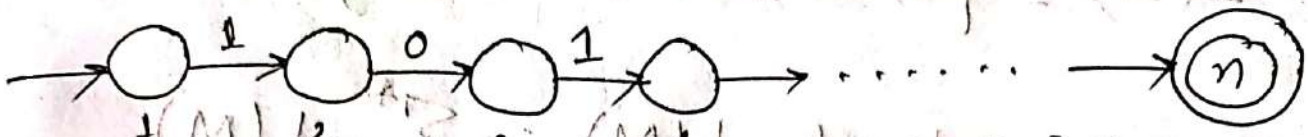
কেন করে না এটা explain করে
দেখাতে হবে,

বইয়ে দুইটা example আছে,

4.2, 4.3

'A' Day
Cycle-09

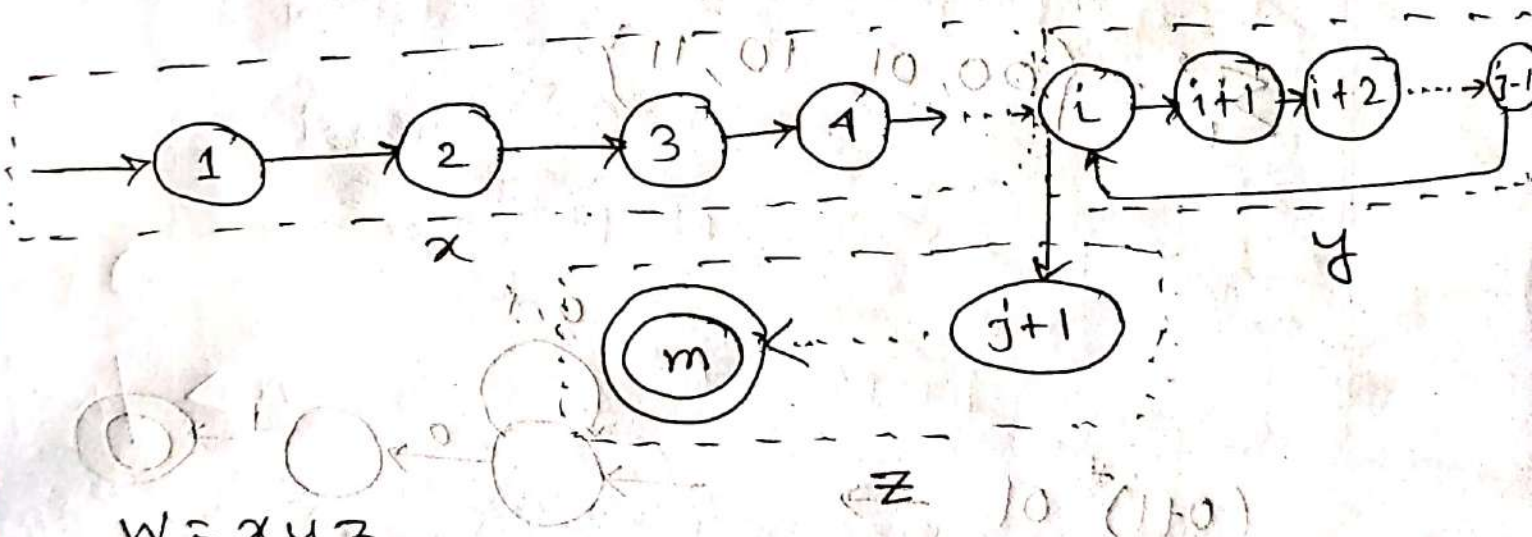
14.1.19



$|w| > n$

1 2 3 .... i i+1 .... j .... m

$|w| = m$



$w = xyz$

$|xy| \leq n$

$xy^k z$

$L(M) = \{0, 00, 000, \ldots\}$

Complement of $L(M) = \Sigma^* - L(M)$

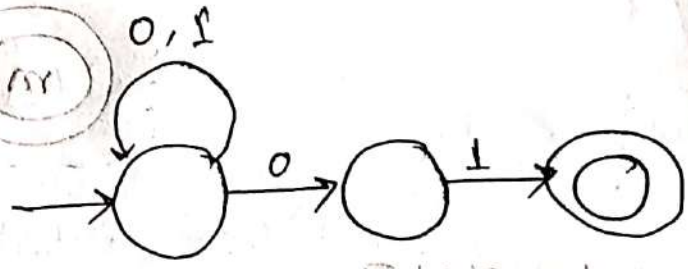
$\Sigma^0 = \epsilon$

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

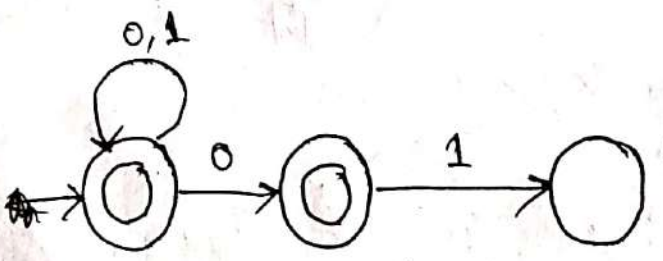
$\Sigma^1 = \{0, 1\}$

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

$(0+1)^* 01 \Rightarrow$



↳ यसको Complementary State



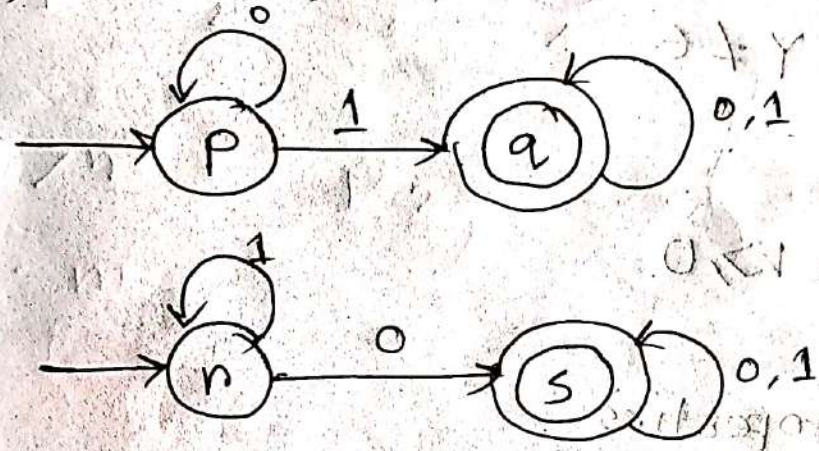
Regular Language ले Complement

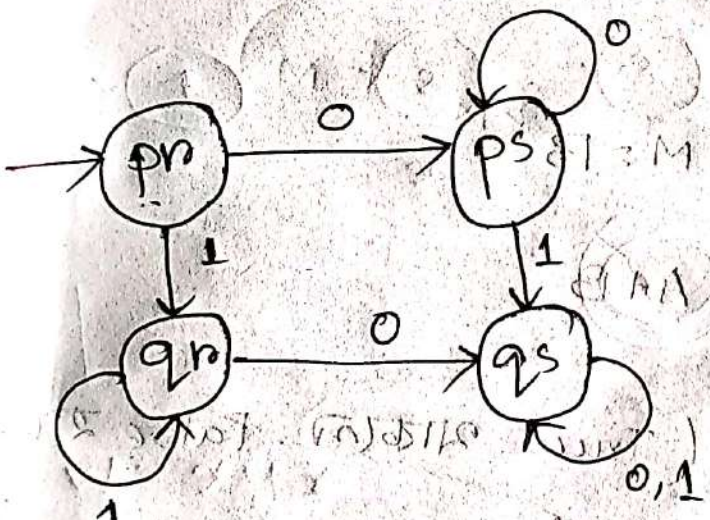
Regular Language हुन्छ।

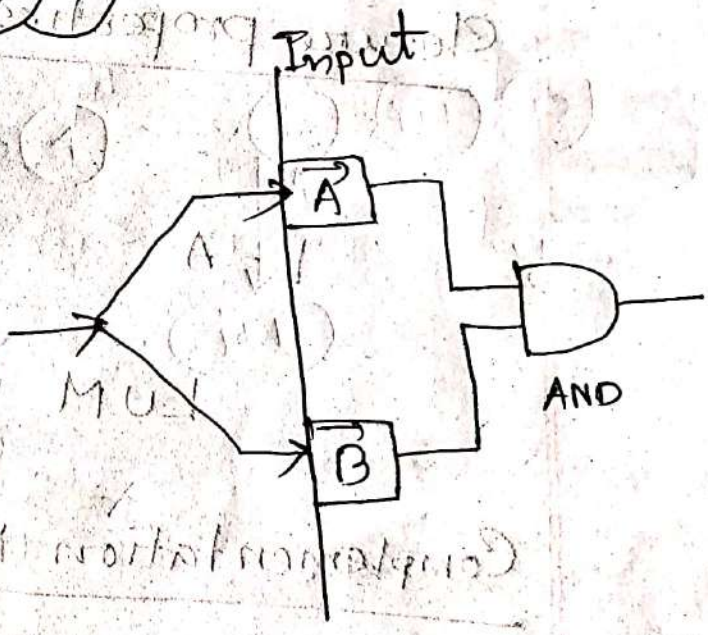
Intersection

$L(M) \cap L(N) = L(M+N)$

$L(M) \cap L(N)$

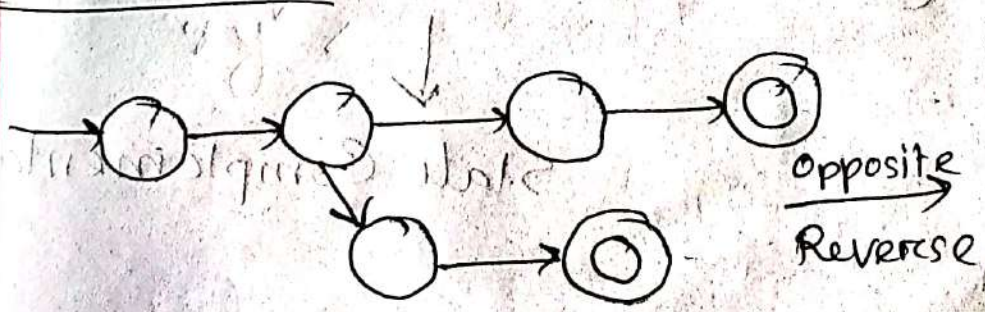


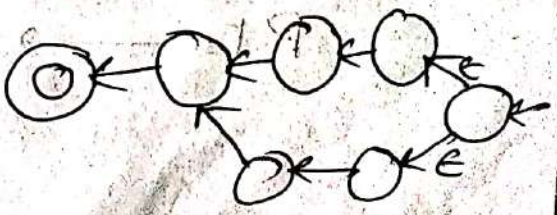




A intersection B

Reversal:





Shakhar sir

'B' Day

Cycle-09

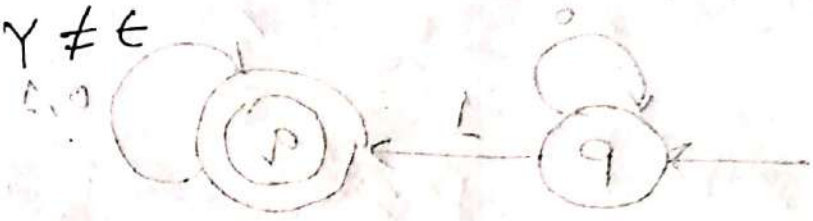
15.1.19

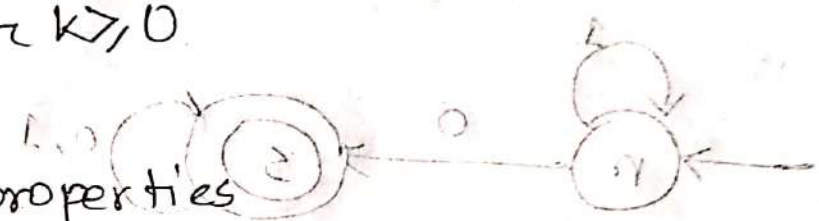
n

$w = xyz \quad y \neq \epsilon$

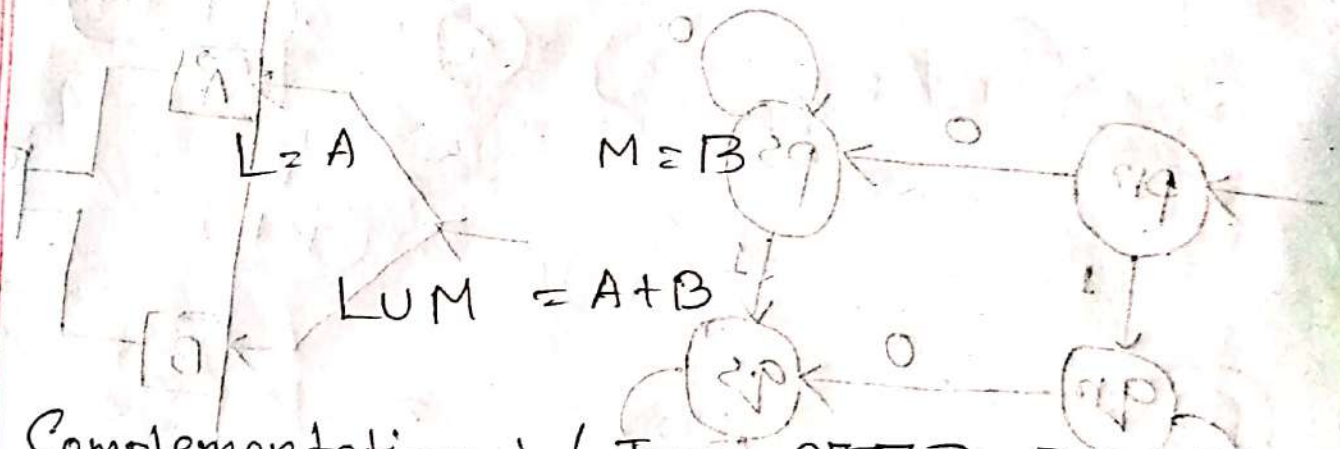
$|xy| \leq n$

$xy^k z$ for $k \geq 0$





Closure properties



Complementation : (True থাকলে False হবে, মোট কথা Opposite হবে)

RE ⟶ ϵ-NFA ⟶ DFA

↓

State Complementation

5-tuple

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

- Q → States
- Σ → Set of input symbols
- δ → Transition function
- q_0 → initial state
- F → set of Final States

Complement of A $\longrightarrow$ $A' = (Q, \Sigma, \delta, q_0, Q-F)$

↳ F ছাড়া বাকি states গুলো এখন Final States

Intersection:

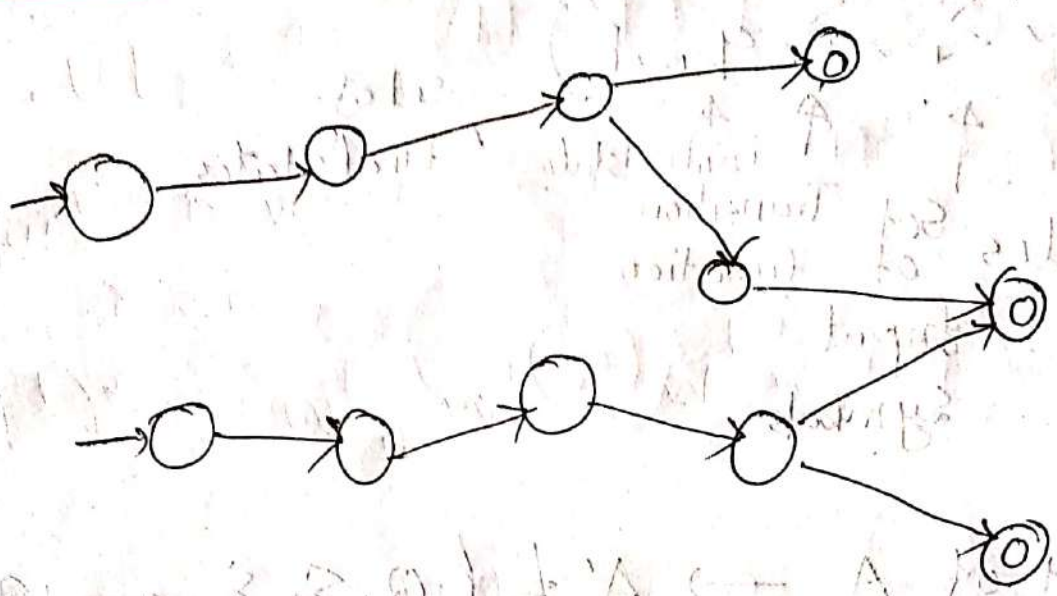
দুটো Regular language আছে L, M

এদের Intersection এ (মানে দুটোতে কমন) যে accepting states গুলো আছে accepted হবে।

$L = \{0, 01, 10, 11\}$ $M = \{1, 10, 11, 00\}$

$L \cup M = \{0, 1, 01, 10, 11, 00\}$

$L \cap M = \{10, 11\}$



$L = (Q_L, \Sigma_L, \delta_L, q_L, F_L)$

$M = (Q_M, \Sigma_M, \delta_M, q_M, F_M)$

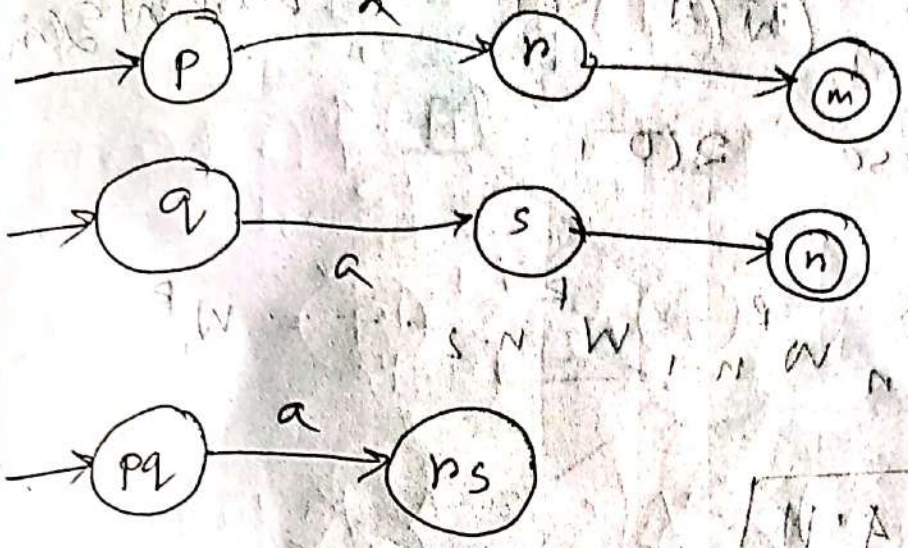
Let,

Σ_L and Σ_M same

$\Sigma = \Sigma_L = \Sigma_M$

Same না হলে,

$\Sigma = \Sigma_L \cup \Sigma_M$



$$L \cap M = (Q_L \times Q_M, \Sigma, \delta(\delta_L, \delta_M), q_L \times q_M, F_L \times F_M)$$

Reversal:

$L = \epsilon$ → Revert → $L^R = \epsilon$

$L = \phi$ → $L^R = \phi$

$L = a$ → $L^R = a$

Regular Language का Reverse भी

Basis Case →

Same करेंगे।

$E_1 = 10\phi1$

$E_2 = 1100$

$E = E_1 \cdot E_2$ → 10111100

Reverse करेंगे $E^R \Rightarrow E_2^R \cdot E_1^R$

$E_1^R = 1101$

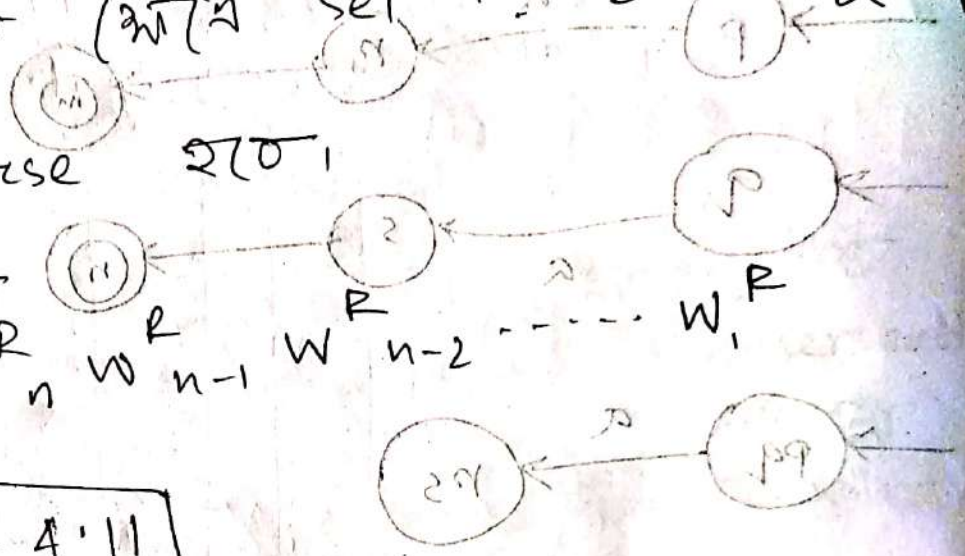
$E_2^R = 0011$

$= 0011 + 1101$

$= 00111101$

* $E_1 \cup E_2$ এর (সাধারণ set এর element হবে

Just Reverse হবে।



Closure এর ক্ষেত্রে —

$$(E^R)^* = W^R_n W^R_{n-1} W^R_{n-2} \cdots\cdots W^R_1$$

Theorem - 4.11

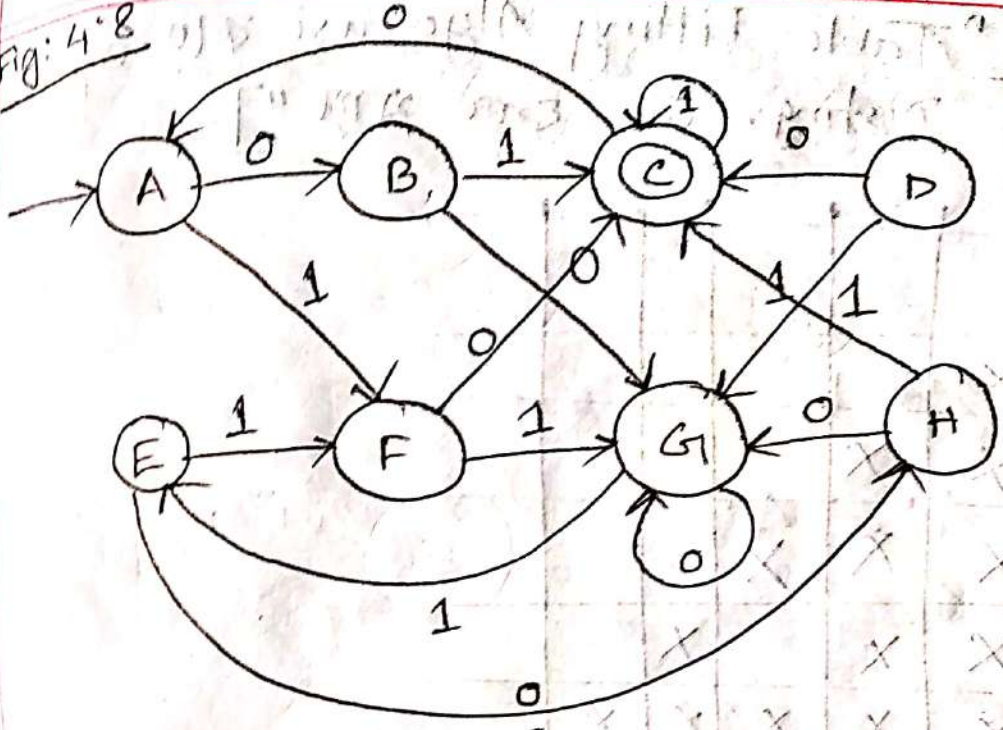
Shabbar Sir.

F/Day

Cycle-10

27.1.19

Fig: 4.8



(*) দুটো কিন্তু state, same input এ গেলে একটা non-accepting, accepting state এ যাবে তাহলে state দুটো equivalent. Otherwise, distinguishable.

Example:

$\delta(A, 0) = B$
$\delta(G, 0) = G$
} So, equivalent

আবার,

$\delta(A, 01) = C$
$\delta(G, 01) = E$
} distinguishable as C is accepting state

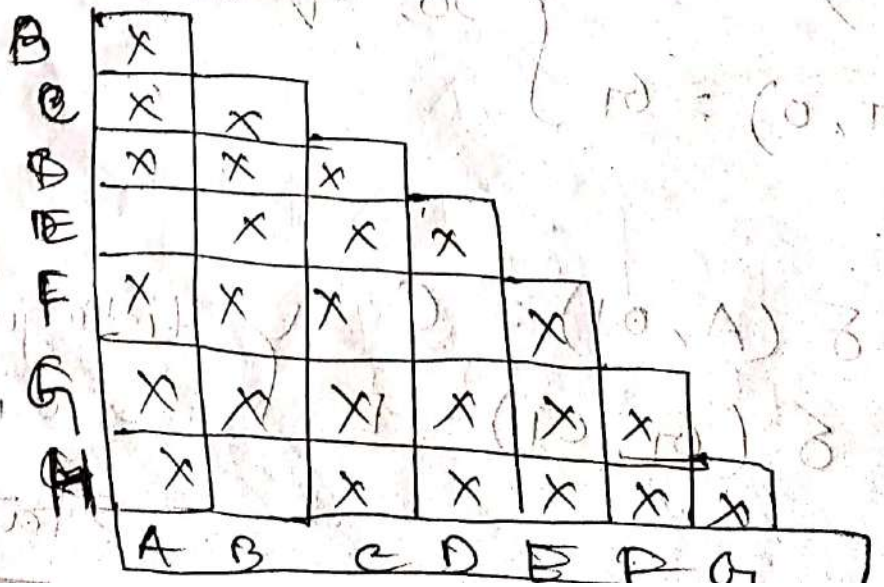
***এর exam এ

["Table Filling Algo. use করে equiv. নাও
Disting. pair এর মধ্যে"]

B	X					
C	X	X				
D	X	X	X			
E		X	X	X		
F	X	X	X		X	
G	X	X	X	X	X	X
H	X	X	X	X	X	X
	A	B	C	D	E	F

(যেগুলো equiv. সেগুলো আলাদা থাকবে)

Basically box গুলা যে এমন:



Homomorphism:

1011001

h(0) = ab
h(1) = ϵ

* এখন 0 এলে ab দিয়ে Replace করো
এবং 1 এলে ϵ দিয়ে Replace করো। এটাই Homomorphism

এখন 1011001 কে homo. apply করো,

ababab

(এখানে Proof দেখার দরকার নাই। Just def, example)

* inverse homo. হলে h^{-1} দিলে এবার এখন ababab এলে 1011001 হবে এবার,

CT-03 (Syllabus)

Chap-4:

Article - 4.1.1 *
4.2, 4.3
4.2.1

(* এই পাঠ থাকলে 10 marks common)

Theorum - 4.4
4.6
4.7
4.8
4.9

4.2.2
4.11 *
4.2.3
4.2.4
4.4.1 *
4.19

Shakhar Sir

'B' Day

Cycle-11

29.1.19

Context Free Grammer:

$\Sigma = \{0, 1\}$ এ Pallindrome কি কি থাকে?

Base Case:

$L = \epsilon$ হলে $L = L^R$

So, format টা হবে হবে 0 W 0 or 1 W 1

Here, W be the pallindrome too

Start Symbol

1. $P \rightarrow \epsilon$
2. $P \rightarrow 0$
3. $P \rightarrow 1$
4. $P \rightarrow 0P0$
5. $P \rightarrow 1P1$

* এগুলো Rule দিয়ে define করা যাবে।

* এখন এটাকে Grammer বলে।

(*) C.F.G হল Recursive way তে define করা Language কে।

Context free Grammer এ কিছু Symbol থাকে যা দিয়ে language Build করা হয়। এই Symbol গুলো Terminal Symbol.

C.F.G এ যা যা থাকে -

1. Terminal Symbol → like {0, 1} (input symbol এর set) → T.S
2. Variables (OPO তে P কে Var.) Non terminal Symbol
Multiple Variable থাকতে পারে
3. Start Symbol (এটা থাকতে হবে First Variable এ)
4. Production / Rules

(উদাহরণ: $\frac{a^* (b+a)}{B \quad C}$ তে Mul. Var. নাও হতে পারে)

A → B * C

A → Start Symbol

Context Free Grammar:

Definition

Rule Criteria →

a) একটা symbol থাকবে define করতে চাইলে like-P

b) ⟶ (Arrow)

c) যে কোনো string এর আকারে Variable (যেমন P) কে define করবে,

So, Grammar:

$$G = (V, T, S, P)$$ → 4 tuple

- V → Var.
- T → Terminal Sym.
- S → Start Sym.
- P → Production/Rules

$(a+b)(a+b+0+1)^*$ এর Context free Grammar —

করার —
শুরুতে E নিলাম

1. $E \rightarrow I$ [I হলো যেকোন একটি a, b, 0, 1 এগুলোকে যে কোনো একটি দ্বারা Replace করতে পারে]
2. $E \rightarrow E+E$ [a+b এর জন্য]
3. $E \rightarrow E*E$ [$(a+b+0+1)^*$ এর জন্য]
4. $E \rightarrow (E)$ [Bracket এর জন্য → $(a+b)$, $(a+b+0+1)^*$]

5. $I \rightarrow a$

6. $I \rightarrow b$

7. $I \rightarrow I0$

8. $I \rightarrow I1$

9. $I \rightarrow Ia$

10. $I \rightarrow Ib$

(অথবা ab দ্বারা কিন্তু অন্য
না করে first এ (a+b)
আসবে)

[So, শুধু I→0 & I→1 হলে ন

*** Exam এ Rule (যেই
কিন্তু এদের derive করতে
বলবে।

Suppose b00 derive করতে বলবে →

$E \rightarrow I$

$\Rightarrow E \rightarrow I0$

$\Rightarrow E \rightarrow I00$

$\Rightarrow E \rightarrow b00$

→ उल्टा क्रम, → उल्टा क्रम

(*) a*(a+boo) derive करो।

V.V.I

String Inferred	For Language	Production/ Rule Used	String Used
a	I	5	
b	I	6	
bo	I	7	(ii)
boo	I	7	(iii)
I a	E	1	(i)
I boo	E	1	(iv)
a+boo	E	2	(v) (vi)
(a+boo)	E	4	(vii)
a*(a+boo)	E	3	(v) (viii)

2nd तरह E*(E+E) format में करो।

প্রোগ্রাম — (Left most derivation)

* Left side থেকে শুরু করে

a * (a+b00)

E = E * E ⇒ E = I * E

⇒ E = a * E

⇒ E = a * (E)

⇒ E = a * (E+E)

⇒ E = a * (I+E)

⇒ E = a * (a+E)

⇒ E = a * (a+I)

⇒ E = a * (a+I0)

⇒ E = a * (a+b00)

* Right side থেকে শুরু করে Right most derivation

(Right most derivation)

like -

$E = E A E$

$\Rightarrow E = E A (E)$

$\Rightarrow E = E A (E + E)$

$\Rightarrow E = E * (E + I)$

$\Rightarrow E = E * (E + I 0)$

⋮

এভাবে.

Shakhar sir

'E' Day

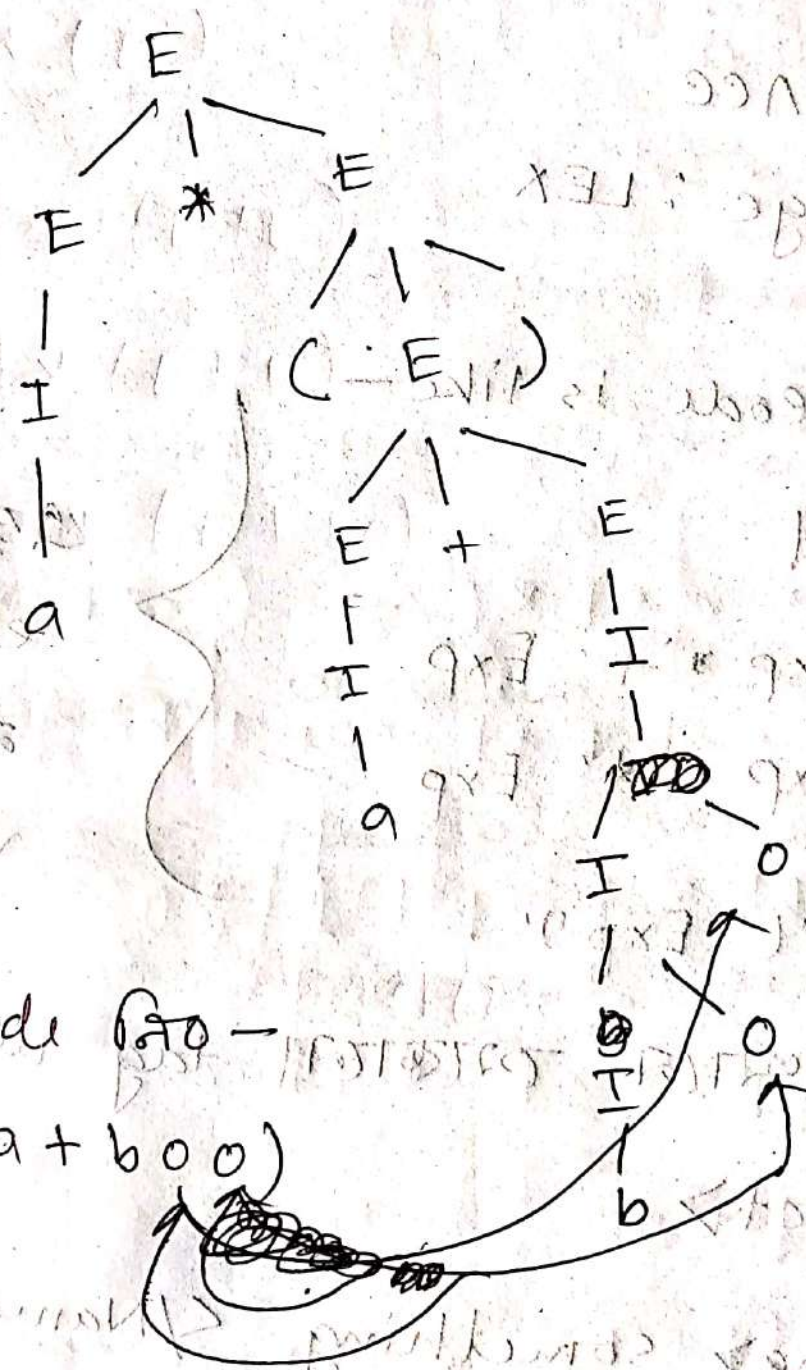
cycle-II

3/2/19

1. E → I
2. E → E+E
3. E → E*E
4. E → (E)
5. I → a
6. I → b
7. I → Ia
8. I → Ib
9. I → I0
10. I → I1

Parse Tree:

a*(a+boo)



leaf node are -

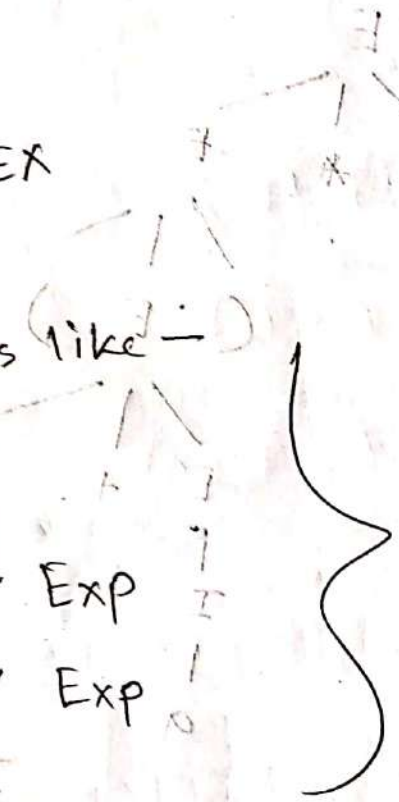
a * (a + b o o)

Application of C.F.G:

→ Parser in Compiler Designing

Tool: YACC

Language : LEX



Lex এ code is like—

Exp: Id

Exp '+' Exp

Exp '*' Exp

'(' Exp ')'

এগুলো exam এ বা 2nd, 3rd year এ থাকবে.

Extensible Markup Language — XML (এখানে যেকোনো tag use করা যায়)

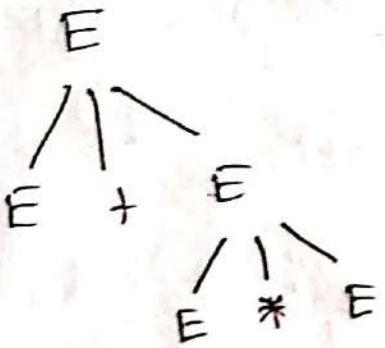
<contact>

<Name> something </Name>

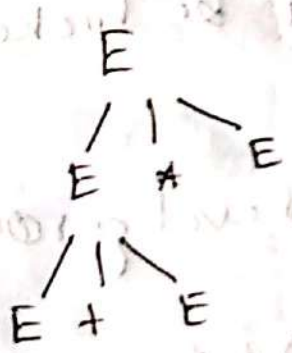
<Phone> . . . . </Phone>

</contact>

1+2*3

1. 

(E ⇒ E+E ⇒ E +E*E)

2. 

(E ⇒ E*E ⇒ E+E*E)

(*) এটা Ambiguous Grammer. যখন একটা str Grammer এর জন্য multiple Derivation পাওয়া যায়, তখন তাকে Ambiguous Grammer বলে

Shakhar sir

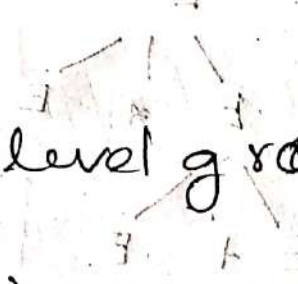
A' Day
Cycle-12

4.2.19

Reason of Ambiguousity:

1) Precedences
2) Wrongly grouping

Factor → Id
Factor → Parenthesis
} Can not be broken



Term → String with same level grouping

like → a*E*E → (a*E)*E

but → a+E*E can't be grouped

I → a | b | Ia | Ib | I1 | I0

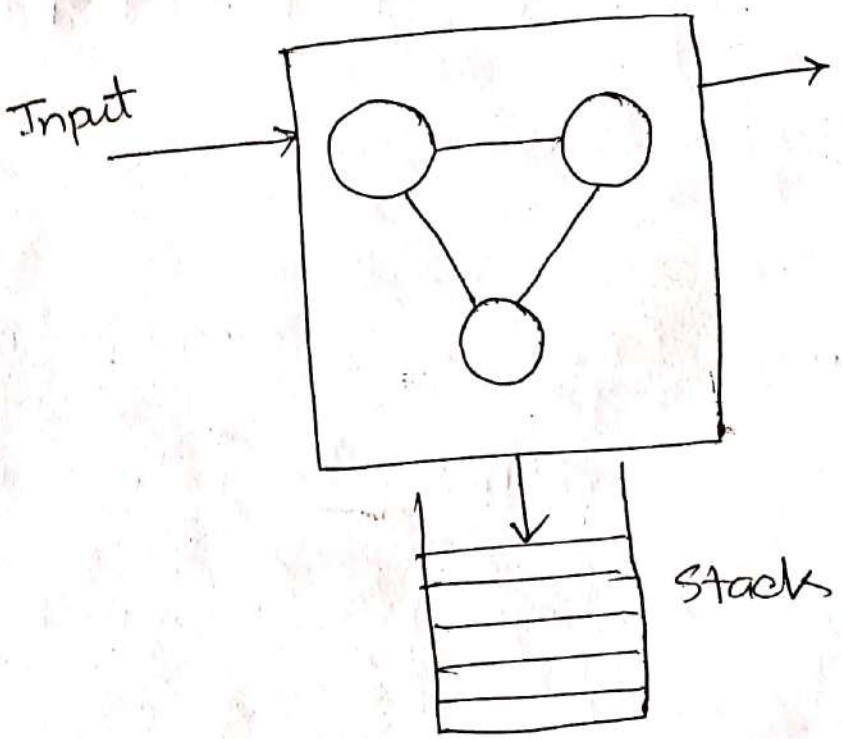
F → I | (E) ; F = Factor

T → F | T*F ; T = Term

E → T | E+T , E = Expression

Push Down Automata (PDA):

(*) ϵ-NFA টাই PDA. এখানে Just একটা stack add করবে, ফলে আগে কী, কোথায় Traverse করবে সব মনে রাখতে পারে.



⇒ $(Q, \Sigma, \delta, q_0, F)$ আগে ছিল

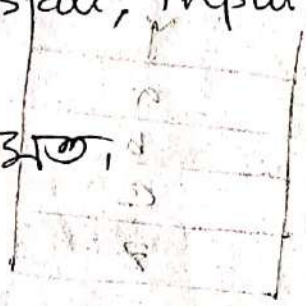
নতুনটা হবে PDA = $(Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$ ← starting symbol

এই δ তে change আসবে ↙ stack এ যেসব symbol থাকতে পারে তাদের set

* δ আগে ছিল (current state, input)

এখন δ = (current state, input, stack)

* আর এর আগের মত,



δ(q, a, X)

q = current state

a = input symbol

X = stack

→ এটা Return করে (P, γ)

↳ এটা stack এর জন্য

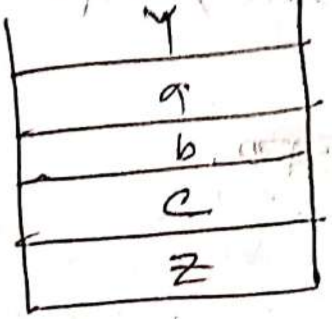
γ = X হলে stack এর value change করবে না।

γ = ∈ হলে Pop করে

γ = YZ থাকে তাহলে Z টাকে X এর আগে Replace করে, এবং stack এ X এর জায়গায় Z আসবে এবং এর top এ থাকবে Y.

Like,

$Yt = Yabez$ হলো Stack এ থাকবে এরকম



Shankar Sir

'B' Day

Cycle-12

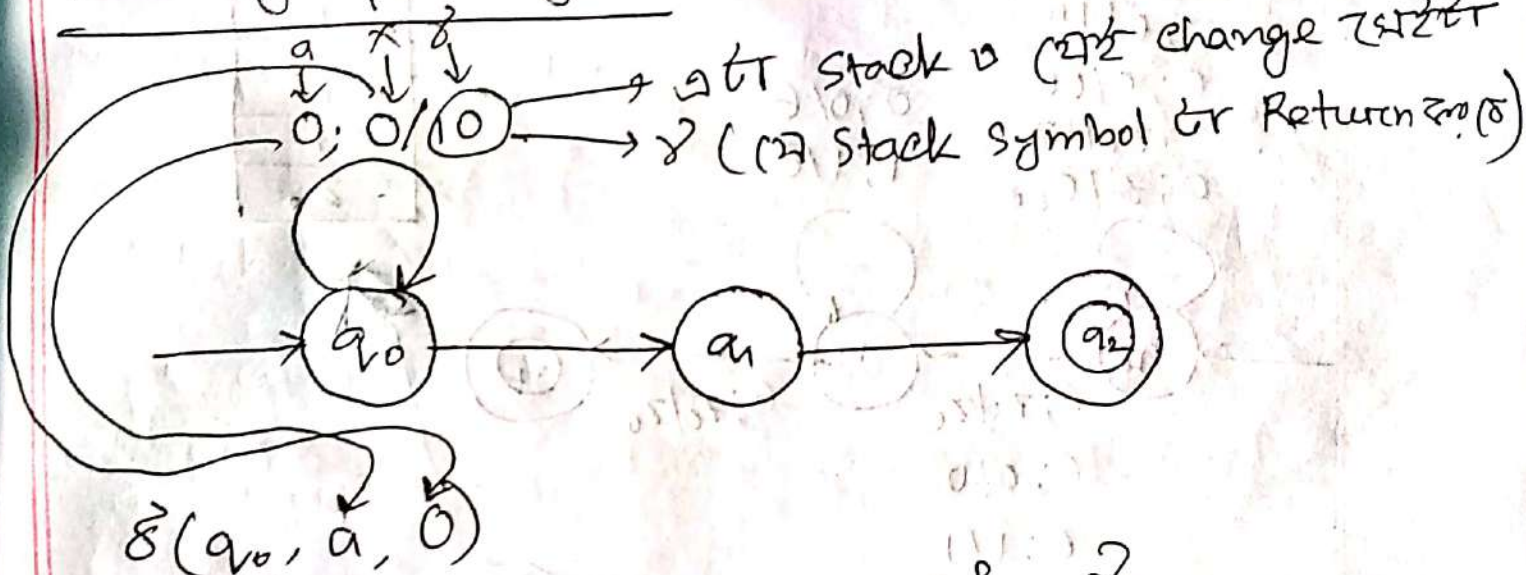
5.2.19

stack symbol

$(Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$

Stack's Start Symbol

PDA graphically:



stack এর change হয়েছে

γ (নতুন stack symbol বা Return হচ্ছে δ)

$\delta(q_0, a, 0)$

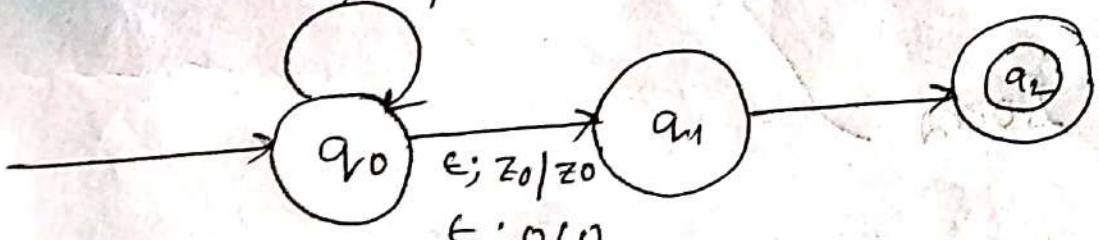
$\delta(q_0, 0, Z0) \longrightarrow (q_0, 0Z0)$

p γ

উদাহরণ

1; 0/10
0; 0/00 (একদম stack এ নতুন element যুক্ত হচ্ছে)
1; Z0/1Z0
0; Z0/0Z0

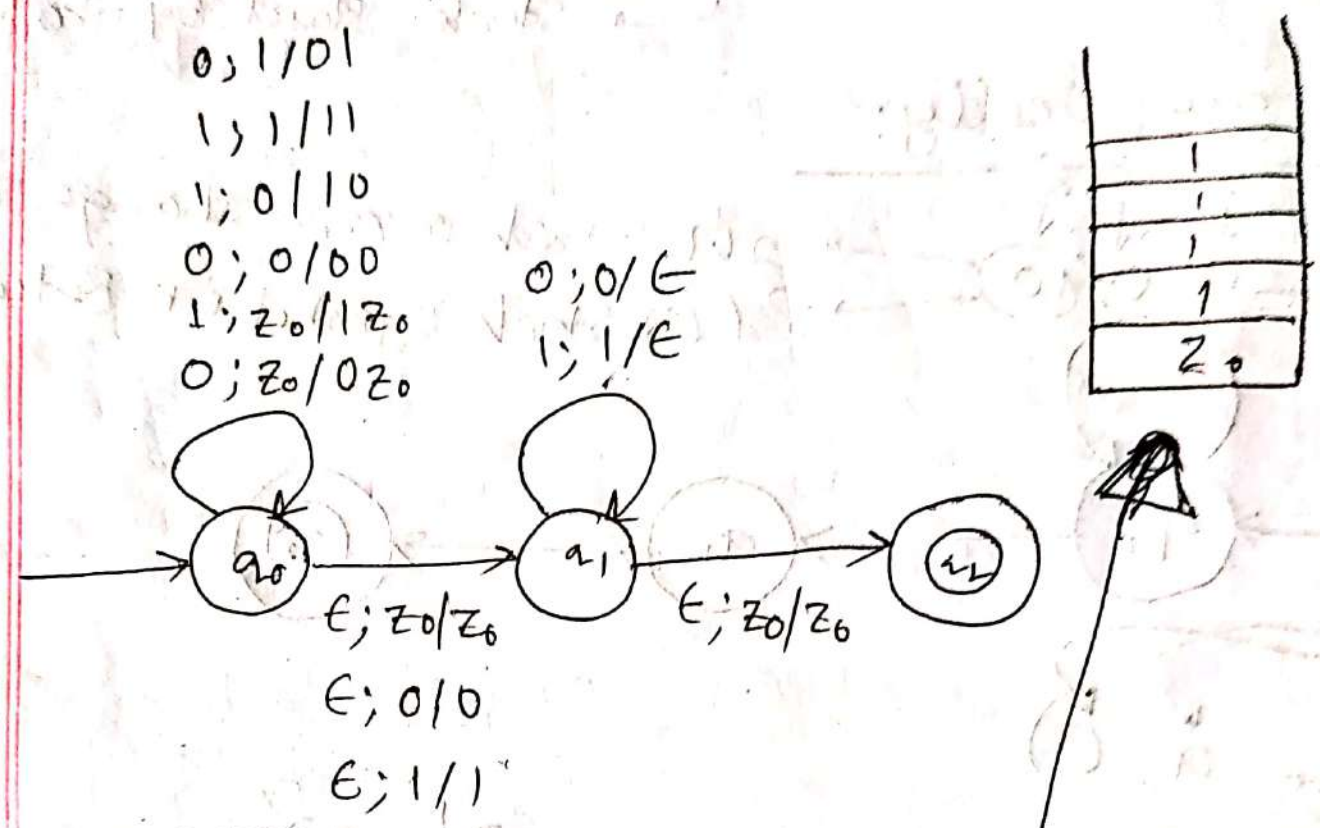
$\delta(q_0, 1, Z0) \longrightarrow (q_0, 1Z0)$



ϵ; 0/0

ϵ; 1/1

Fully draw করতে —

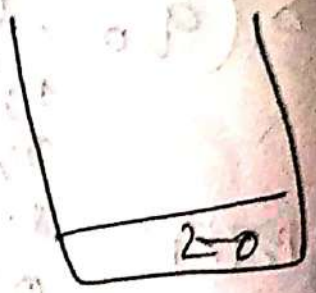


1111 এর জন্য

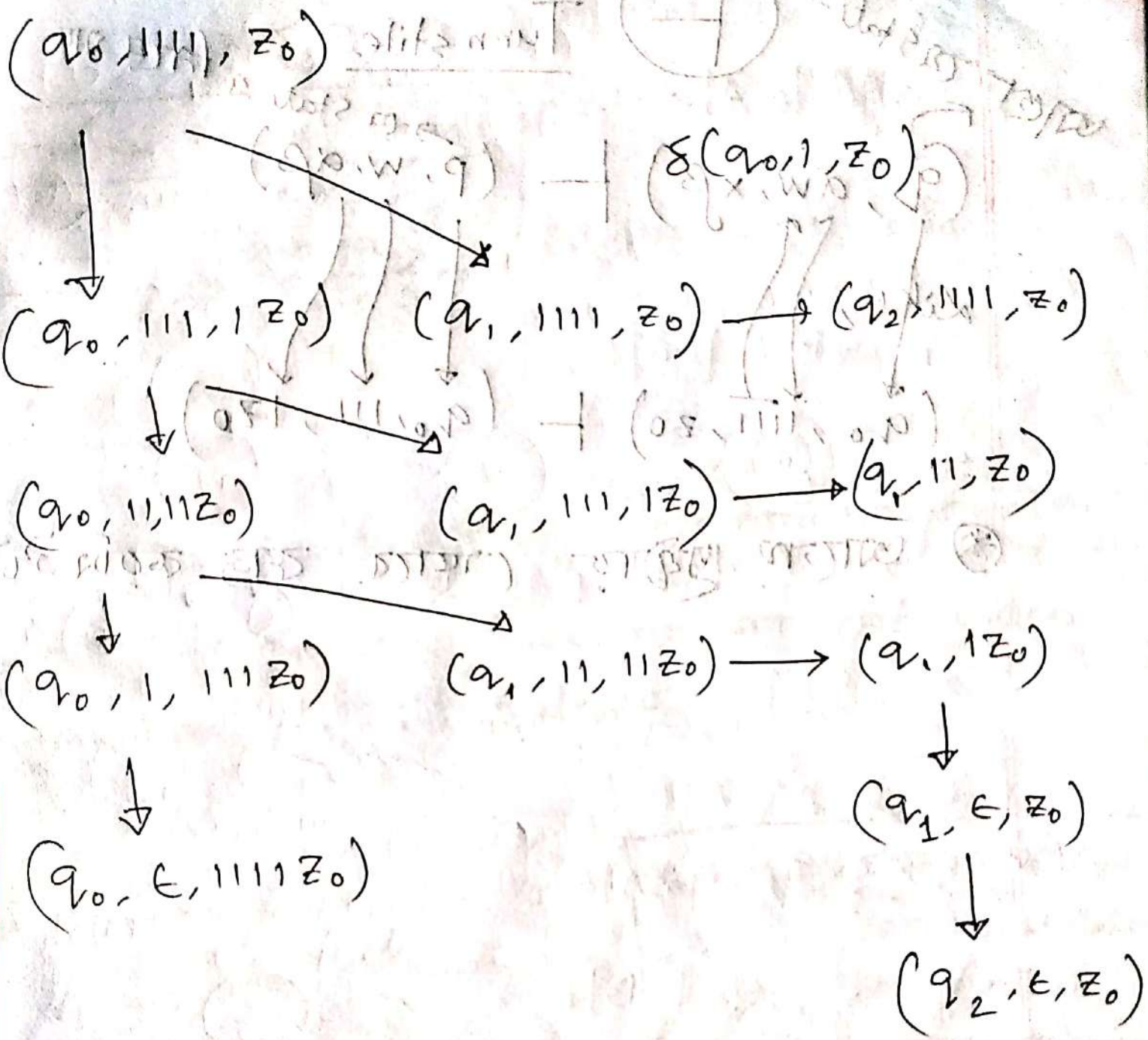
* এক্ষেত্রে যখন q1 এবং যখন q0 এ থাকে

যখনই stack খালি

* সব q0 এ যখনই



যাবে



Turn stile

$(q, aw, x\beta) \vdash (p, w, \alpha\beta)$

$(q_0, 1111, z0) \vdash (q_0, 111, 1z0)$

* আগের পুরো প্রসেস কতটি স্টেপে

Shakhar Sir

E Day

11.2.19

Cycle-12

(V, T, P, S)

Chomsky Normal form

$A \to BC$ (variable)

$A \to a$ (terminal)

$S \to AB \mid a$ ⟶ এটা

$A \to b$

$S \to a$

$A \to b$ ⟶ এটা বাদ দিয়ে দিতে পারি কেননা useless input s এ।

* যে Symbol কখনো use হবে না সেই useless symbol কে বাদ দিয়ে পড়িতে পারি।

$S \to AB$

$A \to aAA \mid \in$

$B \to bBB \mid \in$

$A \to \in$ এরকম থাকলে ওকে nullable, তখন এটাকে যেকোনো সময় বাদ দিতে পারে

যেমন $A \to aAA$ থেকে $A \to aA$ লিখতে পারে, $A \to a$

So,

$A \rightarrow aAA \mid aA \mid a$

এখন B কে বসালে

$B \rightarrow bBB \mid bB \mid b$

So, S কে লিখলে

$S \rightarrow AB \mid A \mid B$

So, System টা নতুন করে লিখলে-

$S \rightarrow AB \mid A \mid B$

$A \rightarrow aAA \mid aA \mid a$

$B \rightarrow bBB \mid bB \mid b$

$I \rightarrow a \mid b \mid Ia \mid Ib \mid I1 \mid I0$

$F \rightarrow I \mid (E)$

$T \rightarrow F \mid T*F$

$E \rightarrow T \mid E+T$

Unit Production: একটা Variable আরেক একটা Single Variable কে Direct করলে. like - $E \rightarrow E$, $E \rightarrow F$

(E, E) একাধিক থাকা যায়।

এখানে-

(E, E) $E \rightarrow T$ (E, T)

(E, T) $T \rightarrow F$ (E, F)

(E, F) $F \rightarrow I$ (E, I)

(E, I)

(T, T) $T \rightarrow F$ (T, F)

(T, F) $F \rightarrow I$ (T, I)

(F, F) $F \rightarrow I$ (F, I)

তাহলে কাজ হল এই Unit Production গুলো Remove করা,

$I \rightarrow a \mid b \mid Ia \mid Ib \mid I0 \mid I1$

$F \rightarrow I \mid (E)$

$I \rightarrow F \mid T * F$

$E \rightarrow T \mid E + T$

Removal of U.P:

Pair	U.P eliminate rule
(E, E)	E → E+T
(E, T)	E → T*F
(E, F)	E → (E)
(E, I)	E → a | b | Ia | Ib | I0 | I1

So,

$E \rightarrow E+T \mid T*F \mid (E) \mid a \mid b \mid Ia \mid Ib \mid I0 \mid I1$ — (1)

Next table

Pair	
(T, T)	T → T*F
(T, F)	T → (E)
(T, I)	T → a | b | Ia | Ib | I0 | I1

$T \rightarrow T*F \mid (E) \mid a \mid b \mid Ia \mid Ib \mid I0 \mid I1$ — (2)

Same way (T, (F, F) এবং (F, I) করে T,

তারপর আমরা,

$F \rightarrow (E) \mid a \mid b \mid Ia \mid Ib \mid I0 \mid I1$ ——— (III)

$I \rightarrow a \mid b \mid Ia \mid Ib \mid I0 \mid I1$ ——— (IV)

তাহলে, Given Grammar (যার কোন U.P Removed

নয়,

Steps:

(1) Remove Useless Symbol

(2) " Epsilon

(3) " Unit Production

প্রথমে Terminal Symbol : a, b, 0, 1, +, (,), *

Variable : E, F, T, I

$A \to a$

$B \to b$

$C \to 0$

$D \to 1$

$G \to ($

$H \to)$ — E, T, F, I use হবে না

$J \to +$ — সেগুলো এগুলোর জায়গায়

$K \to *$ — Variable আছে

এখন (i), (ii), (iii), (iv) কে নিলাম

একটা থাকলে Replace হলেও হবে না।

$E \to EJT \mid TKF \mid GEH \mid a \mid b \mid IA \mid IB \mid IC \mid ID$

$T \to TKF \mid GEH \mid a \mid b \mid IA \mid IB \mid IC \mid ID$

$F \to GEH \mid a \mid b \mid IA \mid IB \mid IC \mid ID$

$I \to a \mid b \mid IA \mid IB \mid IC \mid ID$

Shankhari Sir

'A' Day

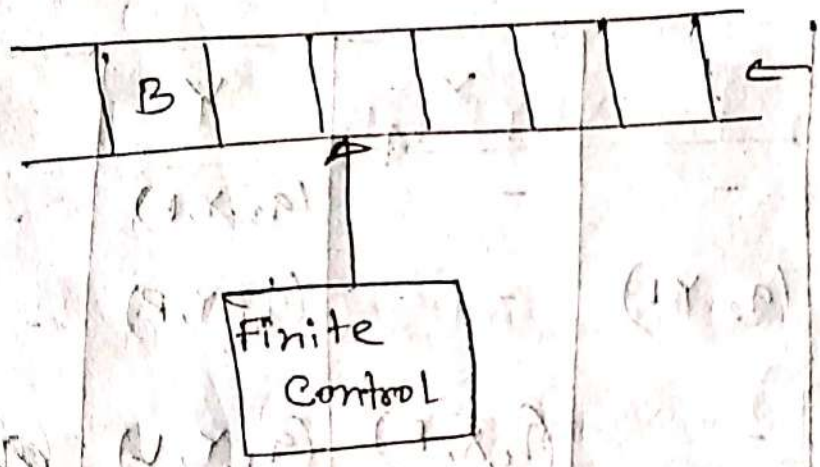
Cycle-13

12.2.19

Undecidability:

$x^n + y^n = z^n \longrightarrow n > 2$ হলে x, y, z কে বের করতে পারব না

Turing Machine (DFA + Infinite Model)



B → Blank (যার input নাই) তাহলে Blank থেকে Move করা শুরু করবে

→ Tape এ কোন কোন Symbol Write করতে পারে

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

★ $(\Gamma \subset \Sigma$ হতে হবে)

★ Γ এর কোন একটা Symbol থাকতে Blank Symbol যেটা Σ তে নাই

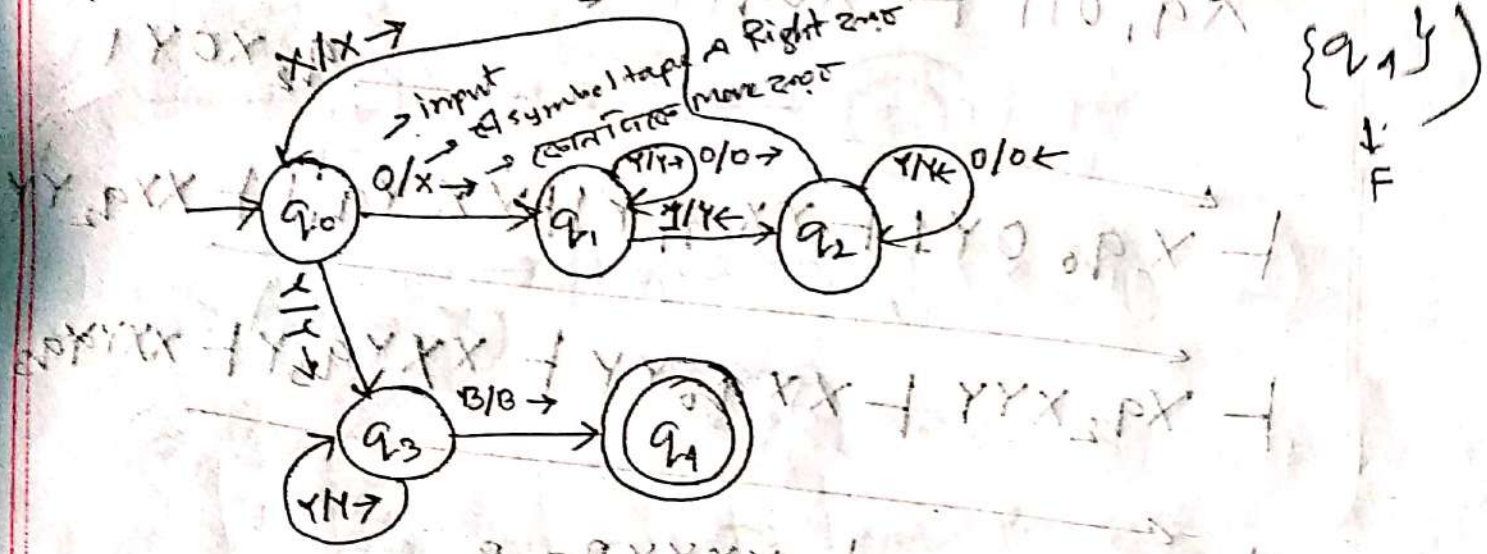
এক্ষেত্রে δ এর কাজ হচ্ছে এরকম:

$\delta(q_0, a) = (P, X, L)$ → direction (head বাম left না Right এ যাবে?)

- q_0 → Current state
- a → input symbol
- P → যে state এ যাবে
- X → যে symbol এ রিপ্লেস করবে, head এ যে symbol write করবে
- L → Right হলে R হবে.

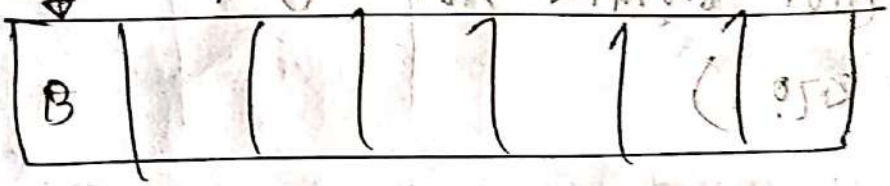
State	0	1	X	Y	B
q_0	(q_1, X, R)	–	–	(q_3, Y, R)	–
q_1	$(q_1, 0, R)$	(q_2, Y, L)	–	(q_1, Y, R)	–
q_2	$(q_2, 0, L)$	–	(q_0, X, R)	(q_2, Y, L)	–
q_3	–	–	–	(q_3, Y, R)	(q_4, B, R)
q_4	–	–	–	–	–

$(\{q_0, q_1, q_2, q_3, q_4\}, \{0, 1\}, \{0, 1, X, Y, B\}, \delta, q_0, B, \{q_4\})$



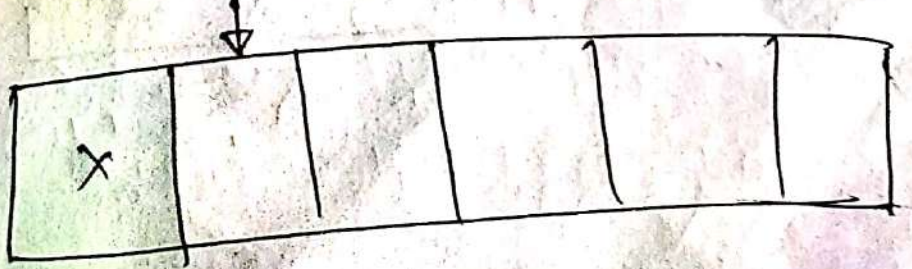
F

একদম শুরুতে Tape-



⊛ ধরি, 0011 Input Process করতে চাইলে Turing Machine এর ডেল্টা-

$q_0 0011 \vdash X q_1 011$



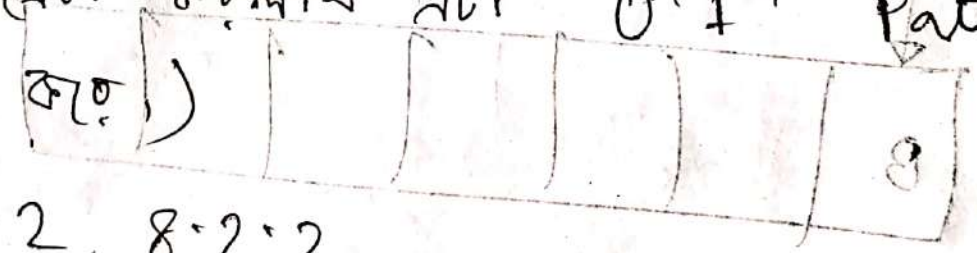
$Xq_1011 \vdash Xq_2011 \vdash Xq_2011$

$\vdash Xq_0 0Y1 \vdash XXq_1Y1 \vdash XXYq_1 1 \vdash XXq_2YY$

$\vdash Xq_2XYY \vdash XXq_0YY \vdash XXYq_3Y \vdash XXYYq_3B$

$\vdash XXYYBq_4B$

* (ওপরের যেটা করলাম এটা 0^n1^n Pattern Recognize করে।)

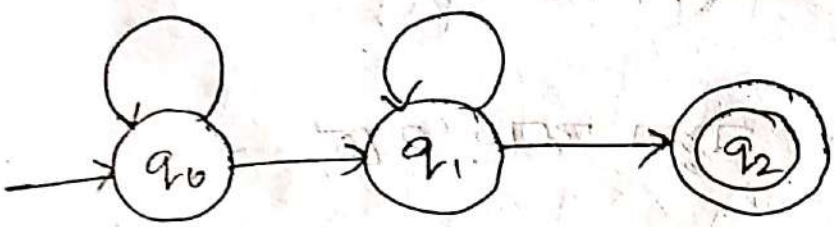


8.2, 8.2.2

Bhaskhar Sir

'A' Day
Cycle - 14

19.2.19



$\delta(q, a, X)$ → stack এর top এ যা আছে

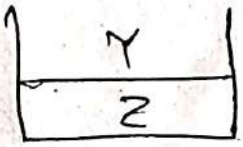
$= (p, \gamma)$

└ stack এ যে symbol টা বসবে

$\gamma = X$ return করলে Stack change হবে না

$\gamma = \in$ " " Pop করে top এর element

$\gamma = YZ$ " " প্রথমে Z থাকবে তারপর Y থাকবে



এখানে 1111 পাওয়া চাইলে

$(q_0, 1111, Z_0)$

ID

$\delta(q_0, 1, Z_0) = (q_1, XZ_0)$

তাহলে ID টা নতুন কত হবে

$(q_1, 111, XZ_0)$

Page - 248, 249, 247