Sl. No.	Date	Title	Page No.	Teacher Sign/ Remarks
		Anshul Jain		

Date: 30/08/18

Unit-I

Central concepts of Automata:

1. Alphabet
2. Strings
3. Languages

→ An alphabet is a finite non-empty set of symbols. It is denoted by Σ.
eg - $\Sigma = \{a, b, c \ldots\ldots z\}$, $\Sigma = \{0, 1\}$

→ A string is a finite sequence of symbols chosen from some alphabet.
eg - 0, 1, 00, 011 etc.

* ε is empty string

Powers of an alphabet (Σ^k):

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

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

* : set of all possible strings.

$\Sigma^* = \Sigma^0 \cup \Sigma^1 \cup \Sigma^2 \cup \ldots\ldots$

$\Sigma^+ = \Sigma^1 \cup \Sigma^2 \cup \Sigma^3 \cup \ldots\ldots$ (except empty string)

Concatenation : A string followed by another string.

$x = 1001$, $y = 0110$

$xy = 10010110$

→ A set of strings all of which chosen from Σ^* is called language.

If Σ is an alphabet and L is a proper subset of Σ^* then, i.e. $L \subseteq \Sigma^*$,

↓

L is the language.

eg: Language of all strings consisting of 'n' zeroes followed by 'n' ones, $n \geq 0$

$L = \{\varepsilon, 01, 0011, \ldots\}$

Set of strings of zero's and one's with an equal number of each, $n \geq 0$

$L = \{\varepsilon, 01, 10, 0011, 1100, 1010, 0101, 1001, 0110, \ldots\}$

→ Σ^* is a language for any alphabet Σ.

→ ϕ is the empty lang. over any alphabet.

→ $\{\varepsilon\}$ is the lang. consisting of only empty strings.

$\phi \neq \varepsilon$

DETERMINISTIC FINITE AUTOMATA :-

→ It consists of :

1. a finite set of states (Q)
2. a finite set of input symbols (Σ)

3. a transition function $\delta(q, a)$ from state q on input symbol 'a'.
4. a start state i.e. one of the state in Q.
5. a set of final or accepting state (F).

F is a subset of Q.

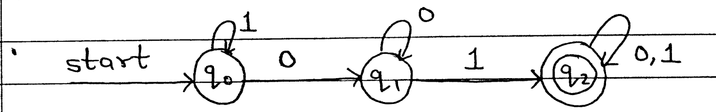
$A = (Q, \Sigma, \delta, q_0, F)$: 5 tuples

→ Construct or design a DFA that accepts all and only strings of 0's and 1's, which has sequence '01' somewhere in the string.

- Transition diagram
- Transition table.

$L = \{W \mid W \text{ is in the form } x01y, \text{ where } x \,\&\, y \text{ are strings of 0's and 1's}\}$

$L = \{01, 001, 101, 010, 011 \ldots\}$

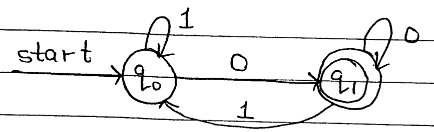


δ	0	1
→ q_0	q_1	q_0
q_1	q_1	q_2
* q_2	q_2	q_2

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

→ Construct a DFA for all strings over $\Sigma = \{0,1\}$ ending with 0.

$L = \{0, 10, 110, 010, 100, \ldots\}$

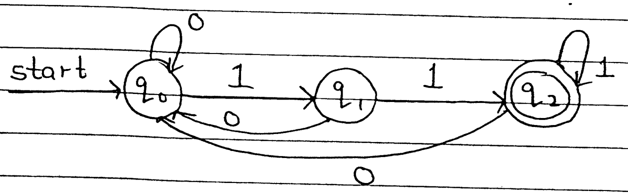


δ	0	1
q_0	q_1	q_0
q_1	q_1	q_0

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

→ Construct a DFA for all strings ending with 11.

$L = \{11, 011, 111, 0111, 1011, \ldots\}$



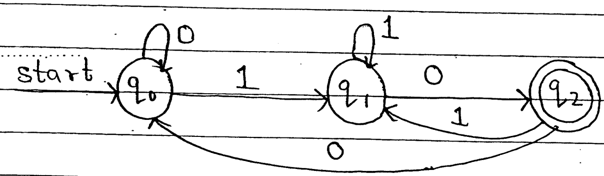
δ	0	1
$\rightarrow q_0$	q_0	q_1
q_1	q_0	q_2
$* q_2$	q_0	q_2

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

Date: 04

→ Construct a DFA that accepts all DFA ending with 10.

$L = \{10, 010, 110, 0110, \ldots\}$

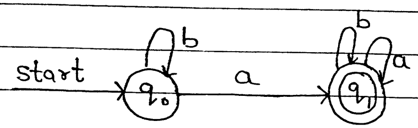


δ	0	1
$\rightarrow q_0$	q_0	q_1
q_1	q_2	q_1
$* q_2$	q_0	q_1

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

→ Construct a DFA for all strings with atleast one 'a' over $\Sigma = \{a,b\}$

$L = \{a, ab, aa, ba, abb, \ldots\}$

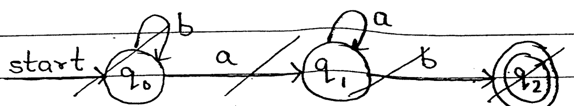


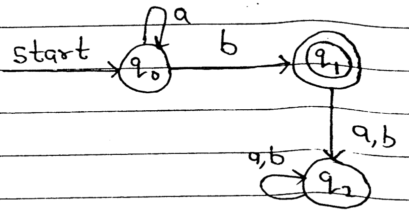
δ	a	b
$\rightarrow q_0$	q_1	q_0
$* q_1$	q_1	q_1

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

→ Construct a DFA that accepts strings with arbitary no. of a's followed by single 'b' over $\Sigma = \{a, b\}$.

$L = \{ b, ab, aab, \cancel{aba}, \ldots \}$





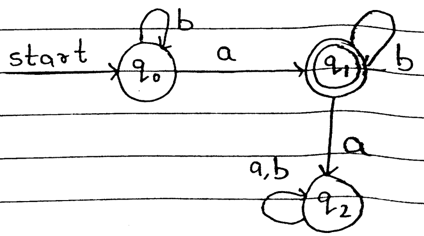
(trap / sink / dead state)

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

$DFA = (\{q_0, q_1, q_2\}, \{a, b\}, \delta, q_0, \{q_1\})$

→ Construct a DFA that accepts strings with exactly one 'a' over $\Sigma = \{a, b\}$

$L = \{ a, ab, ba, abb, bab, bba, \ldots \}$

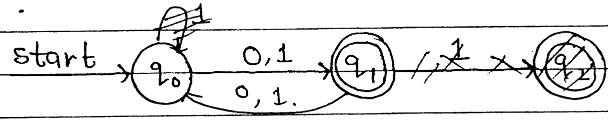


δ	a	b
$\rightarrow q_0$	q_1	q_0
$* q_1$	q_2	q_1
q_2	q_2	q_2

$DFA = (\{q_0, q_1, q_2\}, \{a, b\}, \delta, q_0, \{q_1\})$

→ Construct DFA ... with odd no. of symbols over $\Sigma = \{0, 1\}$.

$L = \{ 0, 1, 000, 111, 101, 010, \ldots \}$

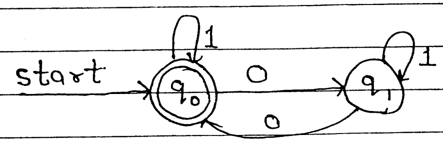


δ	0	1
$\rightarrow q_0$	q_1	q_1
$* q_1$	q_0	q_0

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

** → even no. of 0's and any no. of 1's.

$L = \{\varepsilon, 1, 11, 001, 100, 1100, 1001, 10000, \ldots \}$

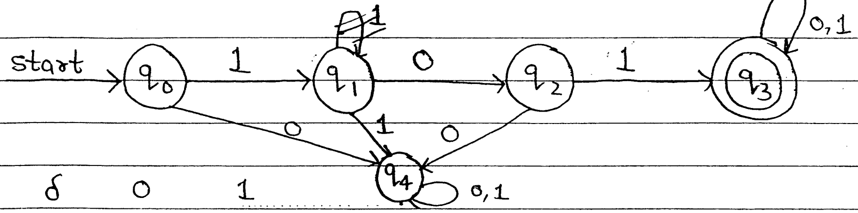


δ	0	1
$\rightarrow, * q_0$	q_1	q_0
q_1	q_0	q_1

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

→ begining with 101 over $\Sigma = \{0,1\}$.

$L = \{101, 1011, 1010, 10101, \ldots\}$

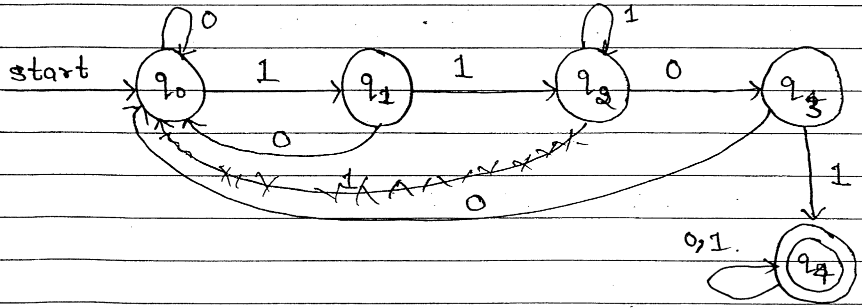


δ	0	1
→ q_0	q_4	q_1
q_1	q_2	q_4
q_2	q_4	q_3
* q_3	q_3	q_3
q_4	q_4	q_4

$DFA = (\{q_0, q_1, q_2, q_3, q_4\}, \{0,1\}, \delta, q_0, \{q_3\})$

→ containing 1101 as a substring.

$L = \{01101, 01101, 111010, \ldots\}$

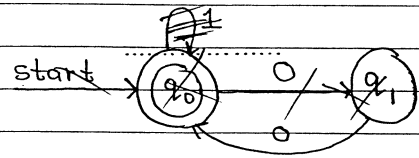


δ	0	1
→ q_0	q_0	q_1
q_1	q_0	q_2
q_2	q_3	q_2
q_3	q_0	q_4
* q_4	q_4	q_4

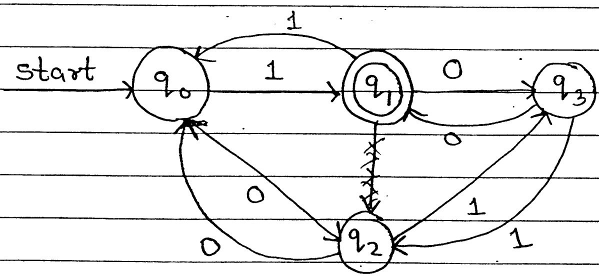
$DFA = (\{q_1, q_2, q_3, q_4\}, \{0,1\}, \delta, q_0, \{q_4\})$

→ ... even no. of 0's and odd no. of 1's

$L = \{1, 001, 100, 010, 00111, \ldots\}$

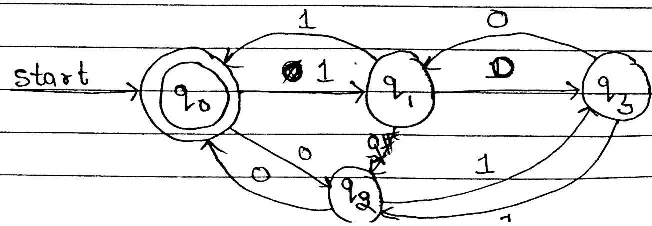


010



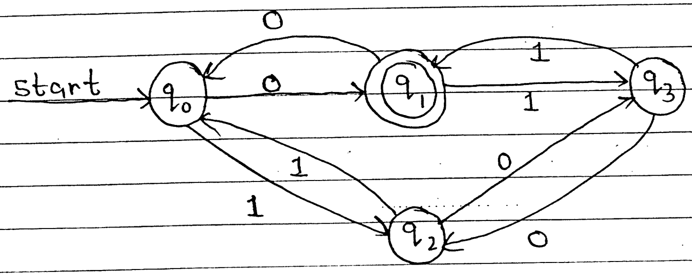
→ ... even no. of 0's & even no. of 1's.

$L = \{\varepsilon, 0011, 1001, 1100, \ldots\}$



→ ..... odd no. of 0's & even no. of 1's.

$L = \{0, 011, 101, 110, ...\}$

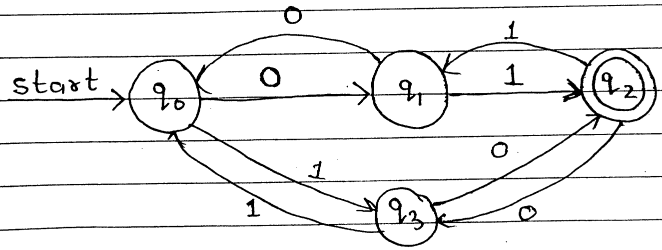


δ	0	1
→ q_0	q_1	q_2
* q_1	q_0	q_3
q_2	q_3	q_0
q_3	q_2	q_1

$A = (\{q_0, q_1, q_2, q_3\}, \{0,1\}, \delta, q_0, \{q_1\})$

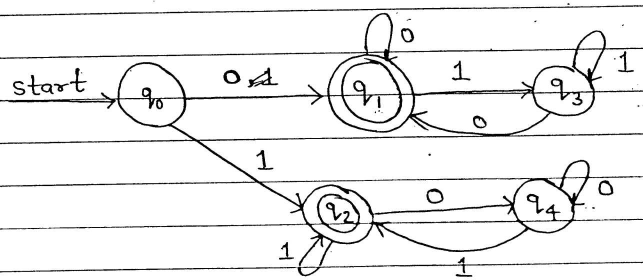
→ ....odd no. of 0's & odd no. of 1's.

$L = \{01, 10, 0001, 1000, ...\}$.



1. Construct a DFA that accepts the string which starts and end with '0' or starts & end with '1'.

$L = \{0, 1, 00, 11, 010, 000, 101, 111, ...\}$

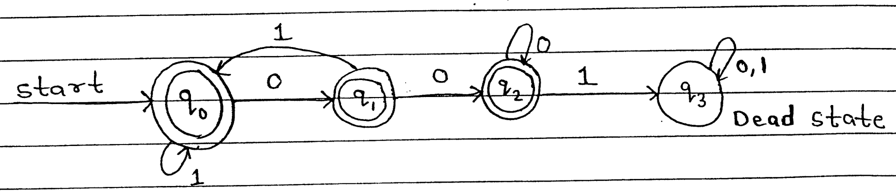


δ	0	1
→ q_0	q_1	q_2
* q_1	q_1	q_3
* q_2	q_4	q_2
q_3	q_1	q_3
q_4	q_2	q_4

$A = (\{q_0, q_1, q_2, q_3, q_4\}, \{0,1\}, \delta, q_0, \{q_1, q_2\})$

2. ... except those containing substring 001.

$L = \{\varepsilon, 0, 1, 01, 10, 11, 010, 00, 100, 111, ...\}$



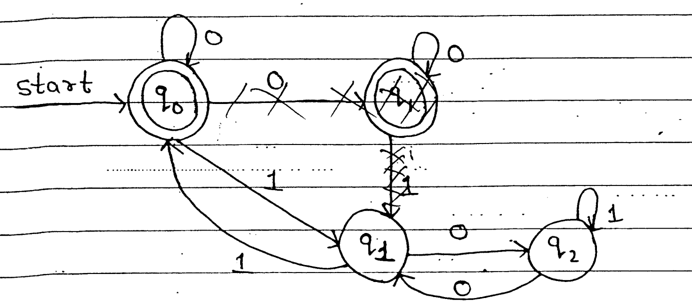
δ	0	1
→ * q_0	q_1	q_0
* q_1	q_2	q_0
* q_2	q_2	q_3
q_3	q_3	q_3

$A = (\{q_0, q_1, q_2, q_3\}, \{0,1\}, \delta, q_0, \{q_0, q_1, q_2\})$

16 8 4 2 1
1 1 0 0 0

3. .... divisible by 3.

$L = \{\varepsilon, 0, 11, 011, 110, 0110, 1001, 001001 \ldots\}$

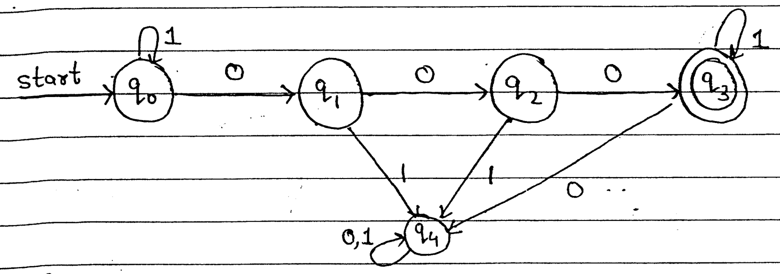


δ	0	1
$\rightarrow * q_0$	q_0	q_1
q_1	q_2	q_0
q_2	q_1	q_2

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

4. ......with exactly 3 consecutive 0's.

$L = \{000, 0001, 1000, 01000, \boxed{00010} \ldots\}$



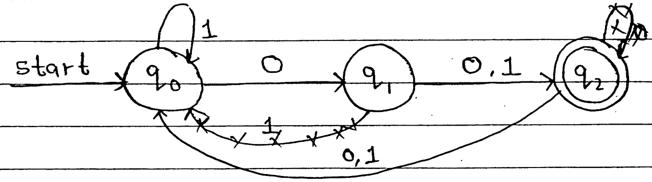
δ	0	1
$\rightarrow q_0$	q_1	q_0
q_1	q_2	q_4
q_2	q_3	q_4
$* q_3$	q_4	q_3
q_4	q_4	q_4

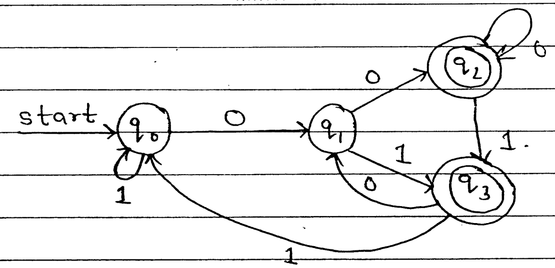
$A = (\{q_0, q_1, q_2, q_3, q_4\}, \{0,1\}, \delta, q_0, \{q_3\})$

5. ...... next to last symbol is '0'.

**

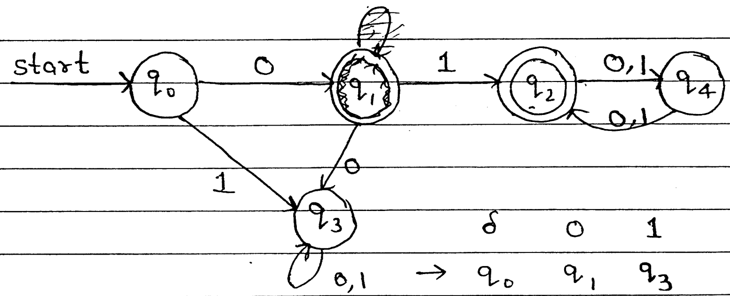
$L = \{01, 00, 101, 000, 001, \ldots\}$





6. $L = \{w \mid w$ is of even length & begins with $01\}$.

$L = \{01, 0100, 0110, 0101, 011100\}$.

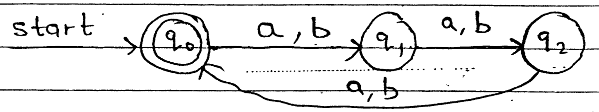


δ	0	1
$\rightarrow q_0$	q_1	q_3
q_1	q_3	q_2
$* q_2$	q_4	q_4
q_3	q_3	q_3
q_4	q_2	q_2

7\. ..over $\Sigma = \{a, b\}$, $L = \{ W \mid |W| \bmod 3 = 0 \}$

↳ length of w

$L = \{ \varepsilon, aaa, aba, abb, bbb, bab, baa, \ldots \}$

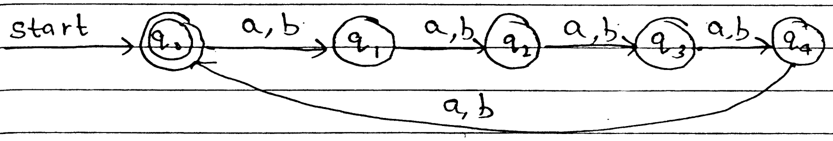


δ	a	b
$\rightarrow * q_0$	q_1	q_1
q_1	q_2	q_2
q_2	q_0	q_0

$A = (\{q_0, q_1, q_2\}, \{a, b\}, \delta, q_0, \{q_0\})$

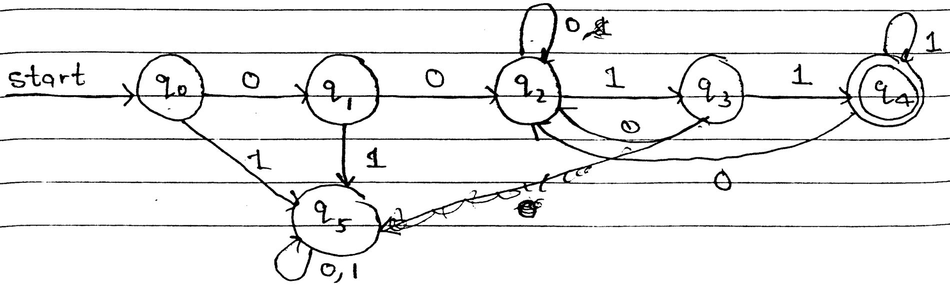
8\. ..... $L = \{ w \mid |w| \bmod 5 = 0 \}$ $\Sigma = \{a, b\}$

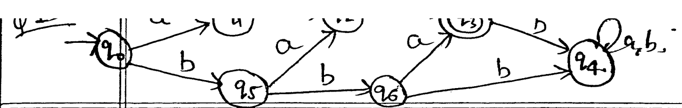
$L = \{ \varepsilon, aaaaa, bbbbb, abbaa, abaaa, \ldots \}$



9\. .... $\Sigma = \{0, 1\}$ starting with 00 & ending with 11.

$L = \{ 0011, 00111, 00011, 001011, 000111, \ldots \}$





10\. .... $\Sigma = \{a, b\}$ that has 'b' as second letter.

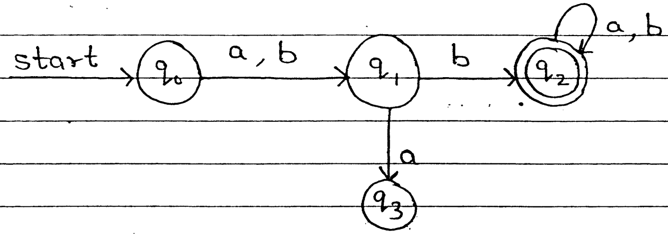
11\. ... $\Sigma = \{0, 1\}$. even no. of 0's

$L \neq$ ~~{ab, b~~

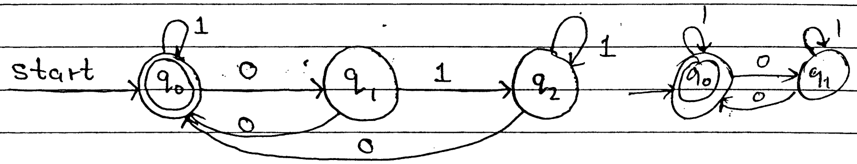
12\. .... $\Sigma = \{0, 1\}$ which are odd binary nos.

13\. .... $\Sigma = \{a, b\}$ that has atleast one 'a' & exactly two b's.

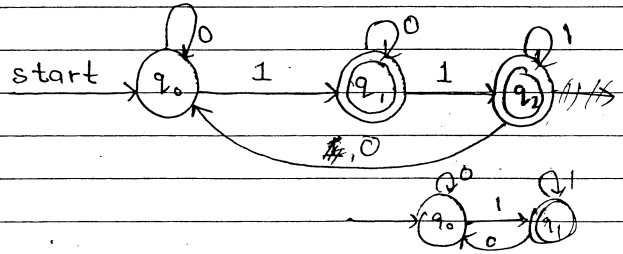
10) $L = \{ ab, bb, abb, bbb, bba, \ldots \}$.



11) $L = \{ \varepsilon, 00, 001, 010, 100, 0011, 00001, 10011, 001100, \ldots \}$



12) $L = \{ \emptyset, 1, 11, 101, 1001, 0101, \ldots \}$



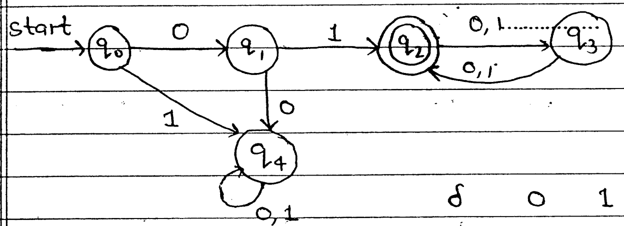
.09.18

Extended Transition Function : $(\hat{\delta})$

~~It describes what happens~~ when we start in any state and follow any sequence.

It is constructed from δ. $[\hat{\delta}]$
$\hat{\delta}$ is a function that ~~text~~ takes a state q and string w and returns a state p. This is the state that automaton reaches when starting in state q and processing the sequence of input in w.

Input string : 011101



δ	0	1
$\rightarrow q_0$	q_1	q_4
q_1	q_4	q_2
$* q_2$	q_3	q_3
q_3	q_2	q_2
q_4	q_4	q_4

$\delta(q_0, 0) = q_1$

$\hat{\delta}(q_0, \varepsilon) = q_0$
$\hat{\delta}(q_0, 0) = \delta(\hat{\delta}(q_0, \varepsilon), 0) = \delta(q_0, 0) = q_1$
$\hat{\delta}(q_0, 01) = \delta(\hat{\delta}(q_0, 0), 1) = \delta(q_1, 1) = q_2$
$\hat{\delta}(q_0, 011) = \delta(\hat{\delta}(q_0, 01), 1) = \delta(q_2, 1) = q_3$
$\hat{\delta}(q_0, 0111) = \delta(\hat{\delta}(q_0, 011), 1) = \delta(q_3, 1) = q_2$
$\hat{\delta}(q_0, 01110) = \delta(\hat{\delta}(q_0, 0111), 0) = \delta(q_2, 0) = q_3$
$\hat{\delta}(q_0, 011101) = \delta(\hat{\delta}(q_0, 01110), 1) = \delta(q_3, 1) = q_2$

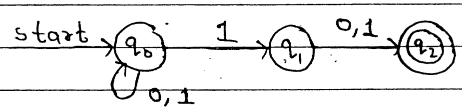
$L(A) = \{ W \mid \hat{\delta}(q_0, W) \text{ is in } F \}$

→ Non-Deterministic Finite Automata:

~~NFA~~ The only difference b/w NFA & DFA is the transition function δ which returns subset of Q for NFA.

→ Construct NFA over $\Sigma = \{0,1\}$ whose second last symbol is '1'.

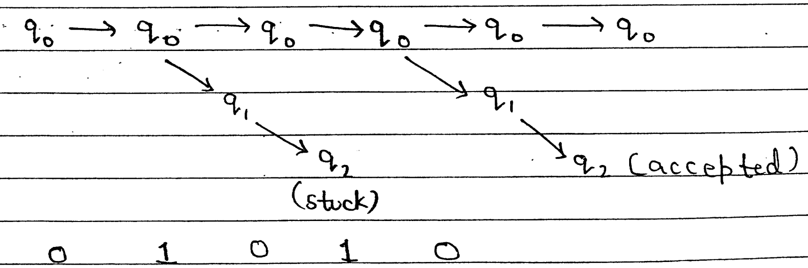
$L = \{ 10, 11, 011, 010, 111, \ldots \}$



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

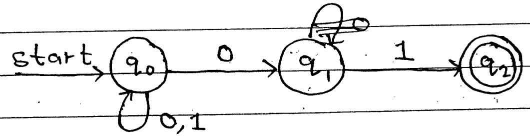
$A = (\{q_0, q_1, q_2\}, \{0,1\}, \delta, q_0, \{q_2\})$

Input string : 01010



→ ..... $\Sigma = \{0, 1\}$ that end in 01.

$L = \{01, 001, 101, 1101, 0001, \ldots\}$



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

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

→ Extended Transition Function ($\hat{\delta}$)

01010

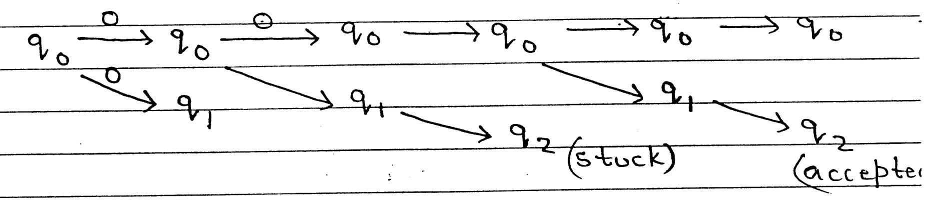
$\hat{\delta}(q_0, \varepsilon) = \{q_0\}$
$\hat{\delta}(q_0, 0) = \delta(\hat{\delta}(q_0, \varepsilon), 0) = \delta(q_0, 0) = \{q_0\}$
$\hat{\delta}(q_0, 01) = \delta(q_0, 1) = \{q_0, q_1\}$
$\hat{\delta}(q_0, 010) = \delta(q_0, 0) \cup \delta(q_1, 0) = \{q_0\} \cup \{q_2\} = \{q_0, q_2\}$
$\hat{\delta}(q_0, 0101) = \delta(q_0, 1) \cup \delta(q_2, 1) = \{q_0, q_1\} \cup \phi = \{q_0, q_1\}$
$\hat{\delta}(q_0, 01010) = \delta(q_0, 0) \cup \delta(q_1, 0) = \{q_0\} \cup \{q_2\}$
$= \{q_0, q_2\}$

NFA: $L(A) = \{w \mid \hat{\delta}(q_0, w) \cap F \neq \phi\}$

Show the states of NFA for input string 00101

$\hat{\delta}(q_0, \varepsilon) = \{q_0\}$
$\hat{\delta}(q_0, 0) = \delta(q_0, 0) = \{q_0, q_1\}$
$\hat{\delta}(q_0, 00) = \delta(q_0, 0) \cup \delta(q_1, 0) = \{q_0, q_1\} \cup \phi$
$= \{q_0, q_1\}$
$\hat{\delta}(q_0, 001) = \delta(q_0, 1) \cup \delta(q_1, 1) = \{q_0\} \cup \{q_2\}$.
$= \{q_0, q_2\}$.
$\hat{\delta}(q_0, 0010) = \delta(q_0, 0) \cup \delta(q_2, 0) = \{q_0, q_1\}$
$\hat{\delta}(q_0, 00101) = \delta(q_0, 1) \cup \delta(q_1, 1) = \{q_0, q_2\}$.

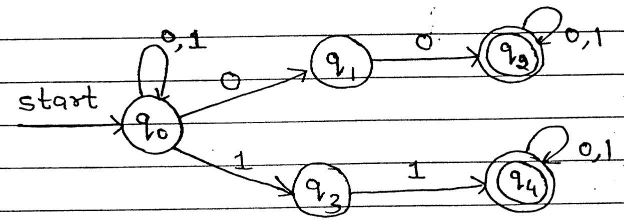
Soln:



0 0 1 0 1

→ ..... $\Sigma = \{0, 1\}$ that has either two consecutive 0's
** or two consecutive 1's.

$L = \{00, 11, 100, 001, 011, 110, 000, 0011, \ldots\}$.



→ Compute $\hat{\delta}(q_0, 01001)$

δ	0	1
$\rightarrow q_0$	$\{q_0,q_1\}$	$\{q_0,q_3\}$
q_1	$\{q_2\}$	ϕ
$*q_2$	$\{q_2\}$	$\{q_2\}$
q_3	ϕ	$\{q_4\}$
$*q_4$	$\{q_4\}$	$\{q_4\}$

01001

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

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

$\hat{\delta}(q_0,01) = \delta(q_0,1) \cup \delta(q_1,1) = \{q_0,q_3\}$

$\hat{\delta}(q_0,010) = \delta(q_0,0) \cup \delta(q_2,0) = \{q_0,q_1,\cancel{q_2}\}$.

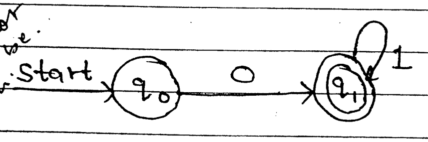
$\hat{\delta}(q_0,0100) = \delta(q_0,0) \cup \delta(q_1,0) \cup \delta(q_2,0)$
$= \{q_0,q_1,q_2\}$

$\hat{\delta}(q_0,01001) = \delta(q_0,1) \cup \delta(q_1,1) \cup \delta(q_2,1)$
$= \{q_0,q_3,q_2\}$

→ ...... $\Sigma = \{0,1\}$ having exactly one '0' & should start 0.

$L = \{0, 01, 011, 0111, \ldots\ldots\}$.

** very trans. need not to be shown.

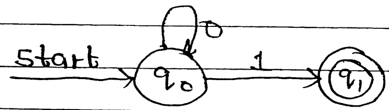


δ	0	1
$\rightarrow q_0$	$\{q_1\}$	ϕ
$*q_1$	ϕ	$\{q_1\}$

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

→ ..... $\Sigma = \{0,1\}$ having exactly one '1' & should end in '1'.

$L = \{1, 01, 001, 0001, \ldots\}$.

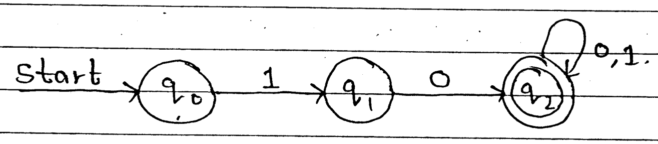


δ	0	1
$\rightarrow q_0$	$\{q_0\}$	$\{q_1\}$
$*q_1$	ϕ	ϕ

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

→ ...... $\Sigma = \{0,1\}$ that begin with 10.

$L = \{10, 100, 101, 1000, 1011, 1010, \ldots\ldots\}$.

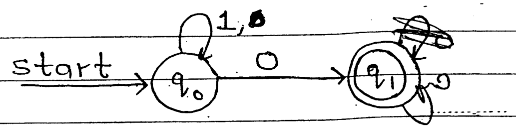


δ	0	1
$\rightarrow q_0$	ϕ	$\{q_1\}$
q_1	$\{q_2\}$	ϕ
$*q_2$	$\{q_2\}$	$\{q_2\}$

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

→ ___ $\Sigma = \{0,1\}$ that accept even binary nos.

$L = \{0, 00, 10, 100, 110, 1000, \ldots\}$.



δ	0	1
$\rightarrow q_0$	$\{q_1\}$	$\{q_0\}$
$* q_1$	ϕ	ϕ

$\hat{\delta}(q_0, 10010)$

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

$\cancel{\hat{\delta}(q_0, 0) = \{q_0, q_1\}}$.

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

$\hat{\delta}(q_0, 10) = \delta(q_0, 0) = \{q_0, q_1\}$.

$\hat{\delta}(q_0, 100) = \delta(q_0, 0) \cup \delta(q_1, 0) = \{q_0, q_1\}$.

$\hat{\delta}(q_0, 1001) = \delta(q_0, 1) \cup \delta(q_1, 1) = \{q_0\}$

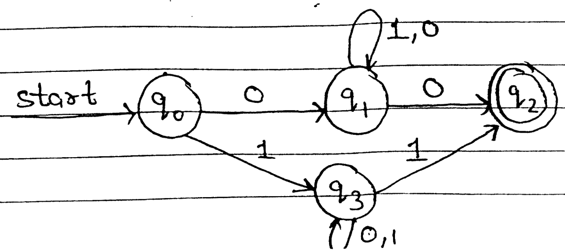
$\hat{\delta}(q_0, 10010) = \delta(q_0, 0) = \{q_0, q_1\}$

→ .... $\Sigma = \{0,1\}$ with same I & last symbol.

→ .... $\Sigma = \{0,1\}$ that begin with 11 & end with 11 or begin with 00 & end with 00.

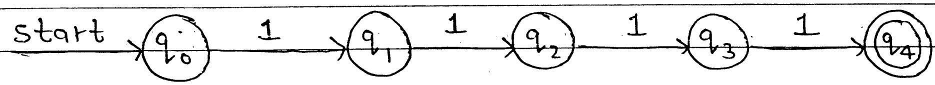
→ --- $\Sigma = \{0,1\}$ with atleast 3 0's or atleast 2,1's

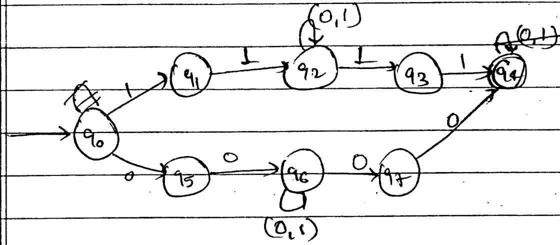
1. $L = \{00, 11, 101, 010, 0110, 1001, 1011 \ldots\}$



δ	0	1
$\rightarrow q_0$	$\{q_1\}$	$\{q_3\}$
q_1	$\{q_1, q_2\}$	$\{q_1\}$
$* q_2$	ϕ	ϕ
q_3	$\{q_3\}$	$\{q_2, q_3\}$.

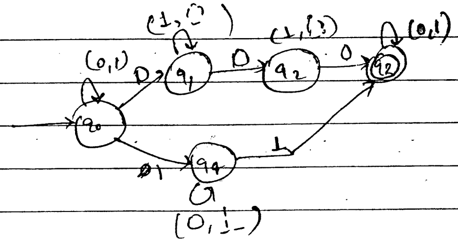
2. $L = \{1111, 0000, 11111, 11011, 00100, 00000, \ldots\}$



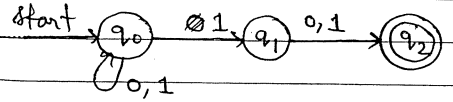


$L = \{000, 11, 0001, 10010, 11000, 110, 011, \ldots\}$.

3.



Equivalence of NFA and DFA :- [Lazy evaluation]



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

① step 1 : $\{q_0\}$ is start state of NFA, so is for DFA.

$\delta_D(\{q_0\}, 0) = \{q_0\}$ — ①

$\delta_D(\{q_0\}, 1) = \{q_0, q_1\}$ — ②

② $\delta_D(\{q_0, q_1\}, 0) = \delta_N(q_0, 0) \cup \delta_N(q_1, 0) = \{q_0\} \cup \{q_2\} = \{q_0, q_2\}$ — ③

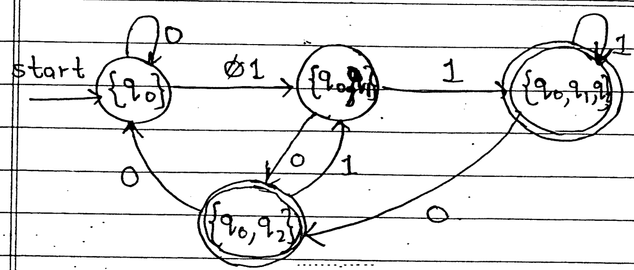
$\delta_D(\{q_0, q_1\}, 1) = \delta_N(q_0, 1) \cup \delta_N(q_1, 1) = \{q_0, q_1\} \cup \{q_2\}$
$= \{q_0, q_1, q_2\}$ — ④

③ $\delta_D(\{q_0, q_2\}, 0) = \delta_N(q_0, 0) \cup \delta_N(q_2, 0) = \{q_0\}$

$\delta_D(\{q_0, q_3\}, 1) = \delta_N(q_0, 1) \cup \delta_N(q_3, 1) = \{q_0, q_1\}$.

④ $\delta_D(\{q_0, q_1, q_2\}, 0) = \delta_N(q_0, 0) \cup \delta_N(q_1, 0) \cup \delta_N(q_2, 0)$
$= \{q_0, q_2\}$

$\delta_D(\{q_0, q_1, q_2\}, 1) = \delta_N(q_0, 1) \cup \delta_N(q_1, 1) \cup \delta_N(q_1, 1)$
$= \{q_0, q_1, q_2\}$



δ_D	0	1
$\rightarrow \{q_0\}$	$\{q_0\}$	$\{q_0, q_1\}$
$\{q_0, q_1\}$	$\{q_0, q_2\}$	$\{q_0, q_1, q_2\}$
$* \{q_0, q_2\}$	$\{q_0\}$	$\{q_0, q_1\}$
$* \{q_0, q_1, q_2\}$	$\{q_0, q_2\}$	$\{q_0, q_1, q_2\}$

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

→ Convert the NFA to its equivalent DFA :-

δ	0	1
$\rightarrow p$	$\{p, q\}$	$\{p\}$
q	ϕ	$\{r\}$
$* r$	$\{p, r\}$	$\{q\}$

① Step 1 : $\{p\}$ is start state of NFA, so ~~for~~ is for DFA.

$\delta_D(\{p\}, 0) = \{p, q\}$ — ①

papergrid

Date: / /

$\delta_D(p,1) = \{p\}$ — ②

② $\delta_D(\{p,q\},0) = \delta_N(p,0) \cup \delta_N(q,0) = \{p,q\}$

$\delta_D(\{p,q\},1) = \delta_N(p,1) \cup \delta_N(q,1) = \{p,r\}$ — ③

③ $\delta_D(\{p,r\},0) = \delta_N(p,0) \cup \delta_N(r,0) = \{p,q,r\}$ — ③

$\delta_D(\{p,r\},1) = \delta_N(p,1) \cup \delta_N(r,1) = \{p,q\}$.

④ $\delta_D(\{p,q,r\},0) = \{p,q,r\}$

$\delta_D(\{p,q,r\},1) = \{p,q,r\}$

NFA – n states

DFA – 2^n states (max.)

Date: 14/09/18

Equivalence of NFA and DFA :

→ Subset Construction

steps: 1) Start with initial state.

2) Mention no. of states for NFA and DFA.

3) Evaluate δ_D using δ_N & Transition table.

4) Renaming states

5) Transition function (δ)

6) Accessible states

7) Transition function after step 6.

8) DFA transition diag.

9) Five tuple definition

- Constructing all subsets of set of states of the NFA.
- The subset construction starts from NFA

  $N = \{Q_N, \Sigma, \delta_N, q_0, F_N\}$
- Its goal is the description of DFA.

  $D = \{Q_D, \Sigma, \delta_D, \{q_0\}, F_D\}$

  such that $L(D) = L(N)$.
- Input alphabets for both automata ~~is~~ are same.
- The start state of D is the set containing only start state of N.
- The other components of D are constructed as follows –

→ Q_D is set of subsets of Q_N ~~i.e.~~

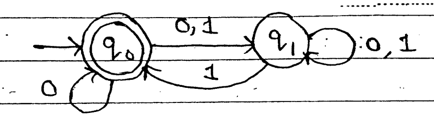
Maximum it can be 2^n.

→ F_D is the set of subsets 'S' of Q_N such that $S \cap F_N \neq \phi$

i.e. F_D is all ~~subsets~~ sets of N that include atleast one accepting state of N.

→ For each $S \subset Q_N$ and for each input symbol 'a' in Σ, $\delta_D(S,a) = \bigcup_{P \in S} \delta(P,a)$

Q.1. Convert given NFA to equivalent DFA using subset construction method.



step 1: start state of DFA is $[q_0]$.

2: NFA has 2 states, so DFA has max. $2^2 = 4$ ~~subsets~~ states.

3: $\delta_D([q_0],0) = \delta_N(\{q_0\},0) = [q_0q_1]$

$\delta_D([q_0],1) = \delta_N(\{q_0\},1) = [q_1]$

$\delta_D([q_1],0) = \delta_N(\{q_1\},0) = [q_1]$

$\delta_D([q_1],1) = \delta_N(\{q_1\},1) = [q_0q_1]$

$\delta_D([q_0q_1],0) = \delta_N(\{q_0\},0) \cup \delta_N(\{q_1\},0) = [q_0q_1]$

$\delta_D([q_0,q_1],1) = \delta_N(\{q_0\},1) \cup \delta_N(\{q_1\},1) = [q_0q_1]$

$\delta_D(\phi,0) = \phi$

$\delta_D(\phi,1) = \phi$

4: $[q_0], [q_1], [q_0q_1], \phi$

Rename the states as A, B, C & D respectively.

5:

δ	0	1
→*A	C	B
B	B	C
*C	C	C
D	D	D

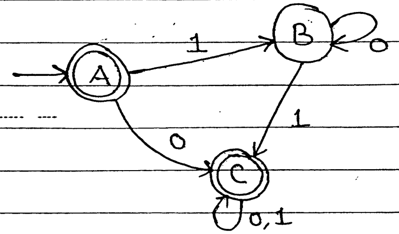
6: Accessible states from A are (A, B, C).

7:

δ	0	1
→*A	C	B
B	B	C
*C	C	C

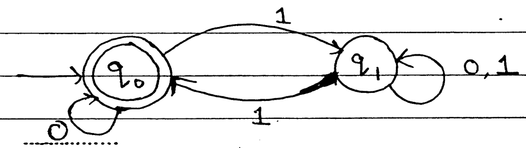
papergrid
Date: / /

8:



9: $D = (\{A,B,C\}, \{0,1\}, \delta, A, \{A,C\})$

Q.2. NFA → DFA



1: Start state of DFA is $[q_0]$.

2: NFA has 2 states, so DFA has max. 4 states.

3: $\delta_D([q_0],0) = \delta_N(\{q_0\},0) = [q_0]$

$\delta_D([q_0],1) = \delta_N(\{q_0\},1) = [q_1]$

$\delta_D([q_1],0) = \delta_N(\{q_1\},0) = [q_1]$

$\delta_D([q_1],1) = \delta_N(\{q_1\},1) = [q_0q_1]$

$\delta_D([q_0q_1],0) = \delta_N(\{q_0\},0) \cup \delta_N(\{q_1\},0) = [q_0q_1]$

$\delta_D([q_0q_1],1) = \delta_N(\{q_0\},1) \cup \delta_N(\{q_1\},1) = [q_0q_1]$

$\delta_D(\phi,0) = \phi$

$\delta_D(\phi,1) = \phi$

4: $[q_0], [q_1], [q_0q_1], \phi$

Rename the states as A, B, C and D respectively.

5:

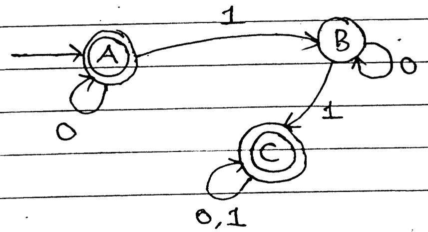
δ	0	1
→*A	A	B
B	B	C
*C	C	C
D	D	D

6: Accessible states from A are (A, B, C)

7:

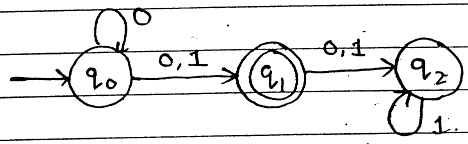
δ	0	1
$\rightarrow *$ A	A	B
B	B	C
$*$ C	C	C

8:



~~9:~~ $D = (\{A, B, C\}, \{0,1\}, \delta_D, A, \{A, C\})$

Q3.



1: start state of DFA is $[q_0]$.

2: NFA has 3 states, max. states of DFA is 8.

3: $\delta_D([q_0],0) = \delta_N(\{q_0\},0) = [q_0 q_1]$

$\delta_D([q_0],1) = \delta_N(\{q_0\},1) = [q_1]$

$\delta_D([q_1],0) = \delta_N(\{q_1\},0) = [q_2]$

$\delta_D([q_1],1) = \delta_N(\{q_1\},1) = [q_2]$

$\delta_D([q_2],0) = \delta_N(\{q_2\},0) = \phi$

$\delta_D([q_2],1) = \delta_N(\{q_2\},1) = [q_2]$

$\delta_D([q_0 q_1],0) = \delta_N(\{q_0\},0) \cup \delta_N(\{q_1\},0) = [q_0 q_1 q_2]$

$\delta_D([q_0 q_1],1) = \delta_N(\{q_0\},1) \cup \delta_N(\{q_1\},1) = [q_1 q_2]$

$\delta_D([q_0 q_2],0) = \delta_N(\{q_0\},0) \cup \delta_N(\{q_2\},0) = [q_0 q_1]$

$\delta_D([q_0 q_2],1) = \delta_N(\{q_0\},1) \cup \delta_N(\{q_2\},1) = [q_1 q_2]$

$\delta_D([q_1 q_2],0) = \delta_N(\{q_1\},0) \cup \delta_N(\{q_2\},0) = [q_2]$

$\delta_D([q_1 q_2],1) = \delta_N(\{q_1\},1) \cup \delta_N(\{q_2\},1) = [q_2]$

$\delta_D([q_0 q_1 q_2],0) = \delta_N(\{q_0\},0) \cup \delta_N(\{q_1\},0) \cup \delta_N(\{q_2\},0) = [q_0 q_1 q_2]$

$\delta_D([q_0 q_1 q_2],1) = \delta_N(\{q_0\},1) \cup \delta_N(\{q_1\},1) \cup \delta_N(\{q_2\},1) = [q_1 q_2]$

$\delta_D(\phi,0) = \delta_N(\phi,0) = \phi$

$\delta_D(\phi,1) = \delta_N(\phi,1) = \phi$

4: $[q_0], [q_1], [q_2], [q_0 q_1], [q_0 q_2], [q_1 q_2], [q_0 q_1 q_2], \phi$

Rename the states as A, B, C, D, E, F, G & H respectively.

5:

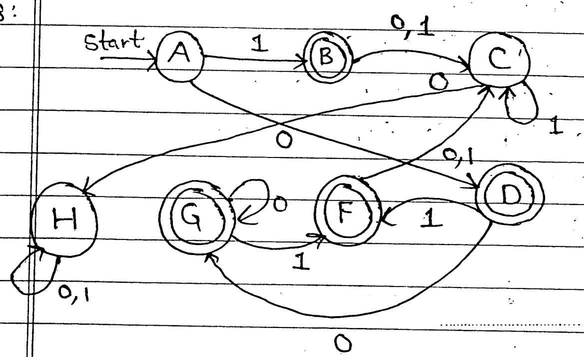
δ	0	1
$\rightarrow$ A	D	B
$*$ B	C	C
C	~~G~~ H	C
$*$ D	G	F
E	D	F
$*$ F	C	C
$*$ G	G	F
H	H	H

6: Accessible states from A are (A, B, C, D, F, G, H)

7:

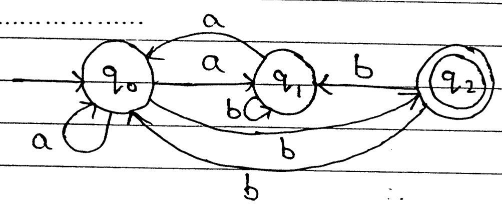
δ	0	1
$\rightarrow$ A	D	B
$*$ B	C	C
C	H	C
$*$ D	G	F
$*$ F	C	C
$*$ G	G	F
H	H	H

8:



~~9:~~ $D: (\{A,B,C,D,F,G,H\}, \{0,1\}, \delta_D, A, \{B,D,F,G\})$

Q4. Convert NFA to DFA using subset construction method:



1. Start state of DFA is $[q_0]$
2. NFA has 3 states, max. states of DFA is 8.
3. $\delta_D([q_0],a) = [q_0q_1]$

$\delta_D([q_0],b) = [q_2]$

$\delta_D([q_1],a) = [q_0]$

$\delta_D([q_1],b) = [q_1]$

$\delta_D([q_2],a) = \phi$

$\delta_D([q_2],b) = [q_0q_1]$

$\delta_D([q_0q_1],a) = \delta_N(\{q_0\},a) \cup \delta_N(\{q_1\},a) = [q_0q_1]$

$\delta_D([q_0q_1],b) = \delta_N(\{q_0\},b) \cup \delta_N(\{q_1\},b) = [q_1q_2]$

$\delta_D([q_0q_2],a) = \delta_N(\{q_0\},a) \cup \delta_N(\{q_2\},a) = [q_0q_1]$

$\delta_D([q_0q_2],b) = \delta_N(\{q_0\},b) \cup \delta_N(\{q_2\},b) = [q_0q_1q_2]$

$\delta_D([q_1q_2],a) = \delta_N(\{q_1\},a) \cup \delta_N(\{q_2\},a) = [q_0]$

$\delta_D([q_1q_2],b) = \delta_N(\{q_1\},b) \cup \delta_N(\{q_2\},b) = [q_0q_1]$

$\delta_D([q_0q_1q_2],a) = \delta_N(\{q_0\},a) \cup \delta_N(\{q_1\},a) \cup \delta_N(\{q_2\},a) = [q_0q_1]$

$\delta_D([q_0q_1q_2],b) = \delta_N(\{q_0\},b) \cup \delta_N(\{q_1\},b) \cup \delta_N(\{q_2\},b) = [q_0q_1q_2]$

~~$\delta_D([q_0q_1q_2], + =$~~

$\delta_D(\phi,a) = \phi$

$\delta_D(\phi,b) = \phi$

δ	a	b
$\rightarrow [q_0]$	$[q_0q_1]$	$[q_2]$
$[q_1]$	$[q_0]$	$[q_1]$
$* [q_2]$	ϕ	$[q_0q_1]$
$[q_0q_1]$	$[q_0q_1]$	$[q_1q_2]$
$* [q_0q_2]$	$[q_0q_1]$	$[q_0q_1q_2]$
$* [q_1q_2]$	$[q_0]$	$[q_0q_1]$
$* [q_1q_2q_3]$	$[q_0q_1]$	$[q_0q_1q_2]$
ϕ	ϕ	ϕ

4: Renaming the states as A, B, C, D, E, F, G, H of $[q_0], [q_1], [q_2], [q_0q_1], [q_0q_2], [q_1q_2], [q_0q_1q_2], \phi$.

5:

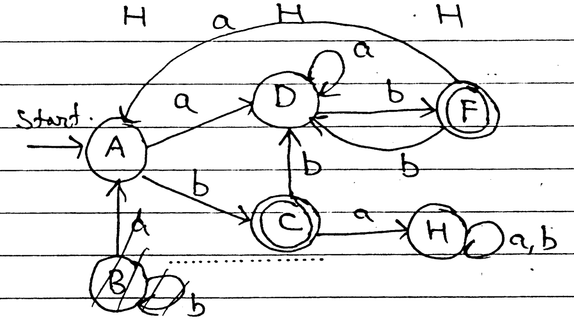
δ	a	b
$\rightarrow$ A	D	C
B	A	B
* C	~~ϕ~~ H	D
D	D	F
* E	D	G
* F	A	D
* G	D	G
H	H	H

ssible states: (A, ~~B~~, C, D, F, H)

7:

δ	a	b
$\rightarrow$ A	D	C
~~B~~	~~A~~	~~B~~
* C	~~ϕ~~ H	D
D	D	F
* F	A	D
H	H	H

8:



9: D : ({A, ~~B~~, C, D, F, H}, {a, b}, δ_D, A, {C, F})

3: Lazy evaluation.

** can also be a state

δ	0	1
$\rightarrow q_0$	$\{q_0 q_1\}$	$\{q_1\}$
* q_1	$\{q_2\}$	$\{q_2\}$
q_2	ϕ	$\{q_2\}$

1: $[q_0]$ is the start state of NFA, so is for DFA. — ①

2: $\delta_D([q_0], 0) = [q_0 q_1]$ — ②
$\delta_D([q_0], 1) = [q_1]$ — ③

$\delta_D([q_0 q_1], 0) = [q_0 q_1 q_2]$ — ④
$\delta_D([q_0 q_1], 1) = [q_1 q_2]$ — ⑤

$\delta_D([q_1], 0) = [q_2]$ — ⑥
$\delta_D([q_1], 1) = [q_2]$

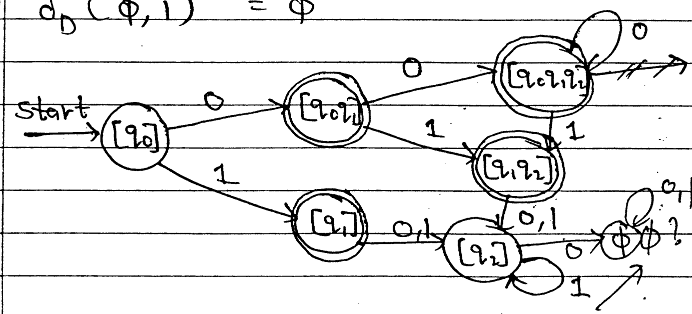
$\delta_D([q_0 q_1 q_2], 0) = [q_0 q_1 q_2]$
$\delta_D([q_0 q_1 q_2], 1) = [q_1 q_2]$

$\delta_D([q_1 q_2], 0) = [q_2]$
$\delta_D([q_1 q_2], 1) = [q_2]$

$\delta_D([q_2], 0) = \phi$ — ⑦
$\delta_D([q_2], 1) = [q_2]$

$\delta_D(\phi, 0) = \phi$
$\delta_D(\phi, 1) = \phi$

3:

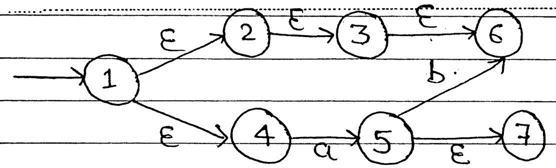


4:

δ_D	0	1
$\rightarrow [q_0]$	$[q_0 q_1]$	$[q_1]$
* $[q_1]$	$[q_2]$	$[q_2]$
$[q_2]$	ϕ	$[q_2]$
* $[q_0 q_1]$	$[q_0 q_1 q_2]$	$[q_1 q_2]$
* $[q_1 q_2]$	$[q_2]$	$[q_2]$
* $[q_0 q_1 q_2]$	$[q_0 q_1 q_2]$	$[q_1 q_2]$
ϕ	ϕ	ϕ

5: D : ({$[q_0]$, $[q_1]$, $[q_2]$, $[q_0 q_1]$, $[q_1 q_2]$, $[q_0 q_1 q_2]$}, {0, 1}, δ_D, $[q_0]$, {$[q_1]$, $[q_0 q_1]$, $[q_1 q_2]$, $[q_0 q_1 q_2]$})

→ FINITE AUTOMATA WITH ε-TRANSITION:-



ε-closure.

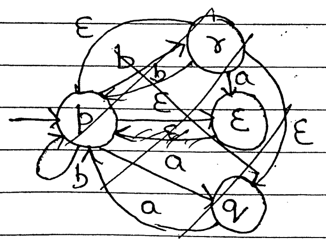
Eclose(1) = {1, 2, 3, 6, 4}
Eclose(2) = {2, 3, 6}
Eclose(3) = {3, 6}
Eclose(4) = {4}
Eclose(5) = {5, 7}
Eclose(6) = {6}
Eclose(7) = {7}

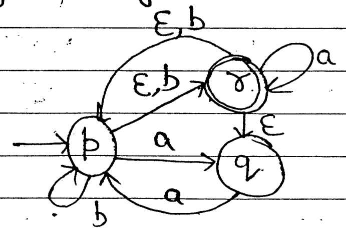
⇒

δ	ε	a	b
→ p	{r}	{q}	{p, r}
q	φ	{p}	φ
* r	{p, q}	{r}	{p}

1. Compute Eclose of each state.
2. Write set of all strings of length `3` or less.

1.





L = {ε, a, ab, aa, aba, abb, ...}

papergrid
Date: / /

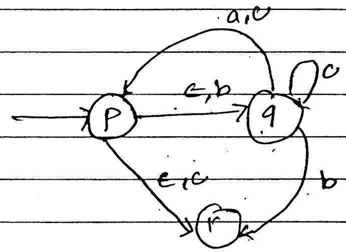
Eclose(p) = {p, q, r}
Eclose(q) = {q}
Eclose(r) = {r, p, q}.

2. L = {ε, b, a,

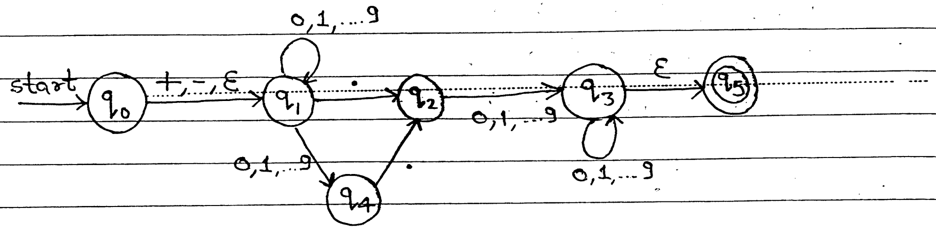
⇒

δ	ε	a	b	c
→ p	{q, r}	φ	{q}	{r}
q	φ	{p}	{r}	{p, q}
* r	φ	φ	φ	φ

Eclose(p) = {p, q, r}.
Eclose(q) = {q}
Eclose(r) = {r}.



1.



δ	ϵ	+,-	.	0,1,...9
$\rightarrow q_0$	$\{q_1\}$	$\{q_1\}$	ϕ	ϕ
q_1	ϕ	ϕ	$\{q_2\}$	$\{q_1, q_4\}$
q_2	ϕ	ϕ	ϕ	$\{q_3\}$
q_3	$\{q_5\}$	ϕ	ϕ	$\{q_3\}$
q_4	ϕ	ϕ	$\{q_2\}$	ϕ
$* q_5$	ϕ	ϕ	ϕ	ϕ

Extended Transition function ($\hat{\delta}$):

i) $\hat{\delta}(q_0, 5.6)$

Eclose(q_0)

$\hat{\delta}(q_0, \epsilon) = \{q_0, q_1\}$.   Eclose($q_1$) $\cup$ Eclose(q_4)

$\hat{\delta}(q_0, 5) = \delta(q_0, 5) \cup \delta(q_1, 5) = \{q_1, q_4\}$

$\hat{\delta}(q_0, 5.) = \delta(q_1, .) \cup \delta(q_4, .) = \{q_2\}$. Eclose($q_2$)

$\hat{\delta}(q_0, 5.6) = \delta(q_2, 6) = \{q_3\}$.

Eclose(q_3) $= \{q_3, q_5\}$

ii) $\hat{\delta}(q_0, +11.24)$

$\hat{\delta}(q_0, \epsilon) = \text{Eclose}(q_0) = \{q_0, q_1\}$.

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

Eclose(q_1) $= \{q_1\}$

$\hat{\delta}(q_0, +1) = \delta(q_1, 1) = \{q_1, q_4\}$.

Eclose(q_1) $\cup$ Eclose(q_4) $= \{q_1, q_4\}$.

$\hat{\delta}(q_0, +11) = \delta(q_1, 1) \cup \delta(q_4, 1) = \{q_1, q_4\}$.

Eclose(q_1) $\cup$ Eclose(q_4) $= \{q_1, q_4\}$.

$\hat{\delta}(q_0, +11.) = \delta(q_1, .) \cup \delta(q_4, .) = \{q_2\}$.

Eclose(q_2) $= \{q_2\}$.

$\hat{\delta}(q_0, +11.2) = \delta(q_2, 2) = \{q_3\}$.

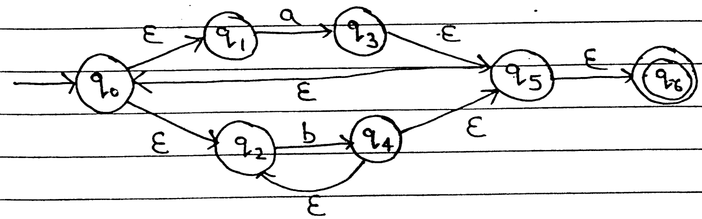
Eclose(q_3) $= \{q_3, q_5\}$.

$\hat{\delta}(q_0, +11.24) = \delta(q_3, 4) \cup \delta(q_5, 4)$

$= \{q_3\}$.

Eclose(q_3) $= \{q_3, q_5\}$

iii) Find $\hat{\delta}$ for given NFA :



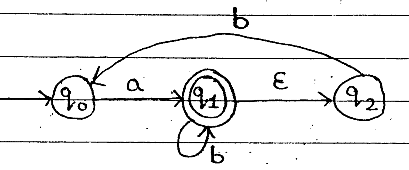
$\hat{\delta}(q_0, abba)$

$\hat{\delta}(q_0, \epsilon) = \text{Eclose}(q_0) = \{q_0, q_1, q_2\}$

$\hat{\delta}(q_0, a) = \delta(q_0, a) \cup \delta(q_1, a), \delta(q_2, a)$

Eclose($\{q_3\}$) $= \{q_3, q_5, q_0, q_6\} q_2$

Conversion of ε-NFA to DFA:



Step 1: Compute Eclose of all states.

$Eclose(q_0) = \{q_0\}$

$Eclose(q_1) = \{q_1, q_2\}$

$Eclose(q_2) = \{q_2\}$

Step 2: Start state of DFA is $Eclose(q_0) = [q_0]$ — ①

Step 3: $\delta_D([q_0], a) = Eclose(\delta_N([q_0], a)) = Eclose(q_1) = [q_1 q_2]$ - ②

$\delta_D([q_0], b) = Eclose(\phi) = \phi$ - ③

Step 4: $\delta_D([q_1 q_2], a) = Eclose(\delta_N([q_1 q_2], a)) = Eclose(\delta_N(q_1, a) \cup \delta_N(q_2, a))$

$= Eclose(\phi) = \phi$

$\delta_D([q_1 q_2], b) = Eclose(\delta_N(q_1, b) \cup \delta_N(q_2, b)) = Eclose(q_0 \cup q_1)$

$= Eclose(\{q_0, q_1\})$

$= [q_0 q_1 q_2]$ - ④

$\delta_D(\phi, a) = \phi$

$\delta_D(\phi, b) = \phi$

$\delta_D([q_0 q_1 q_2], a) = Eclose(\delta_N(q_0, a) \cup \delta_N(q_1, a) \cup \delta_N(q_2, a))$

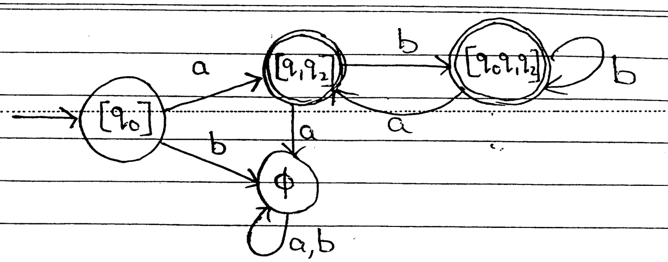
$= Eclose(\{q_1\})$

$= [q_1 q_2]$

$\delta_D([q_0 q_1 q_2], b) = Eclose(\delta_N(q_0, b) \cup \delta_N(q_1, b) \cup \delta_N(q_2, b))$

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

Step 5:



Step 6:

	δ	a	b
→	$[q_0]$	$[q_1 q_2]$	ϕ
*	$[q_1 q_2]$	ϕ	$[q_0 q_1 q_2]$
*	$[q_0 q_1 q_2]$	$[q_1 q_2]$	$[q_0 q_1 q_2]$
	ϕ	ϕ	ϕ

Step 7: $D = (\{[q_0], [q_1 q_2], [q_0 q_1 q_2]\}, \{a, b\}, \delta_D, [q_0], \{[q_1 q_2], [q_0 q_1 q_2]\})$

Eliminating ε-transitions:

- Let, $E = (Q_E, \Sigma, \delta_E, q_0, F_E)$, then equivalent DFA is $D = (Q_D, \Sigma, \delta_D, q_D, F_D)$ is defined as follows,

→ Q_D is set of subsets of Q_E such that $S \subseteq Q_E$ and $S = Eclose(S)$.

- → q_D is the start state of DFA, $q_D = Eclose(q_0)$.

→ F_D is set of all ~~strings~~ subsets S
i.e. $F_D = \{S \mid S \text{ is in } Q_D \text{ and } S \cap F_E \neq \phi\}$

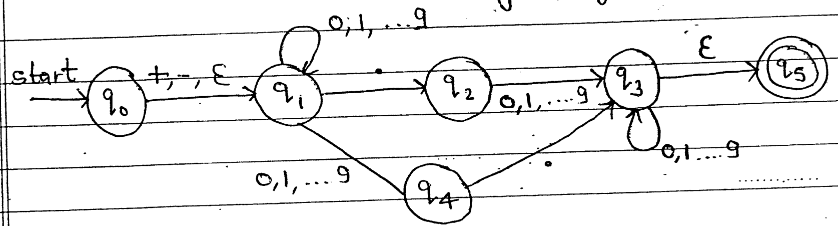
→ $\delta_D(S, a)$ is computed as

a) Let $S = \{p_1, p_2, \ldots p_k\}$

b) Compute $\bigcup_{i=1}^{k} \delta(p_i, a)$ let this set be $\{r_1, r_2 \ldots r_m\}$

c) $\delta_D(S, a) = \bigcup_{j=1}^{m} Eclose(r_j)$

→ Convert ε-NFA to DFA using Lazy Evaluation.



Step 1: Compute Eclose of all states:

$Eclose(q_0) = \{q_0, q_1\}$

$Eclose(q_1) = \{q_1\}$

$Eclose(q_2) = \{q_2\}$

$Eclose(q_3) = \{q_3, q_5\}$

$Eclose(q_4) = \{q_4\}$

$Eclose(q_5) = \{q_5\}$

Step 2: Start state of DFA is $Eclose(q_0) = [q_0 q_1]$ — ①

Step 3: $\delta_D([q_0 q_1]), 0,1,\dots 9) = Eclose(\delta_N(q_0, 0,1,\dots 9) \cup \delta_N(q_1, 0,1,\dots 9))$

$= Eclose(\{q_1, q_4\})$

$= [q_1 q_4]$ — ②

$\delta_D([q_0 q_1], .) = Eclose(\delta_N(q_0, .) \cup \delta_N(q_1, .))$

$= Eclose(q_2)$

$= [q_2]$ — ③

$\delta_D([q_0 q_1], +) = Eclose(\delta_N(q_0, +) \cup \delta_N(q_1, +))$

$= Eclose(q_0)$

$= [q_0 q_1]$

$\delta_D([q_0 q_1], -) = Eclose(\delta_N(q_0, -) \cup \delta_N(q_1, -))$

$= Eclose(q_0)$

$= [q_0 q_1]$

$\delta_D([q_1 q_4], 0,1,\dots 9) = Eclose(\delta_N(q_1, 0,1,\dots 9) \cup \delta_N(q_4, 0,1,\dots 9))$

$= Eclose(\{q_1, q_4\})$

$= [q_1 q_4]$

$\delta_D([q_1 q_4], .) = Eclose(\delta_N(q_1, .) \cup \delta_N(q_4, .))$

$= Eclose(\{q_2, q_3\})$

$= [q_2 q_3 q_5]$ — ④

$\delta_D([q_1 q_4], +) = Eclose(\delta_N(q_1, +) \cup \delta_N(q_4, +))$

$= Eclose(\{\phi\})$

$= \phi$ — ⑤

$\delta_D([q_2], 0,1,\dots 9) = Eclose(\delta_N(q_2, 0,1,\dots 9))$

$= Eclose(q_3) = [q_3 q_5]$ — ⑥

$\delta_D([q_2], .) = Eclose(\delta_N(q_2, .))$

$= \phi$

$\delta_D([q_2], +, -) = Eclose(\delta_N(q_2, +, -))$

$= \phi$

$\delta_D([q_2 q_3 q_5], 0,1,\dots 9) = Eclose(\delta_N(q_2, 0,1,\dots 9) \cup \delta_N(q_3, 0,1,\dots 9) \cup \delta_N(q_5, 0,1,\dots 9))$

$= Eclose(q_3) = [q_3 q_5]$

$\delta_D([q_2 q_3 q_5], .) = Eclose(\delta_N(q_2, .) \cup \delta_N(q_3, .) \cup \delta_N(q_5, .))$

$= Eclose(\phi) = \phi$

$\delta_D([q_2 q_3 q_5], +, -) = Eclose(\delta_N(q_2, +, -) \cup \delta_N(q_3, +, -) \cup \delta_N(q_5, +, -))$

$= Eclose(\phi) = \phi$

$\delta_D(\phi, 0,1\ldots 9) = \phi$

$\delta_D(\phi, \cdot) = \phi$

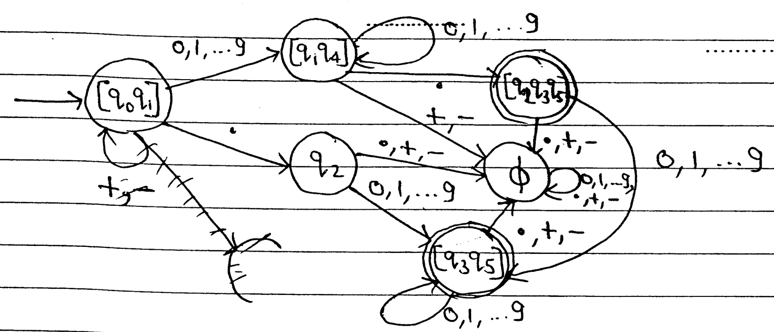
$\delta_D(\phi, +,-) = \phi$

$\delta_D([q_3q_5], 0,1\ldots 9) = \text{Eclose}(\delta_N(q_3, 0,1,\ldots 9) \cup \delta_N(q_5, 0,1,\ldots 9))$
$= \text{Eclose}(q_3) = [q_3q_5]$

$\delta_D([q_3q_5], \cdot) = \text{Eclose}(\delta_N(q_3, \cdot) \cup \delta_N(q_5, \cdot))$
$= \text{Eclose}(\phi) = \phi$

$\delta_D([q_3q_5], +,-) = \text{Eclose}(\delta_N(q_3, +,-) \cup \delta_N(q_5, +,-))$
$= \text{Eclose}(\phi) = \phi$

ep 4:



ep5:

δ	0,1,...9	$\cdot$	+,-
$\rightarrow [q_0q_1]$	$[q_1q_4]$	$[q_2]$	$[q_0q_1]$
$[q_1q_4]$	$[q_1q_4]$	$[q_2q_3q_5]$	ϕ
$[q_2]$	$[q_3q_5]$	ϕ	ϕ
$* [q_2q_3q_5]$	$[q_3q_5]$	ϕ	ϕ
$* [q_3q_5]$	$[q_3q_5]$	ϕ	ϕ
ϕ	ϕ	ϕ	ϕ

.6: $D = (\{[q_0q_1], [q_1q_4], [q_2], [q_2q_3q_5], [q_3q_5], \phi\}, \{0,1,\ldots 9, \cdot, +, -\}, \delta, [q_0q_1], \{[q_2q_3q_5], [q_3q_5]\})$

Theorems :

{Subset Construction}

1) If $D = (Q_D, \Sigma, \delta_D, \{q_0\}, F_D)$ is the DFA constructed from NFA $N = (Q_N, \Sigma, \delta_N, q_0, F_N)$ by subset construction then

$L(D) = L(N)$

$L(D) = \{w \mid \hat{\delta}_D(q_0, w) \text{ is in } F\}$

$L(N) = \{w \mid \hat{\delta}_N(q_0, w) \cap F \neq \phi\}$.

Proof: We proof first by induction on $|w|$ that

$\hat{\delta}_D(\{q_0\}, w) = \hat{\delta}_N(\{q_0\}, w)$

Notice that each of the $\hat{\delta}$ returns a set of states Q_N. But $\hat{\delta}_D$ interprets this as one of the states of Q_D. Which is the power set of Q_N while $\hat{\delta}_N$ interprets this as subset of Q_N.

Basis: Let, $|w| = 0$ i.e. $w = \varepsilon$

By the basis def. of $\hat{\delta}$ for both DFA's & NFA

$\hat{\delta}_D(\{q_0\}, \varepsilon) = \{q_0\}$

$\hat{\delta}_N(q_0, \varepsilon) = \{q_0\}$

Induction: Let, w be the length $n+1$ and assume that the statement is true for n length. Break w as $w = xa$ where a is the final symbol of w.

By inductive hypothesis;

$\hat{\delta}_D(\{q_0\}, w) = \hat{\delta}_N(q_0, w)$

Let, both these set of n states be $\{P_1, P_2, \ldots P_n\}$.
The inductive part for the definition of $\hat{\delta}$ for NFA is

$$\hat{\delta}_N(q_0, w) = \bigcup_{i=1}^{K} \delta_N(P_i, a) \quad — ①$$

The subset construction on the other hand tells us that

$$\delta_D(\{P_1, P_2 \ldots P_K\}, a) = \bigcup_{i=1}^{K} \delta_N(P_i, a) \quad — ②$$

Now, let us use eq^n ① and the fact that
$\hat{\delta}_D(\{q_0\}, x) = \{P_1, P_2 \ldots P_K\}$
in the inductive part of definition of $\hat{\delta}$ for DFA is

$$\hat{\delta}_D(\{q_0\}, w) = \delta_D(\hat{\delta}_D(\{q_0\}, x), a) = \delta_D(\{P_1, P_2 \ldots P_K\}, a)$$
$$= \bigcup_{i=1}^{K} \delta_N(P_i, a) \quad — ③$$

From eq^n ① & ③, it is clear that

$$\hat{\delta}_D(\{q_0\}, w) = \hat{\delta}_N(q_0, w)$$

When we observe that D and N both accept w if and only if $\hat{\delta}_D(\{q_0\}, w)$ or $\hat{\delta}_N(q_0, w)$ contain a state in F_N.

So, we have a complete proof that $L(D) = L(N)$.

2) A language L is accepted by some DFA if and only if L is accepted by some NFA.

If part is the subset construction and theorom 1.

Only if part :
Here, we have only to convert DFA into an identical NFA.
If we have the transition diag for DFA, we can also interpret as transition diag of NFA which happens to have one choice of transition in ~~equ~~ not ~~actionary~~ any situation.

More formally, Let $D = (Q_D, \Sigma, \delta_D, \{q_0\}, F_D)$ defines NFA, $N = (Q_N, \Sigma, \delta_N, q_0, F_N)$ where δ_N is defined by the rule
if $\delta_D(q, a) = P$
then $\delta_N(q, a) = \{P\}$

It is easy to show by induction on $|w|$ that

$$\hat{\delta}_D(q_0, \omega) = P \text{ then } \hat{\delta}_N(q_0, w) = \{P\}.$$

Unit - 2
Regular Expressions

* Operators of regular expressions -

1) Union of two languages
L & M , $L \cup M = \{001, 10, 11, \epsilon\}$
The union of two lang. L & M is denoted by $L \cup M$ and it is set of strings that are either in L or M or both. eg- $L = \{001, 10, 111\}$, $M = \{\epsilon, 001\}$

2) Concatenation of two languages -
$L.M = LM = \{001, 001001, 10, 10001, 111, 111001\}$
It is denoted by LM and it is set of strings that can be formed by taking any string L and concatenating with M.

3) Closure (Kleen / star) -
L^*
It is denoted by L^* and it represents the set of those strings that can be formed by taking ~~those~~ any no. of strings from possible with repitition and concatenating all of them.

$L = \{0, 1\}$
$L^* = \{\epsilon, 011, 001, 0, 1, 01, 10, 00, 11, \ldots\}$

→ Find L^* for $L = \{0, 11\}$.
$L^* = \{\epsilon, 0, 11, 011, 110, 00, 1111, 000, \ldots\}$

* Building regular expressions :

1) The constants ϵ & ϕ are regular expressions.
2) If 'a' is a symbol then 'a' is a regular expression
$L(a) = \{a\}$.
3) E & F are regular expressions then E + F is also a regular expression, L(E) & L(F)

$$L(E+F) = L(E) \cup L(F)$$

4) If E and F are two regular expression then E.F is also a regular expression denoting L(E) and L(F).
$L(E.F) = L(E).L(F)$
eg- $L(E) = \{\epsilon\}$
$L(F) = \{a\}$
$L(E.F) = \{a\}$.

5) If E is a regular expression then E^* is also a regular expression denoting closure of E.

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

$E = a$ $L(a) = \{a\}$ $L(E^*) = \{\epsilon, a, aa, aaa, \ldots\}$

6) If E is a regular expression then (E) is also a regular expression denoting

$$L((E)) = L(E)$$

→ Write the regular expression for set of all possible strings over $\Sigma = \{0, 1\}$.

$L = \{\epsilon, 0, 1, 00, 01, \ldots\}$
The regular expression is -

R.E. = $(0+1)^*$

→ .... of 0's & 1's begining with 0 & ending with 1.

L = { 01 , 001 , 011 , 0001, 0101, 0111, ... }

R.E. = $0.(0+1)^*.1$

→ .... containing either an arbitary no. of 0's or an arbitary no. of 1's.

L = {ε, 0 , 1 , 00, 000, 111, 0000, ... }

R.E. = $(0)^* + (1)^*$

→ ..... ending with 011.

L = { 011, 1011, 0011, 00011, 10011, ---}.

R.E. = $(0+1)^*.(011)$

→ ..... strings of 0's, 1's & 2's with any no. of 0's followed by any of 1's followed by any no. of 2's.

L = { ε, 0 , 01 , 012 , ... }

R.E. = $(0)^*.(1)^*.(2)^*$

→ .... strings of 0's & 1's in which all the 0's if any comes before all the 1's if any.

L = { ε, 0, 1, 01, 001, 0011, ... }

R.E. = $(0)^*.(1)^*$

→ .... strings of a's & b's that have atleast 2 letters that begin & end with 'a' and have nothing other than b's inside.

L = { aa, aba, abba, ...}

R.E. = $a.(b)^*.a$

→ .... strings of 0's & 1's with atleast 2 consecutive zeroes.

L = { 00, 001, 1001, 0000, 0001 ...}

R.E. = $(0+1)^*(00)^*(0+1)^*$

→ ... 0's & 1's that have either a single '0' followed by any no. of 1's or a single '1' followed any no. of 0's.

L = { 01, 0, 10, 1, 100, ... }

R.E. : $(0.(1)^*)+(1.(0)^*)$

→ ... that begin with 110

L = { 110, 1101, 1100 ... }

R.E. : $(110)(0+1)^*$

papergrid

Date: / /

→ .... that exactly contain 3 one's-

$L = \{111, 010101, 10101, \ldots\}$

R.E. : $\cancel{(0)^*111(0)^*} + \underbrace{(0)^*1(0)^*1(0)^*1(0)^*} + \cancel{(0)^*11(0)^*} +$

$0^*10^*10^*10^*$

$\cancel{(0)^* 11(0)^*1(0)^*} + \cancel{0^*(1)0^*11(0)^*}$

→ .... which are of even length.

$L = \{\varepsilon, 00, 01, 11, 01, \ldots\}$.

R.E. $= \cancel{(00)^*(11)^*} \; (00+01+10+11)^*$

$= [(0+1)(0+1)]^*$

→ .... for set of all strings of 0's & 1's such that no. of 0's is odd.

$L = \{0, 0100, 01, 011, 1011 \ldots\}$.

R.E. $= 0(\cancel{0+1})(1^*0.1^*).(0\,1^*\,0)^*$

→ ... even binary nos. -

$L = \{0, 10, 100, 110 \ldots\}$

R.E. $= (1+0)^*0$

1. ... 0's & 1's that begin with 1 and do not have 2 consecutive 0's.
2. ... 0's & 1's that do not have two consecutive zeroes
3. ... 0's & 1's with exactly 1 pair of consecutive 0's.

3. R.E $= (1+01)^*.(00).(1+10)^*$

papergrid

Date: / /

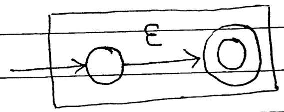
2. R.E $= (11^*0)^*$

* Precedence of RE operators:-
  1. star (*)
  2. Concatenation (.)
  3. Union (+)

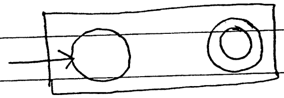
Identities of RE -

1) $\phi + r = r$
2) $\phi . r = \phi$
3) $\varepsilon . r = r.\varepsilon = r$
4) $\varepsilon^* = \{\varepsilon\}$
5) $\phi^* = \cancel{\{\}} = \phi\ \varepsilon$
6) $r + r = r$
7) $r^*.r^* = r^*$
8) $r.r^* = r^*$
9) $(r^*)^* = r^*$
10) $\varepsilon + r.r^* = r^*$
11) $(p+q)^* = (p^*.q^*)^* = (p^*+q^*)^*$
12) $(p+q)r = p.r + q.r$
13) $r(p+q) = r.p + r.q$

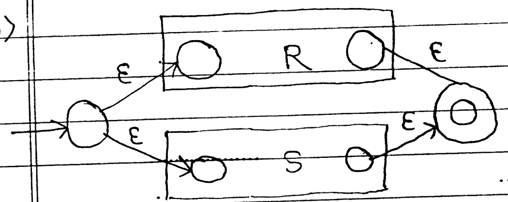
* Regular Expression to Finite Automata :

1) 

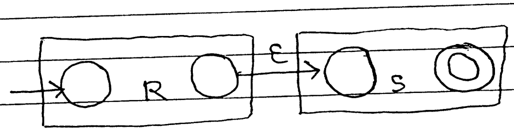
RE = ε

2) 

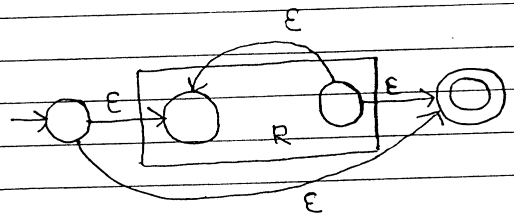
RE = φ

3) 

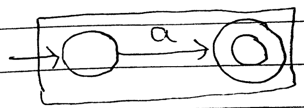
RE = R+S

4) 

RE = R.S

5) 

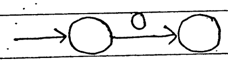
RE = R^*

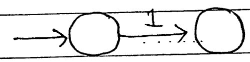
6) 

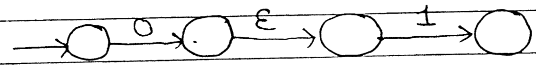
RE = a

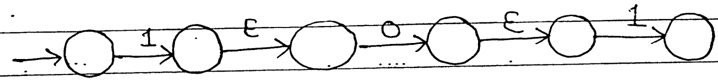
→ Convert the following R.E. to equivalent finite automata.

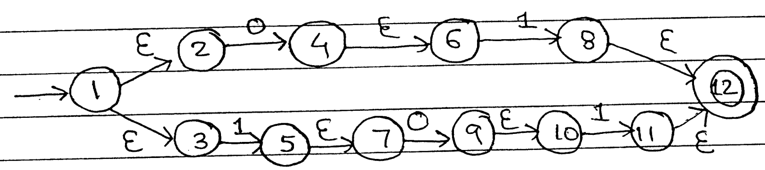
1) 01 + 101

a. 

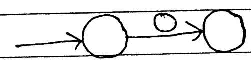
b. 

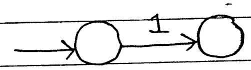
c. 

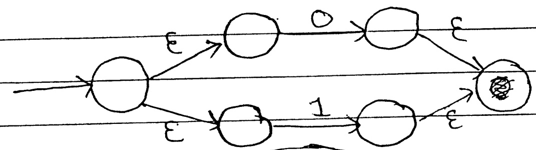
d. 

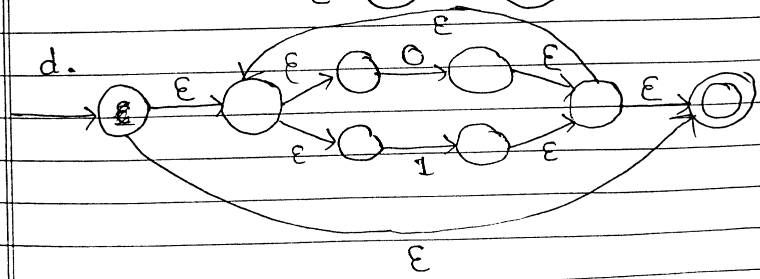
e. 

2) RE = $(0+1)^*$

a. 

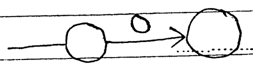
b. 

c. 

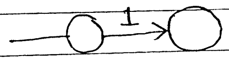
d. 

3) RE = $(0+1)^* 1 (0+1)$

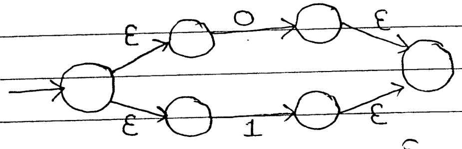
a.



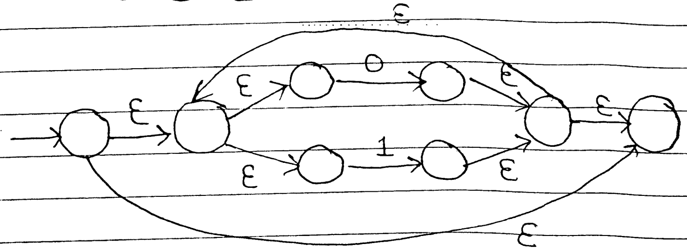
b.



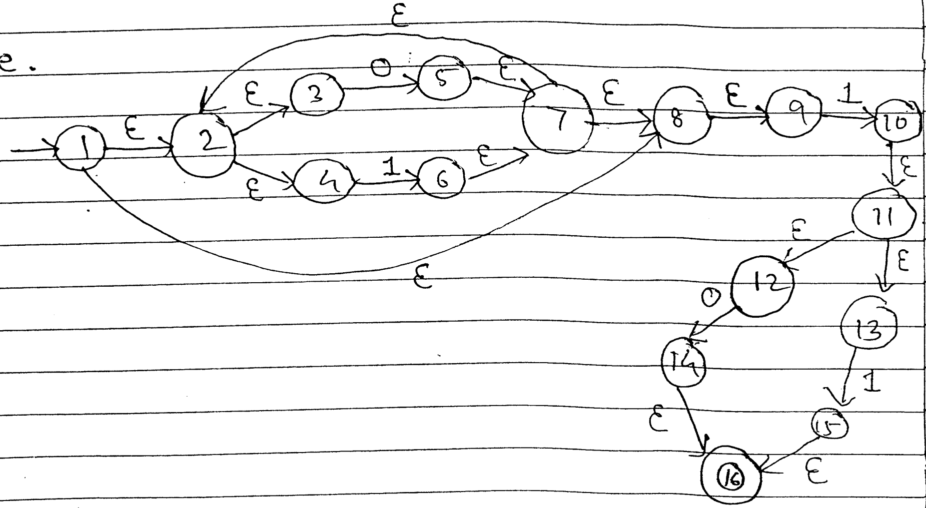
c.



d.

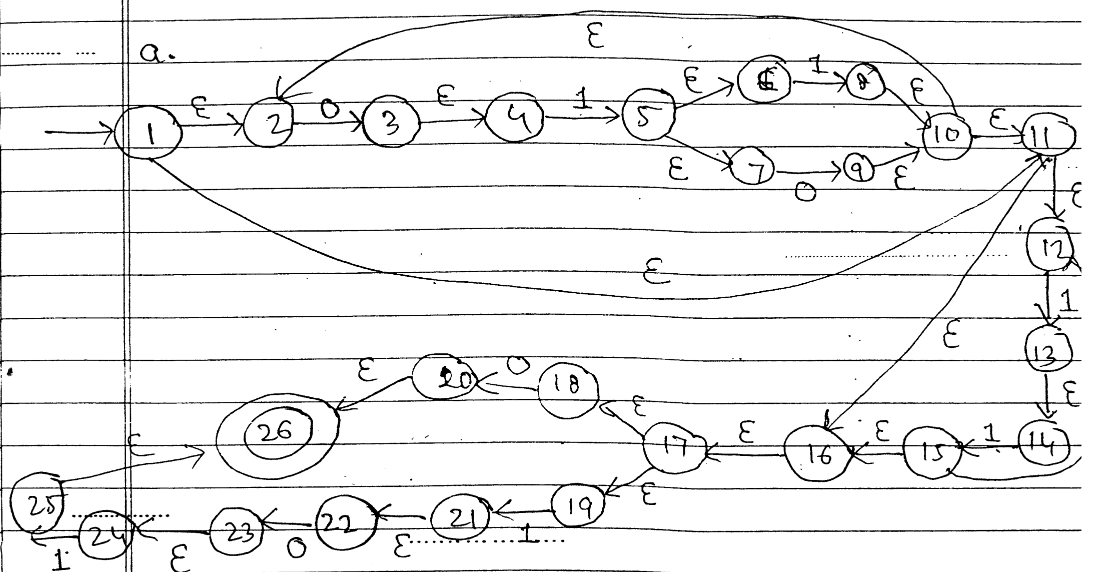


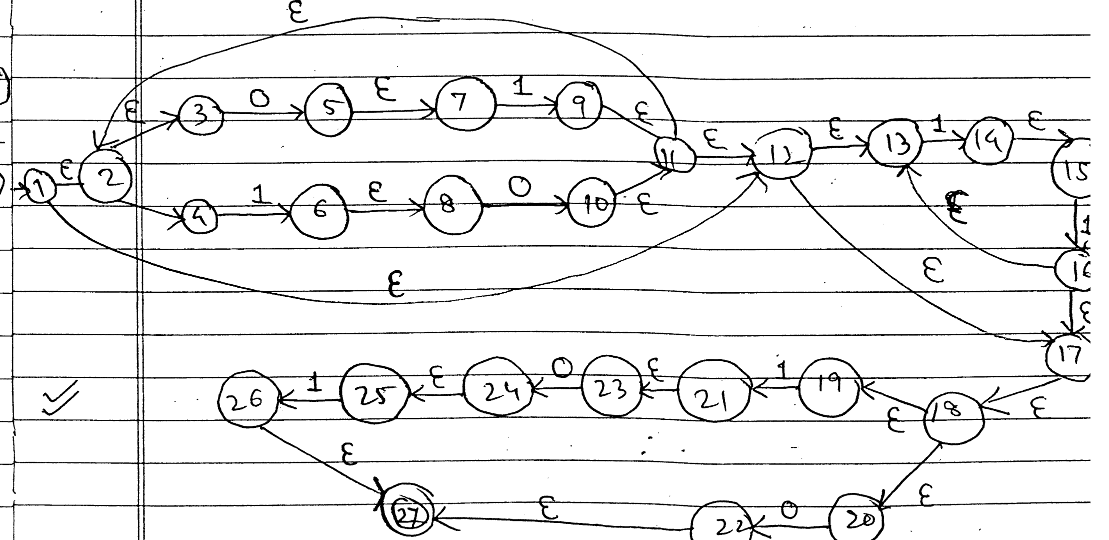
e.



4) RE = $(01+10)^* (1 \cdot 1)^* (101+0)$

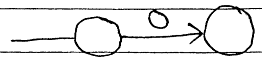
a.



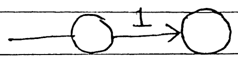


3) RE = $(0+1)^* 1 (0+1)$

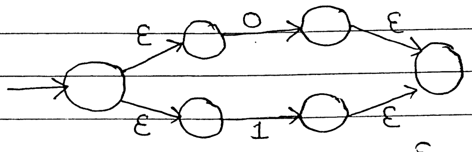
a.



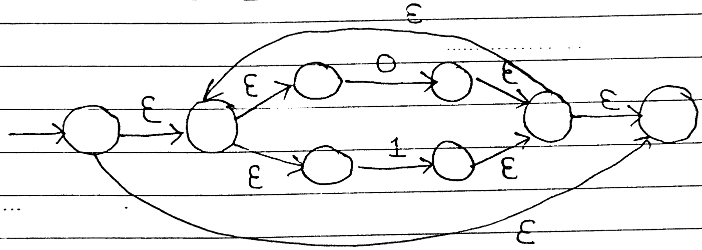
b.



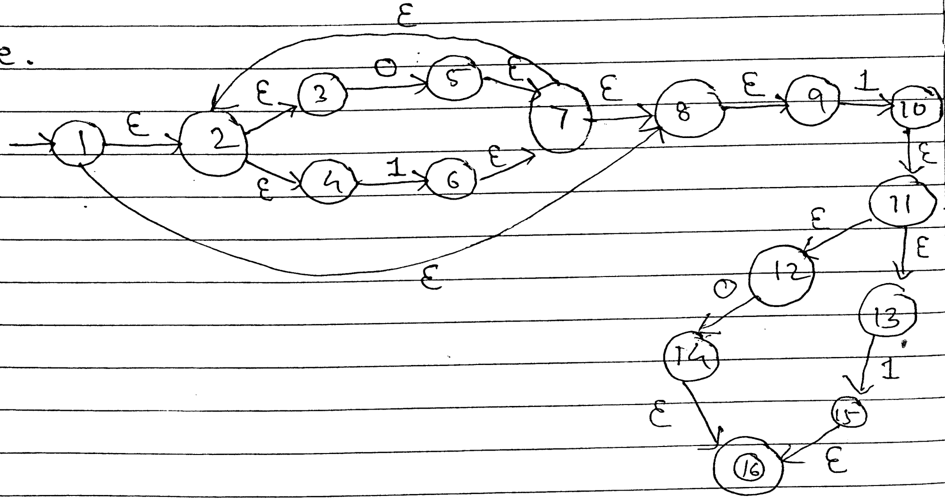
c.



d.

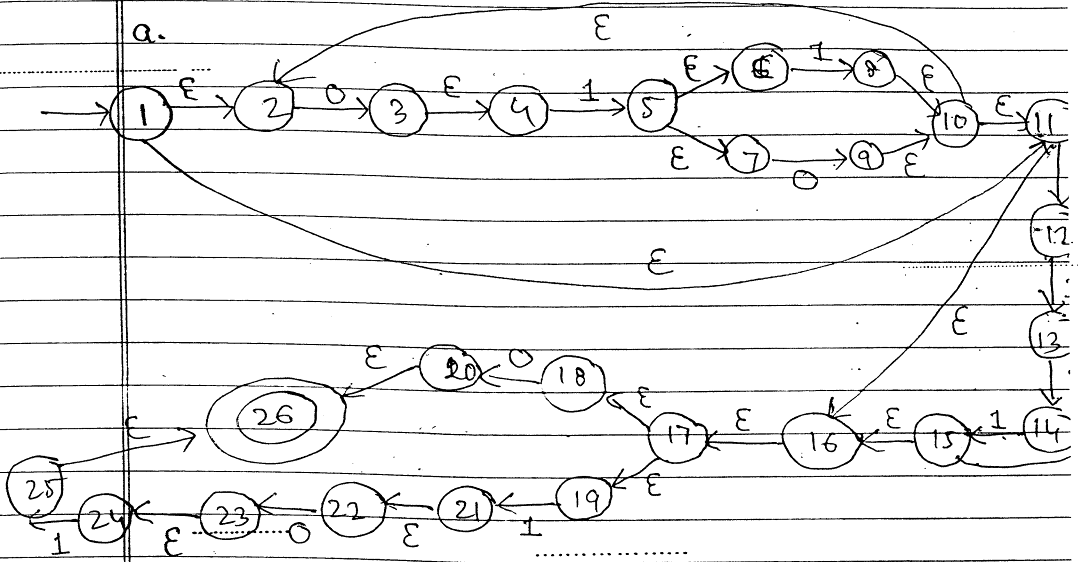


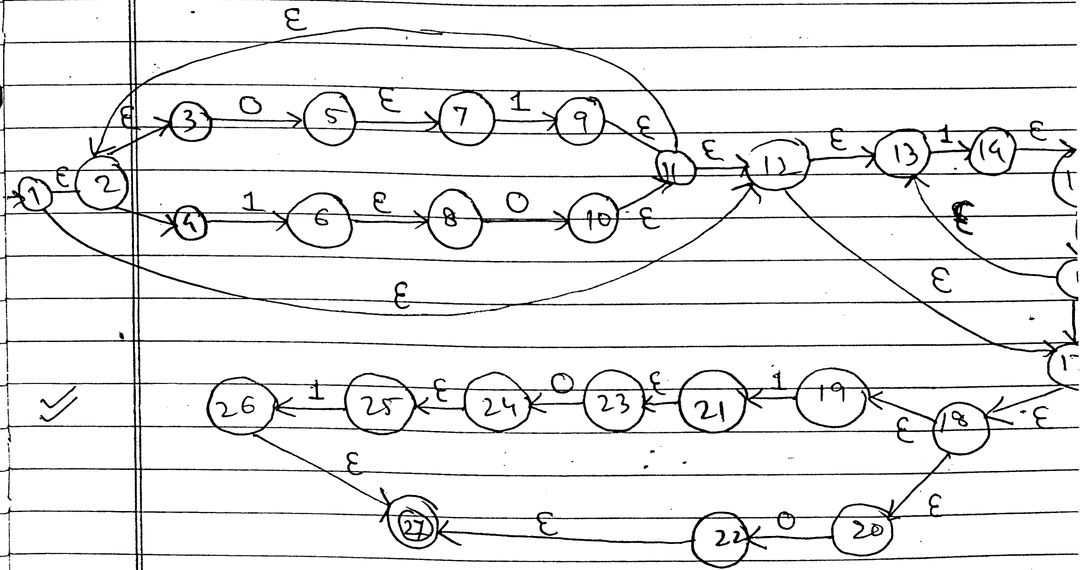
e.



4) RE = $(01+10)^* (1.1)^* (101+0)$

a.



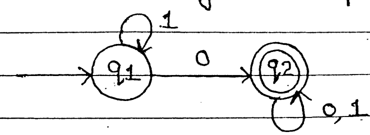


papergrid

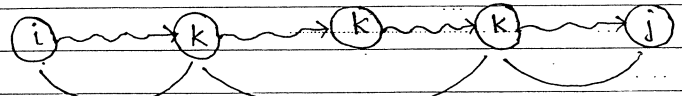
09/10/18

Date: / /

→ DFA to Regular Expression :-

Q. 

: RE = ?



$R_{ik}^{(k-1)}$ zero/more strings in $R_{kk}^{(k-1)}$ $R_{kj}^{(k-1)}$

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

- $R_{ij}^{(k)}$ is the regular expression for the lang. of some string ω such that 'ω' is the label of path from state 'i' to state 'j' with no intermediate node, whose no. is $\cancel{\geq}$ > k.

- Final regular expression is expressed as:
  $R_{1j}^{(n)}$ where 1 is start state & j is accepting state.

1. Convert the above DFA to its equivalent R.E.

	0	1	2
R_{11}	$1+\varepsilon$	1^*	1^*
R_{12}	0	$\cancel{0+}1^*.0$	$1^*.0(0+1)^*$
R_{21}	ϕ	ϕ	X
R_{22}	$\varepsilon+0+1$	$\varepsilon+0+1$	X

$R_{12}^{(2)}$: max:

$R_{11}^{(0)} = 1+\varepsilon$, $R_{12}^{(0)} = 0$ $R_{21}^{(0)} = \phi$, $R_{22}^{(0)} = (\varepsilon+0+1)$

$$R_{11}^{(1)} = R_{11}^{(0)} + R_{11}^{(0)} \cdot \left(R_{11}^{(0)}\right)^* \cdot R_{11}^{(0)}$$
$$= (1+\varepsilon) + (1+\varepsilon)(1+\varepsilon)^*(1+\varepsilon)$$

papergrid

Date: / /

$$= (1+\varepsilon) + 1^* \qquad \{1 \cdot 1^* \cdot 1 = 1^*\}$$
$$= 1^*$$

$$R_{12}^{(1)} = R_{12}^{(0)} + R_{11}^{(0)} \cdot \left(R_{11}^{(0)}\right)^* \cdot R_{12}^{(0)}$$
$$= 0 + (1+\varepsilon)(1+\varepsilon)^* \cdot 0$$
$$= 0 + 1^* . 0$$
$$= 1^* . 0$$

$$R_{21}^{(1)} = R_{21}^{(0)} + R_{21}^{(0)} \cdot \left(R_{11}^{(0)}\right)^* \cdot R_{11}^{(0)}$$
$$= \phi + \phi \cdot (1+\varepsilon)^*(1+\varepsilon)$$
$$= \phi + \cancel{\phi^*}\phi$$
$$= \phi$$

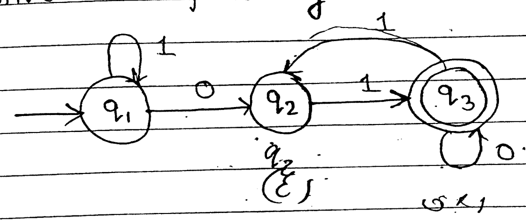
$$R_{22}^{(1)} = R_{22}^{(0)} + R_{21}^{(0)} \cdot \left(R_{11}^{(0)}\right)^* R_{12}^{(0)}$$
$$= (\varepsilon+0+1) + \phi \cdot (1+\varepsilon)^* . 0$$
$$= \varepsilon+0+1$$

$$R_{11}^{(2)} = R_{11}^{(1)} + R_{12}^{(1)} \cdot \left(R_{22}^{(1)}\right)^* \cdot R_{21}^{(1)}$$
$$= 1^* + 1^* . 0 \, (\varepsilon+0+1)^* . \phi$$
$$= 1^*$$

$$R_{12}^{(2)} = R_{12}^{(1)} + R_{12}^{(1)} \cdot \left(R_{22}^{(1)}\right)^* \cdot R_{22}^{(1)}$$
$$= 1^*.0 + 1^*.0.(\varepsilon+0+1)^*(\varepsilon+0+1)$$
$$= 1^*.0 + \cancel{1^*.0.1^*}(0+1)^*$$
$$= \cancel{1^*.0.1^*} \; 1^*.0.(0+1)^*$$

Final R.E. is $R_{12}^{(2)} = 1^*.0.(0+1)^*$

2. Convert the following automata to R.E.:



	0	1	2	3
R_{11}	$(1+\varepsilon)$	1^*	1^*	1^*
R_{12}	0	$1^*.0$	$1^*.0$	$1^*.0 + 1^*01(0+11)^*.1$
R_{13}	ϕ	ϕ	$1^*.0.1$	$1^*.0.1(0+11)^*$
R_{21}	ϕ	ϕ	ϕ	X
R_{22}	$\cancel{\phi}\varepsilon$	$\cancel{\phi}\varepsilon$	ε	X
R_{23}	1	$\cancel{\phi}1$	$1.\cancel{\phi}$	X
R_{31}	ϕ	ϕ	ϕ	X
R_{32}	1	$\cancel{\phi}1$	$\cancel{\phi}1.$	X
R_{33}	$(0+\varepsilon)$	$(0+\varepsilon)$	$(0+\varepsilon)+1.1$	X

$$R_{13}^{(1)} = R_{13}^{(0)} + R_{11}^{(0)}.(R_{11}^{0})^*.R_{13}^{(0)}$$
$$= \phi + (1+\varepsilon)(1+\varepsilon)^*\phi$$
$$= \phi$$

$$R_{32}^{(1)} = R_{32}^{(0)} + R_{31}^{(0)}.(R_{11}^{0})^*.R_{12}^{(0)}$$
$$= 1 + \phi$$
$$= 1$$

$$R_{33}^{(1)} = R_{33}^{(0)} + R_{31}^{(0)}.(R_{11}^{(0)})^*.R_{13}^{(0)}$$
$$= (0+\varepsilon) + \phi$$
$$= (0+\varepsilon)$$

$$R_{12}^{(2)} = (R_{12})^{1} + R_{12}^{(1)}.(R_{22}^{(1)})^*.R_{22}^{(1)}$$
$$= 1^*.0 + 1^*.0.(\varepsilon)^*.\varepsilon$$
$$= 1^*.0$$

$$R_{13}^{(2)} = R_{13}^{(1)} + R_{12}^{(1)}(R_{22}^{(1)})^* R_{23}^{(1)}$$
$$= \phi + 1^*.0.\varepsilon^*.\cancel{\varepsilon}1$$
$$= 1^*.0.1$$

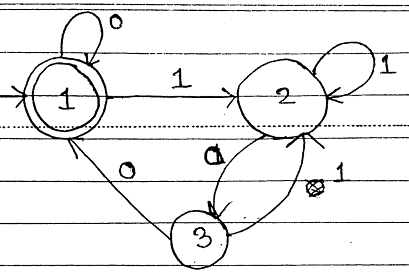
$$R_{21}^{(2)} = R_{21}^{(1)} + R_{22}^{(1)}(R_{22}^{(1)})^* R_{21}^{(1)}$$
$$= \phi + \varepsilon.\varepsilon^*.\phi$$
$$= \phi$$

$$R_{13}^{(3)} = R_{13}^{(2)} + R_{13}^{(2)}(R_{33}^{(2)})^*.R_{33}^{(2)}$$
$$= 1^*.0.1 + 1^*.0.1(0^*).0.$$
$$= 1^*.0.1 + 1^*.0.1.0^*$$
$$= 1^*.0.10^*$$

$$R_{32}^{(2)} = R_{32}^{(1)} + R_{32}^{(1)}(R_{22}^{(1)})^*(R_{22}^{(1)})$$
$$= 1 + 1(\varepsilon)^*\varepsilon.$$
$$= 1$$

$$R_{23}^{(2)} = R_{23}^{(1)} + R_{22}^{(1)}(R_{22}^{(1)})^*(R_{23}^{(1)})$$
$$= 1 + \varepsilon.\varepsilon^*.1.$$
$$= 1$$

3.



	0	1	2	3
R_{11}	$(0+\epsilon)$	0^*	0^*	$0^*.1^*.0(\epsilon+0^*.1^*.0)^*0^*$
R_{12}	1	$0^*.1$	$0^*.1^*$	$0^*.1^*.0(\epsilon+0^*.1^*.0)^*.0^*.1^*$
R_{13}	ϕ	ϕ	$0^*.1^*.0$	
R_{21}	ϕ	ϕ	ϕ	X
R_{22}	$(1+\epsilon)$	$1+\epsilon$	$(1+\epsilon)^*=1^*$	X
R_{23}	0	0	$1^*.0$	X
R_{31}	0	0^*	0^*	X
R_{32}	1	$0^*.1$	$0^*.1^*$	X
R_{33}	ϵ	ϵ	$\epsilon+0^*.1.0$	X

$$R_{11}^{(1)} = R_{11}^{(0)} + R_{11}^{(0)}.\left(R_{11}^{(0)}\right)^* R_{11}^{(0)}$$
$$= (0+\epsilon) + (0+\epsilon).(0+\epsilon)^*.(0+\epsilon)$$
$$= (0+\epsilon)^* = 0^*$$

$$R_{12}^{(1)} = R_{12}^{(0)} + R_{11}^{(0)}\left(R_{11}^{(0)}\right)^* R_{12}^{(0)}$$
$$= 1 + (0+\epsilon)(0+\epsilon)^*.1$$
$$= 1 + 0^*.1$$
$$= 0^*.1$$

$$R_{13}^{(1)} = R_{13}^{(0)} + R_{11}^{(0)}\left(R_{11}^{(0)}\right)^* R_{13}^{(0)}$$
$$= \phi + (0+\epsilon)(0+\epsilon)^*.\phi$$
$$= \phi$$

papergrid
Date: / /

$$R_{21}^{(1)} = R_{21}^{(0)} + R_{21}^{(0)}\left(R_{11}^{(0)}\right)^* R_{11}^{(0)}$$
$$= \phi + \phi(0+\epsilon)^*(0+\epsilon)$$
$$= \phi$$

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

$$R_{23}^{(1)} = R_{23}^{(0)} + R_{21}^{(0)}\left(R_{11}^{(0)}\right)^* R_{13}^{(0)}$$
$$= 0 + \phi(0+\epsilon)^*.\phi$$
$$= 0$$

$$R_{31}^{(1)} = R_{31}^{(0)} + R_{31}^{(0)}\left(R_{11}^{(0)}\right)^* R_{11}^{(0)}$$
$$= 0 + 0.(0+\epsilon)^*(0+\epsilon)$$
$$= 0 + 0^*$$
$$= 0^*$$

$$R_{32}^{(1)} = R_{32}^{(0)} + R_{31}^{(0)}\left(R_{11}^{(0)}\right)^* R_{12}^{(0)}$$
$$= 1 + 0.(0+\epsilon)^* 1.$$
$$= 1 + 0^*.1$$
$$= 0^*.1$$

$$R_{33}^{(1)} = R_{33}^{(0)} + R_{31}^{(0)}\left(R_{11}^{(0)}\right)^* R_{13}^{(0)}$$
$$= \epsilon + 0.(0+\epsilon)^*\phi$$
$$= \epsilon$$

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

$$R_{12}^{(2)} = 0^*.1 + 0^*.1(1+\epsilon)^*(1+\epsilon)$$
$$= 0^*.1 + 0^*.1^* = 0^*.1^*$$

$R_{13}^{(2)} = R_{13}^{(1)} + R_{12}^{(1)} (R_{22}^{(1)})^* . R_{23}^{(1)}$
$= \phi + 0^*.1(1+\varepsilon)^*.0$
$= 0^*.1^*.0$

$R_{21}^{(2)} = R_{21}^{(1)} + R_{22}^{(1)} (R_{22}^{(1)})^* . R_{21}^{(1)}$
$= \phi + (1+\varepsilon)(1+\varepsilon)^*.\phi$
$= \phi$

$R_{22}^{(2)} = R_{22}^{(1)} + R_{22}^{(1)} (R_{22}^{(1)})^* R_{22}^{(1)}$
$= (1+\varepsilon) + (1+\varepsilon)^*$
$= (1+\varepsilon)^* = 1^*$

$R_{23}^{(2)} = R_{23}^{(1)} + R_{22}^{(1)} (R_{22}^{(1)})^* R_{23}^{(1)}$
$= 0 + (1+\varepsilon)(1+\varepsilon)^*.0$
$= 0 + (1+\varepsilon)^*.0$
$= 1^*.0$

$R_{31}^{(2)} = R_{31}^{(1)} + R_{32}^{(1)} (R_{22}^{(1)})^* R_{21}^{(1)}$
$= 0^* + 0^*.1(1+\varepsilon)^*.\phi$
$= 0^*$

$R_{32}^{(2)} = 0^*.1 + 0^*.1(1+\varepsilon)^*.(1+\varepsilon)$
$= 0^*.1 + 0^*.1^*$
$= 0^*.1^*$

$R_{33}^{(2)} = \varepsilon + 0^*.1(1+\varepsilon)^*.0$
$= \varepsilon + 0^*.1^*.0$

#

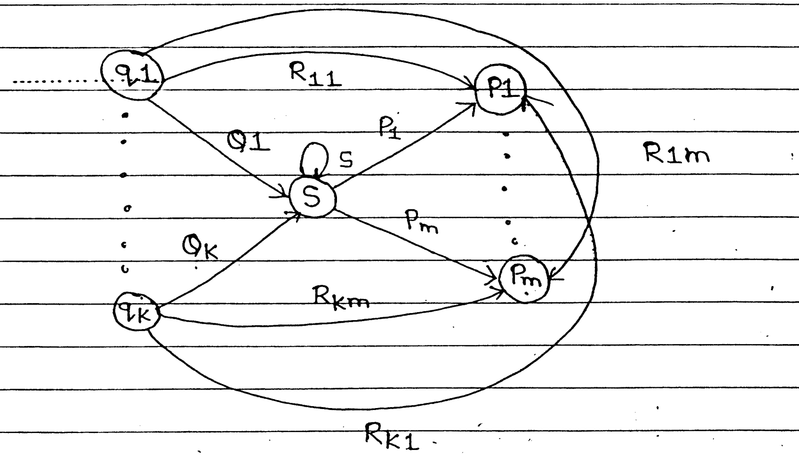
$R_{11}^{(3)} = 0^* + 0^*.1^*.0(\varepsilon+0^*.1^*.0)^*.0^*$
$= 0^*.1^*.0(\varepsilon+0^*.1^*.0)^*0^*$

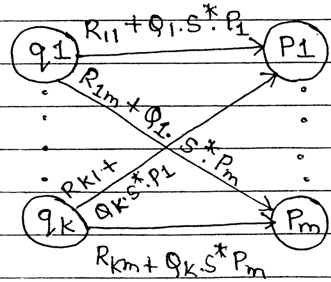
$R_{12}^{(3)} = R_{12}^{(2)} + R_{13}^{(2)} (R_{33}^{(2)})^* R_{32}^{(2)}$
$= 0^*.1^* + 0^*.1^*.0(\varepsilon+0^*.1^*.0)^* 0^*.1^*$
$= 0^*.1^*.0(\varepsilon+0^*.1^*.0)^*.0^*.1^*$

$R_{13}^{(3)} = R_{13}^{(2)} + R_{13}^{(2)} (R_{33}^{(2)})^* . R_{33}^{(2)}$
$= 0^*.1^*.0 + 0^*.1^*.0(\varepsilon+0^*.1^*.0)^*.(\varepsilon+0^*.1^*.0)$
$= 0^*.1^*.0(\varepsilon+0^*.1^*.0)^*$

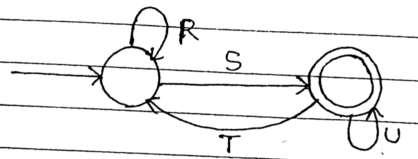
Final R.E is $R_{13}^{(3)} = 0^*1^*.0(\varepsilon+0^*.1^*.0)^*$

→ DFA to R.E. using state elimination method:



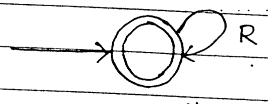


Two state Automaton :



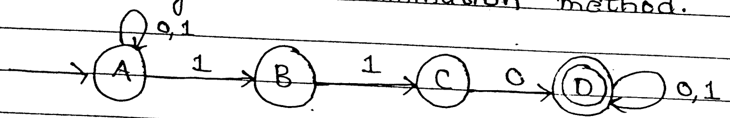
$$R.E. = (R + SU^*T)^* S U^*$$

One state Automaton:

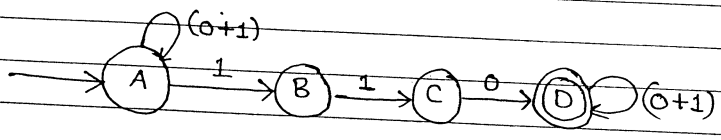


$$R.E. = R^*$$

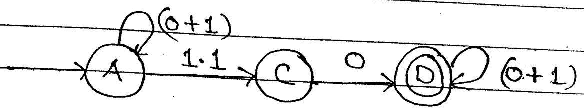
1. Convert the following finite automata to its equivalent R.E. using state elimination method.



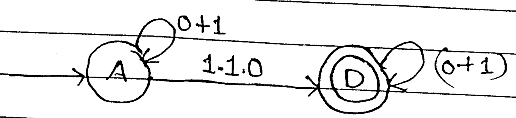
Solⁿ: Automaton with RE.



Eliminate B :



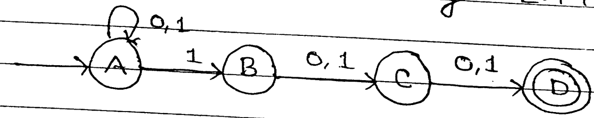
Eliminate C :



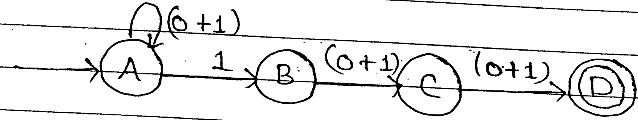
$$R.E. \Rightarrow \quad R = (0+1)$$
$$S = 110$$
$$T = \phi$$
$$U = (0+1)$$

$$R.E. = \left((0+1) + 110(0+1)^*.\phi\right)^* 110.(0+1)^*$$
$$= (0+1)^*.110.(0+1)^*$$

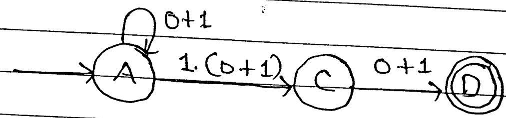
2. Convert F.A. to R.E. using S.E.M.



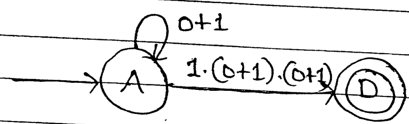
Solⁿ: Automaton with RE.



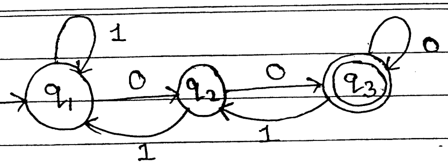
Eliminate B.



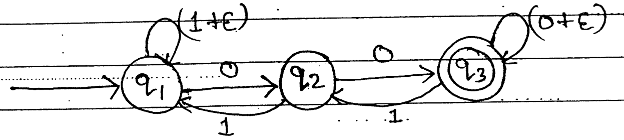
Eliminate C :



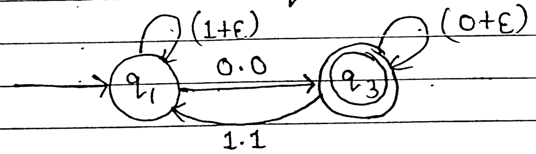
$$R.E. = \left((0+1) + (1.(0+1).(0+1).\phi^*.\phi)\right)^* . 1(0+1)(0+1).\phi^*$$
$$= (0+1)^*. 1.(0+1).(0+1)\,\varepsilon$$

3. 

Solⁿ: Automaton with R.E.



Eliminate q2.

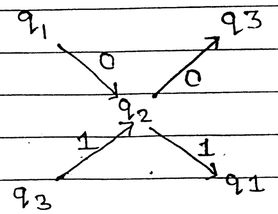


$$\text{R.E.} = \left((1+\varepsilon) + (0.0.(0+\varepsilon)^*.1.1\right)^* . 0.0.(0+\varepsilon)^*$$

$$= \left(1 + (0^*.1.1)\right)^* . 0^*$$

$\neq \emptyset$

Solⁿ: Eliminating q2

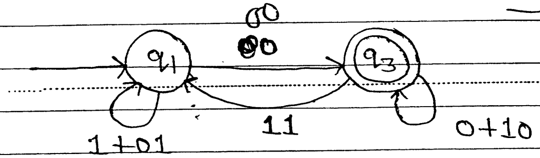


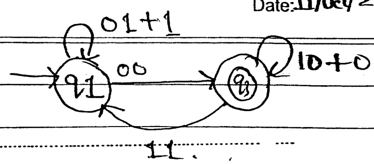
$q1 \xrightarrow{01} q_1$

$q1 \xrightarrow{00} q3$

$q3 \xrightarrow{10} q3$

$q3 \xrightarrow{11} q1$





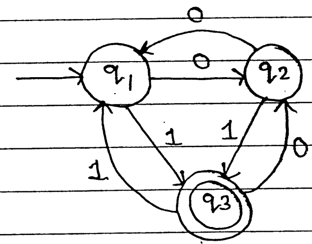
$R = 1+01$

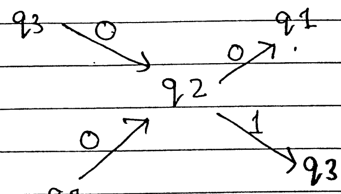
$S = 00$

$T = 11$

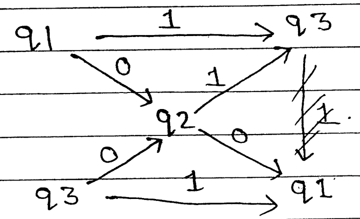
$U = 0+10$

$$\text{R.E.} = \left(1+01 + 00(0+10)^*11\right)^* 00(0+10)^*$$

4. 



Eliminating q2:

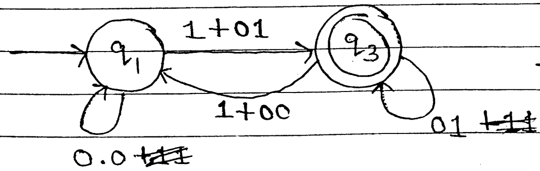


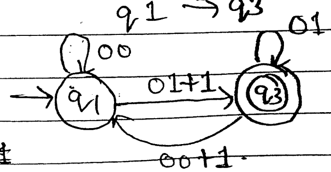
$q3 \xrightarrow{00} q1$

$q3 \xrightarrow{01} q3$

$q1 \xrightarrow{00} q1$

$q1 \xrightarrow{01} q3$

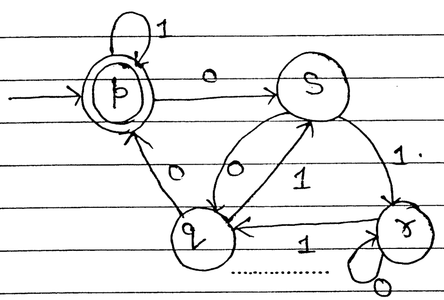




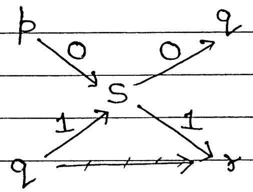
$$\text{R.E.} = \left((0.0 + (1+01)(0.1)^*(1+00)\right)^* (1+01)(01)^*$$

5. Convert DFA to R.E. using S.E.M.

δ	0	1
→* p	s	p
q	p	s
r	r	q
s	q	r



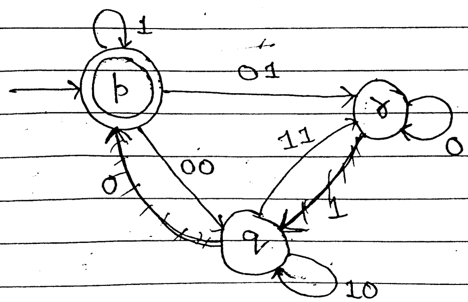
Solⁿ: Eliminate s



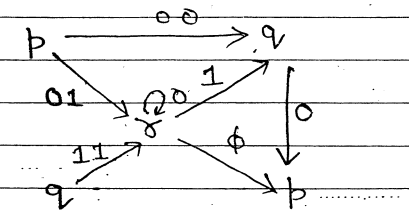
$p \xrightarrow{00} q$ $\quad q \xrightarrow{11} r$

$q \xrightarrow{10} q$

$p \xrightarrow{01} r$



Eliminate r.



$p \xrightarrow{\phi} p$

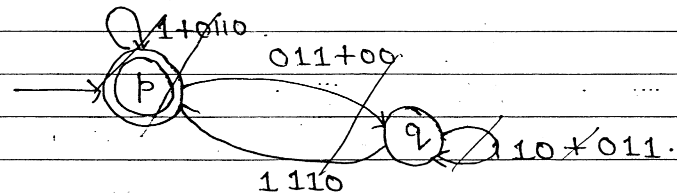
$p \xrightarrow{010^*1} q$

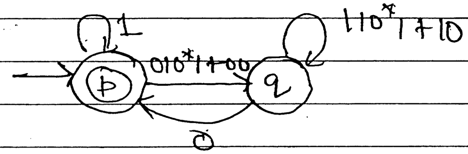
$q \xrightarrow{110^*1} q$

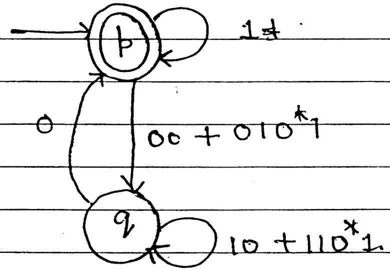
$q \xrightarrow{\phi} p$

$p \xrightarrow{\cancel{0110}\ 1+\phi} p$, $q \xrightarrow{10+110^*1} q$, $p \xrightarrow{00} q$

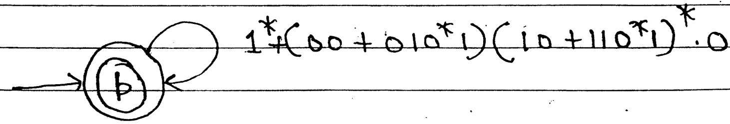
$p \xrightarrow{\cancel{0011}\ 010^*1} q$, $q \xrightarrow{\cancel{1110}\ 0} p$







Eliminate q.



R.E. = $((1^* (00+010^*1)(10+110^*1)^* \cdot 0)^*)^*$

1. R.E., $\Sigma(a,b)$ whose length is multiple of 3.
$L = \{\varepsilon, aaa, bbb, aab, aba, bba, \ldots\}$

R.E. $= \left((a+b)^{\cancel{*}}(a+b)^{\cancel{*}}(a+b)^{\cancel{*}}\right)^*$

2. $\Sigma(0,1)$ with exactly one pair of consecutive zeroes.

R.E. $= \cancel{\phi \cdot \phi 1001^* \phi}\ (1+01)^* 00 (1+10)^*$

3. $\Sigma(a,b)$ containing both aa & bab as substring.

R.E. $= \cancel{a^* aab^* bab}$
$(a+b)^* aa (a+b)^* bab (a+b)^* +$

$(a+b)^* bab (a+b)^* aa (a+b)^*$

4. $\Sigma(0,1)$ with odd no. of zone's followed by even no. of zeroes.

$L = \{ \cancel{0, 0}\ 1, 100, 10000$

R.E.: $\cancel{1 + 1.(00)^*}\ (11)^* . 1 (00)^* . \cancel{00}$

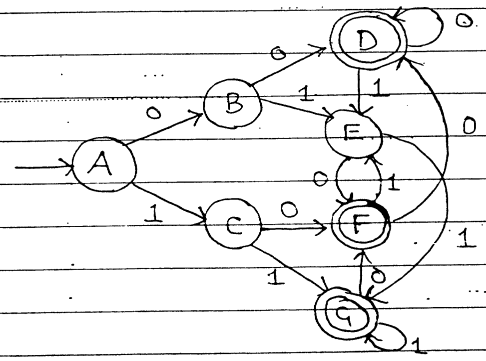
5. $\Sigma(a,b)$ such that:
$L = \{a^m b^n \mid m+n \text{ is even}\}$.

$L = \{\varepsilon, ab, aabb, abb, \ldots$

R.E. $= \cancel{a^*}\left((aa)^* . \cancel{a}(bb)^* + (aa)^* . a (bb)^* . b\right)^*$

Equivalence and minimization of finite Automata

→ Testing equivalence of states



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

Testing the equivalence of states:
When 2 distinct states p & q can be replaced by a single state that behaves like both p & q, we say that states p & q are equivalent. Also if for all strings (i/p) ~~is~~ on w then, $\hat{\delta}(p,w)$ is an accepting state if and only if $\hat{\delta}(q,w)$ is an accepting state.

Table - filling algo -

B	X					
C	X	X				
* D	X	X	X			
E	X	X	✓	X		
x F	X	X	X	✓	X	
x G	X	X	X	X	X	X
	A	B	C	D	E	F

$\delta(D,G) \xrightarrow{0} DF$, $\xrightarrow{1} EG$ x

$\delta(C,E) \xrightarrow{0} FF$ ✓, $\xrightarrow{1} GG$ ✓

$\delta(D,F) \xrightarrow{0} DD$, $\xrightarrow{1} EE$ ✓

$\delta(F,G) \xrightarrow{0} DF$, $\xrightarrow{1} EG$ x

$\delta(A,E) \xrightarrow{0} BF$ x, $\xrightarrow{1} CG$

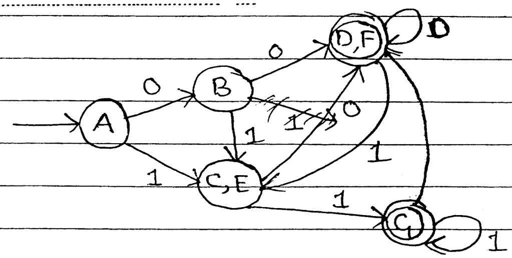
$\delta(A,C) \xrightarrow{0} BF$ x, $\xrightarrow{1} CG$

$\delta(A,B) \xrightarrow{0} BD$ x, $\xrightarrow{1} CE$

$\delta(B,E) \xrightarrow{0} DF$, $\xrightarrow{1} EG$ x

$\delta(B,C) \xrightarrow{0} DF$, $\xrightarrow{1} EG$ x

~~Position~~ Partition set = {{A}, {B}, {C,E}, {D,F}, {G}}



δ	0	1
→ {A}	B	C,E
{B}	D,F	C,E
{C,E}	D,F	G
* {D,F}	~~D,F~~	C,E
* {G}	D,F	G

N = ({Q, {0,1}, δ, {A}, {{D,F},{G}}})

Q = {{A}, {B}, {C,E}, {D,F}, {G}}

2.

δ	0	1
→ q_1	q_2	q_3
q_2	q_3	q_5
* q_3	q_4	q_3
q_4	q_3	q_5
* q_5	q_2	q_5.

q2	X			
q3	X	X		
q4	X	✓	X	
q5	X	X	✓	X
	q1	q2	q3	q4

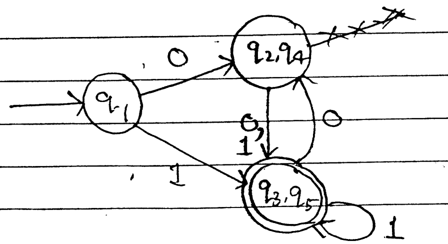
$\delta(q_1, q_4) \xrightarrow{0} q_2 q_3$ X
$ \xrightarrow{1}$

$\delta(q_1, q_2) \xrightarrow{0} q_2 q_3$ X
$ \xrightarrow{1}$

$\delta(q_2, q_4) \xrightarrow{0} q_3 q_3$ ✓
$ \xrightarrow{1} q_5 q_5$

$\delta(q_3, q_5) \xrightarrow{0} q_4 q_2$ ✓
$ \xrightarrow{1} q_3 q_5$

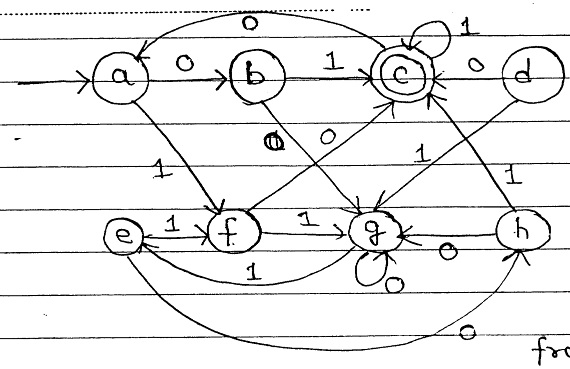
Partition set = $\{\{q_1\}, \{q_2, q_4\}, \{q_3, q_5\}\}$.



δ	0	1.
$\rightarrow \{q_1\}$	q_2, q_4	q_3, q_5
$\{q_2, q_4\}$	q_3, q_5	q_3, q_5
* $\{q_3, q_5\}$	q_2, q_4	q_3, q_5

$A = (\{\{q_1\}, \{q_2, q_4\}, \{q_3, q_5\}\}, \{0, 1\}, \delta, \{q_1\}, \{q_3, q_5\})$

(3)



from start state 'a'
'd' is not reachable.

δ	0	1.
$\rightarrow$ a	b	f
b	g	c
* c	a	c
e	h	f
f	c	g
g	g	e
h	g	c

b	X					
c	X	X				
e	✓	X	X			
f	X	X	X	X		
g	✓	X	X	✓	X	
h	X	✓	X	X	X	X
	a	b	c	e	f	g

$\delta(a, h) \xrightarrow{0} bg$
$ \xrightarrow{1} fc$ X

$\delta(a, g) \xrightarrow{0} bg$ ✓
$ \xrightarrow{1} fe$

$\delta(a,f) \xrightarrow{0} bc$ X
$\xrightarrow{1}$

$\delta(a,e) \xrightarrow{0} bh$
$\xrightarrow{1} ff$ ✓

$\delta(a,b) \xrightarrow{0} bg$
$\xrightarrow{1} fc$ X

$\delta(b,h) \xrightarrow{0} gg$
$\xrightarrow{1} cc$ ✓

$\delta(b,g) \xrightarrow{0} gg$
$\xrightarrow{1} ce$ X

$\delta(b,f) \xrightarrow{0} gc$ X
$\xrightarrow{1}$

$\delta(b,e) \xrightarrow{0} gh$
$\xrightarrow{1} cf$ X

$\delta(e,h) \xrightarrow{0} hg$
$\xrightarrow{1} fc$ X

$\delta(e,g) \xrightarrow{0} hg$
$\xrightarrow{1} fe$ ✓

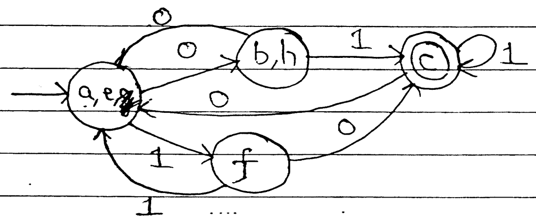
$\delta(e,f) \xrightarrow{0} hc$ X
$\xrightarrow{1}$

$\delta(f,h) \xrightarrow{0} cg$ X
$\xrightarrow{1}$

$\delta(f,g) \xrightarrow{0} cg$ X
$\xrightarrow{1}$

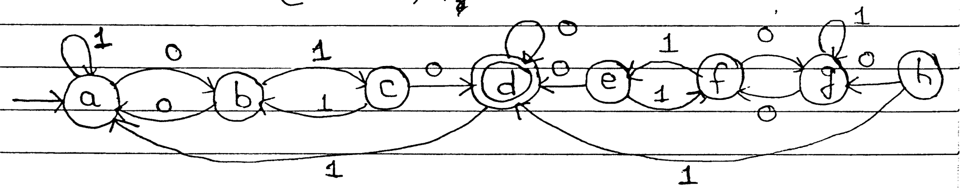
$\delta(g,h) \xrightarrow{0} gg$
$\xrightarrow{1} ec$ X

Partition set = { {a,e,~~g~~} {b,h}, {c} {f} }



δ	0	1
→ {a,e,~~g~~}	b,h	f
{b,h}	a,e,~~g~~	c
* {c}	a,e,~~g~~	c
{f}	~~a,e,g~~ c	~~1c~~ a,e,~~g~~

4.



b	✓		
c	X	X	
d	X	X	X
	a	b	c

$\delta(a,c) \xrightarrow{0} bd$ X
$\xrightarrow{1}$

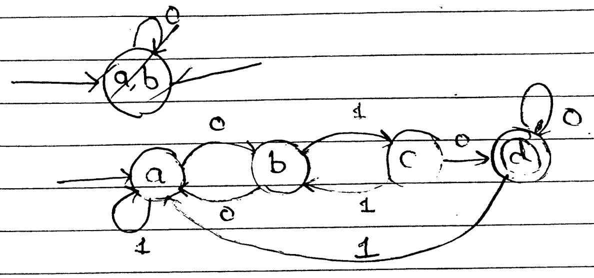
$\delta(a,b) \xrightarrow{0} ba$
$\xrightarrow{1} ac$ ✓

$\delta(b,c) \xrightarrow{0} ad$ X
$\xrightarrow{1}$

papergrid

Date: / /

Partition set : { {a}{b}, {c}, {d} }



papergrid

Date: / /

Algorithm for minimization of DFA:

1. Eliminate any state that can't be reached from start state.

2. Partition the remaining states into blocks so that all states in same block are equivalent and no pair of states from diff. blocks are equivalent.

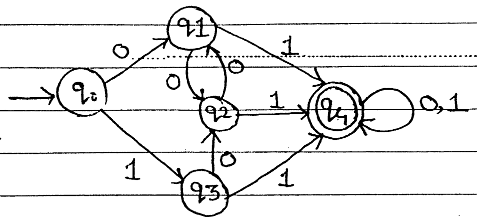
Suppose, DFA $A = (Q, \Sigma, \delta, q_0, F)$ is to be minimized then follow the following steps:

i) Use table filling algorithm to find all pairs of equivalent state.

ii) Partition the set of states Q into blocks of mutually equivalent states.

iii) Construct the min. state equivalent DFA 'B' by using blocks as its state. Let, γ be the transition function of B. Suppose, S is set of equivalent states A and a is the input symbol then there must exist one block T of states such that for all the states q in S, $\delta(q, a)$ is a member of block T. In addition, the start state of B is the block containing start state of A. The set of accepting states B is the set of blocks containing accepting states of A.

1.



q1	×			
q2	×	✓		
q3	×	✓	✓	
q4	×	×	×	×
	q0	q1	q2	q3

$\delta(q_0, q_3) \xrightarrow{0} q_1 q_2$
$\qquad \xrightarrow{1} q_3 q_4$ ×

$\delta(q_0, q_2) \xrightarrow{0} q_1 q_1$
$\qquad \xrightarrow{1} q_3 q_4$ ×

$\delta(q_0, q_1) \xrightarrow{0} q_1 q_2$
$\qquad \xrightarrow{1} q_3 q_4$ ×

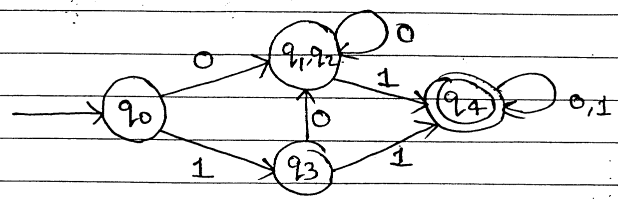
$\delta(q_1, q_3) \xrightarrow{0} q_2 q_2$
$\qquad \xrightarrow{1} q_4 q_4$

$\delta(q_1, q_2) \xrightarrow{0} q_2 q_1$
$\qquad \xrightarrow{1} q_4 q_4$

$\delta(q_2, q_3) \xrightarrow{0} q_1 q_2$
$\qquad \xrightarrow{1} q_4 q_4$

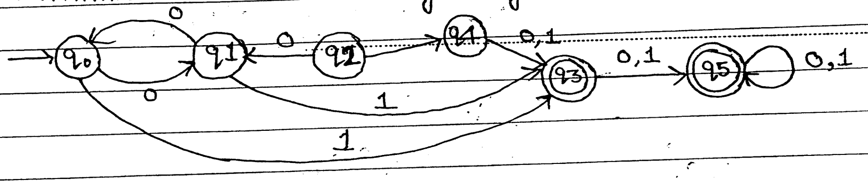
papergrid
Date: / /

Partition set = $\{ \cancel{\{q_1, q_3\}}, \cancel{\{q_0\}},$
$\{ \{q_0\}, \{q_1, q_2, q_3\} \cancel{\{q_3\}}, \{q_4\} \}$



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

1. Minimize the DFA using algorithm.



I.

q1	✓		
q3	X	X	
q5	X	X	✓
	q0	q1	q3

II. $\delta(q_0,q_1) \xrightarrow{0} q_1q_0$ ✓

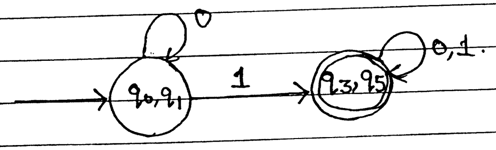
$\xrightarrow{1} q_3q_3$

$\delta(q_3,q_5) \xrightarrow{0} q_5q_5$

$\xrightarrow{1} q_5q_5$ ✓

III. Partition states are: $\{\{q_0,q_1\},\{q_3,q_5\}\}$

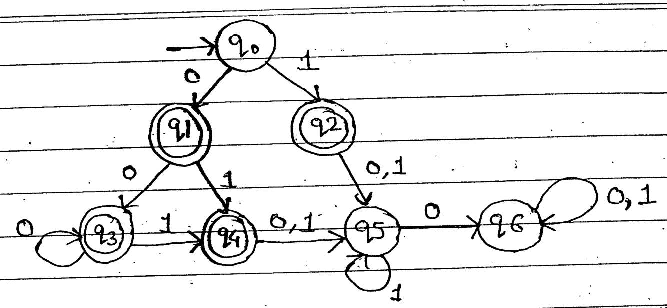
IV.



δ	0	1
→ q_0q_1	q_0,q_1	q_3,q_5
* q_3,q_5	q_3,q_5	q_3,q_5

$A = (\{\{q_0,q_1\},\{q_3,q_5\}\}, \{0,1\}, \delta, \{q_0,q_1\}, \{\{q_3,q_5\}\})$.

2.



q1	X					
q2	X	X				
q3	X	✓	X			
q4	X	X	✓			
q5	X	X	X	X	X	
q6	X	X	X	X	X	
	q0	q1	q2	q3	q4	q5

$\delta(q_0,q_6) \xrightarrow{0} q_1q_6$

$\xrightarrow{1} q_4q_6$ X

$\delta(q_2,q_3) \xrightarrow{0} q_5q_3$

$\xrightarrow{1} q_3q_4$ X

$\delta(q_0,q_5) \xrightarrow{0} q_1q_6$ X

$\xrightarrow{1}$

$\delta(q_1,q_4) \xrightarrow{0} q_3q_5$ X

$\xrightarrow{1} q_4q_5$

$\delta(q_1,q_3) \xrightarrow{0} q_3q_3$

$\xrightarrow{1} q_4q_4$ ✓

$\delta(q_1,q_2) \xrightarrow{0} q_3q_5$ X

$\xrightarrow{1}$

$\delta(q_2,q_4) \xrightarrow{0} q_5q_5$

$\xrightarrow{1} q_5,q_5$ ✓

Unit-3

Context Free Grammars

The context free grammar is denoted by $G = (V, T, P, S)$ where V & T are finite set of variables ~~called~~ and terminals.

- We assume V & T to be disjoint.
- P is finite set of productions. Each production is of the form $A \rightarrow \alpha$ where A is a variable and α is a string of symbols from $(V \cup T)^*$.
- S is a special variable called start symbol.

1. Construct CFG for generating all integers with sign.

$G = (V, T, P, S)$

V = {S, <sign>, <integer>, <digit>}
T = {0, 1, ...9, +, -}
P = { S → <sign><integer>
<integer> → <digit><integer> | <digit>
<digit> → 0|1|2|....|9
<sign> → + | -
}

S is the start symbol.

→ 8 +1234

S ⇒ <sign><integer>
⇒ + <integer>
⇒ + <digit><integer>
⇒ + 1 <integer>
⇒ +1 <digit><integer>

⇒ +12 <integer>
⇒ +12 <digit><integer>
⇒ ~~+123 <integer>~~
⇒ +123 <digit>
⇒ +1234

2. Consider the given grammar $G = (V, T, P, S)$
$V = \{S\}$.
$T = \{a, b\}$
$P = \{ S \rightarrow aSb \mid ab \}$

Find out language generated by given CFG.

S ⇒ aSb
S ⇒ aaSbb
S ⇒ aaaSbbb
S ⇒ aaaabbbb.

$L = \{ a^n b^n \mid n \geq 1 \}$

3. Consider the given grammar $G = (V, T, P, S)$
$V = \{S, A, B\}$
$T = \{a, b\}$. Find language.
$P = \{ S \rightarrow aB \mid bA$
$A \rightarrow a \mid aS \mid bAA$
$B \rightarrow b \mid bS \mid aBB \}$.

S ⇒ aB ~~bA~~	S ⇒ bA
S ⇒ abS	S ⇒ bbAA
S ⇒ abbA	S ⇒ bbaA
S ⇒ abbaS	S ⇒ bbaaS
S ⇒ abbabA	S ⇒ bbaaaB
S ⇒ abbaba	S ⇒ bbaaab

$L = \{ n_a(w) = n_b(w) \}$

4. Consider the grammar $G = (V, T, P, S)$

$V = \{S, A\}$
$T = \{a, b\}$
$P = \{ S \to aA \mid \varepsilon$
$\quad A \to bS \}$

Find the language.

$S \Rightarrow aA$ $\qquad S \Rightarrow \varepsilon$
$S \Rightarrow abS$
$S \Rightarrow abaA$
$S \Rightarrow ababS$
$S \Rightarrow abab\varepsilon$

$L = \{ (ab)^n \mid n \geq 0 \}$

5. Construct the CFG for following language
$L = \{ a^n b^n \mid n \geq 0 \}$.

Sol^n: $G = (V, T, P, S)$

$V = \{S\}$
$T = \{a, b\}$
$P = \{ S \to \varepsilon \mid aSb \}$.

S is the start symbol.

papergrid
Date: / /

6. Construct the CFG for following lang.

$L = \{ a^n b^{n+1} \mid n \geq 0 \}$.

Sol^n: $L = \{ b, abb, aabbb \ldots \}$

$G = (V, T, P, S)$

$V = \{S\}$.
$T = \{a, b\}$
$P = \{ S \to b \mid aSb \}$

S is start symbol.

7. Construct the CFG for generating set of palindromes over alphabet $\Sigma = \{a, b\}$.

Sol^n: $L = \{ \varepsilon, aba, \cancel{aabbaa}, abba, bab, a, b \ldots \}$.

$G = (V, T, P, S)$

$V = \{S, A, B\}$ $\qquad // \{S\}$
$T = \{a, b\}$.
$P = \{ S \to \varepsilon \mid ABA \mid BAB$ $\qquad // \{ S \to \varepsilon \mid a \mid b \mid aSa \mid bSb \}$
$\quad A \to \varepsilon \mid aA$
$\quad B \to \varepsilon \mid bB$
$\}$

→ Derive the string ababbaba:

a b a b b a

papergrid
Date: / /

$S \Rightarrow aSa$
$S \Rightarrow absba$
$S \Rightarrow abasaba$
$S \Rightarrow ababsbaba$
$S \Rightarrow ababbaba$

8. Construct the CFG for string of a's & b's which are of even length.

Sol^n: $L = \{ \varepsilon, aa, bb, abab, aaba, bbbb, aaaa \ldots \}$

$G = (V, T, P, S)$.
~~T = {~~
$V = \{S, \cancel{A, B}\}$
$T = \{a, b\}$.
$P = \{ S \rightarrow \varepsilon \mid aBa \mid bBb \mid ab \mid ba$
$B \rightarrow \varepsilon \mid ba \mid ab \mid aa \mid bb$
$A \rightarrow \varepsilon \mid ab \mid ba \mid aa \mid bb \}$ ✗

$P = \{ S \rightarrow \varepsilon \mid asa \mid asb \mid bsa \mid bsb \}$ ✓

$P = \{ S \rightarrow \varepsilon \mid \cancel{AS}$
$A \rightarrow aa \mid ab \mid ba \mid bb \}$

AS
~~AAS~~

9. For the given CFG, find the language.

$V = \{S\}$
$T = \{a, b\}$
$P = \{ S \rightarrow aSbS \mid bSaS \mid \varepsilon \}$.

$L = \{ n_a(\omega) = n_b(\omega) \}$

10. Find out the lang. for CFG.

$V = \{S\}$.
$T = \{a, b\}$.
$P = \{ S \rightarrow asb \mid bsa \mid ss \mid \varepsilon \}$.

$L = \{ n_a(\omega) = n_b(\omega) \}$

11. Construct a CFG for set of strings of balanced parenthesis i.e each left (has matching) & pairs of matching parenthesis are properly nested.
i.e. Construct a CFG for arithmatic ~~operato~~ expression consisting of +, -, *, /, ^

$G = (V, T, P, S)$

$V = \{E\}$
$T = \{ id, +, -, *, /, \wedge, (,) \}$
$P = \{ E \rightarrow E+E \mid E-E \mid E*E \mid E/E \mid id \mid E \wedge E \mid (E) \}$

→ $(((id + id) * id) \wedge id)$

$E \Rightarrow (E)$
$E \Rightarrow (E \wedge E)$
$E \Rightarrow ((E) \wedge E)$
$E \Rightarrow ((E * E) \wedge E)$
$E \Rightarrow (((E) * E) \wedge E)$
$E \Rightarrow (((E + E) * E) \wedge E)$
$E \Rightarrow (((id + E) * E) \wedge E)$
$E \Rightarrow (((id + id) * E) \wedge E)$
$E \Rightarrow (((id + id) * id) \wedge E)$
$E \Rightarrow (((id + id) * id) \wedge id)$

12. Construct the CFG for language
$L = \{a^n b^m \mid n \neq m\}$

Sol^n: $L = \{a, b, aa, bb, abb, \cancel{baa}, aaaa, bbbb, \ldots\}$.

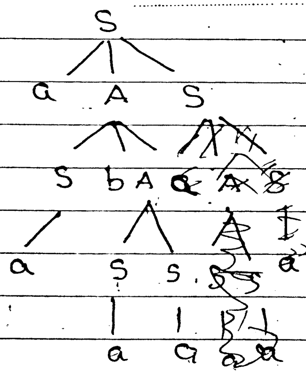
~~V = {~~
~~T = {~~
~~P = { S → aA | bB~~
~~A → ε | aA | bB~~
~~B → ε | b | aA~~
~~}~~

$P = \{ S \rightarrow AS_1 \mid S_1B$
$S_1 \rightarrow aS_1b \mid \varepsilon$
$A \rightarrow a \mid aA \mid B \rightarrow b \mid bB$

aaaabb
$S \Rightarrow AS_1$
$S \Rightarrow AaAS_1b$
$S \Rightarrow aaaaAbb$

1. Construct the parse tree / derivation tree for following grammar.
The given string is aabaaaa.

$P = \{ S \rightarrow aAS \mid a$
$A \rightarrow SbA \mid SS \mid ba \}$



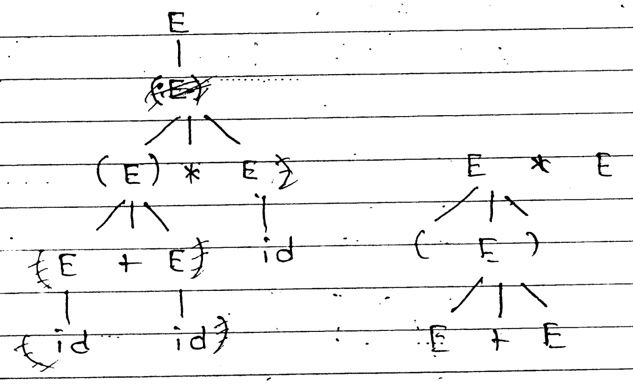
papergrid

Date: 13/11/18

1) Construct the parse tree.

E → E+E | E*E | (E) | id

yield : (id+id) * id



2) For the given grammar, construct the leftmost & rightmost derivation for the string aabaaa.

S → aAS | a

A → SbA | SS | bA

Soln:

S ⇒(lm) aAS

⇒ aSbAS

⇒ aabAS

⇒ aabSSS

⇒ aabaSS

⇒ aabaaS

⇒ aabaaa

S ⇒(rm) aAS

⇒ aAa

⇒ aSbAa

⇒ aSbSSa

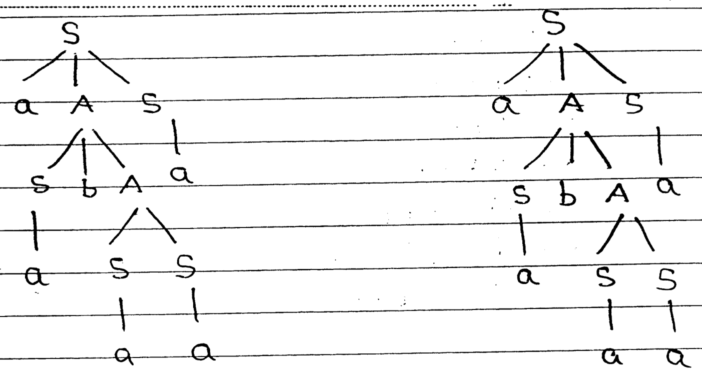
⇒ aSbSaa

papergrid

Date: / /

⇒ aSbaaa

⇒ aabaaa



3) S → aB | bA

A → a | aS | bAA

B → b | bS | aBB

string : aaabbabbba

S ⇒(lm) aB

⇒ aaBB

⇒ aaaBBB

⇒ aaabBB

⇒ aaabbSB

⇒ aaabbaBB

⇒ aaabba~~aBB~~bBB

⇒ aaabbab~~bA~~B

⇒ aaabbabB

aaabbabbS

aaabbabbbA

aaabbabbba

S → aB

S → aBB

B → b

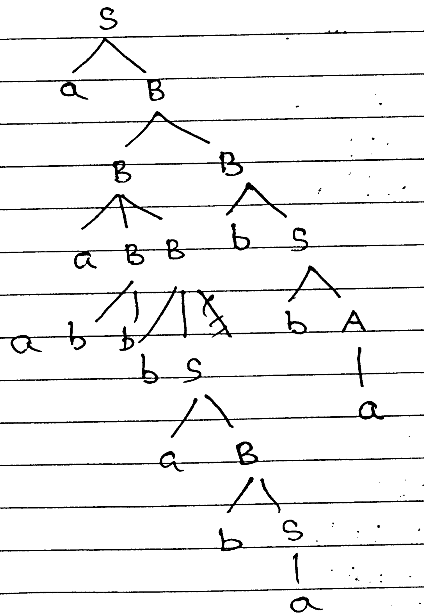
B → bS

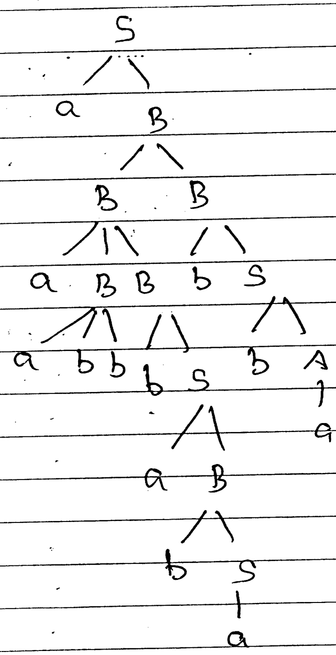
S → aB

B →

Date: / /

$S \Rightarrow aB$
$\underset{rm}{\Rightarrow} aaBB$
$\Rightarrow aaBbS$
$\Rightarrow aaBbbA$
$\Rightarrow aaBbba$
$\Rightarrow aaaBBbba$
$\Rightarrow aaaBbbba$
$\Rightarrow aaabSbbba$
$\Rightarrow aaabbAbbba$
$\Rightarrow aaabbabbba$





Date: / /

* Ambiguity in CFG :-

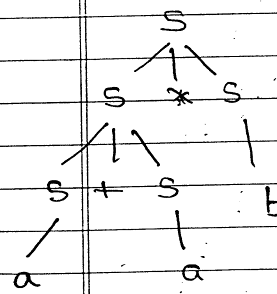
If a string in a grammar gives 2 leftmost or 2 rightmost derivation then grammar is said to be ambiguous.

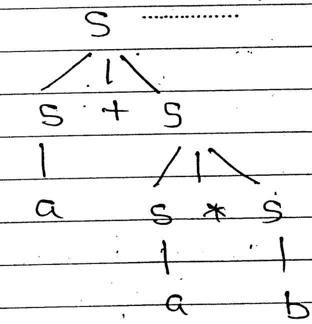
eg - $S+S \mid S*S \mid a \mid b$

string: $a+a*b$.

$S \underset{lm}{\Rightarrow} S*S$
$\Rightarrow S+S*S$
$\Rightarrow a+S*S$
$\Rightarrow a+a*S$
$\Rightarrow a+a*b$.

or, $S \underset{lm}{\Rightarrow} S+S$
$\Rightarrow a+S*S$
$\Rightarrow a+S*S$
$\Rightarrow a+a*S$
$\Rightarrow a+a*b$.

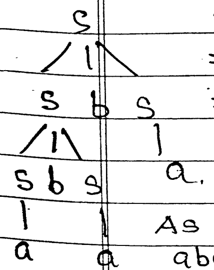


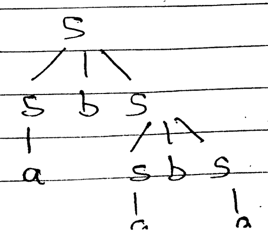


1. Show that the given grammar is ambiguous
$S \rightarrow SbS \mid a$.

$S \Rightarrow SbS$
$\Rightarrow SbSbS$
$\Rightarrow abSbS$
$\Rightarrow ababS$
$\Rightarrow ababa$

$S \Rightarrow SbS$
$\Rightarrow abSbS$
$\Rightarrow ababS$
$\Rightarrow ababa$.





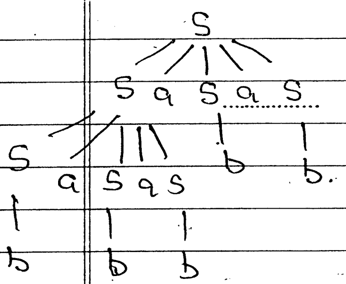
As the ~~two~~ strings ababa can be derived by 2 lm derivation.

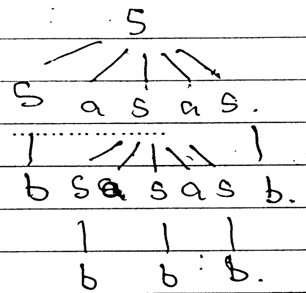
papergrid

Date: / /

2. Check whether given grammar is ambiguous.
$S \rightarrow SaSaS \mid b.$

$S \Rightarrow SaSaS$	$S \Rightarrow SaSaS$
$\Rightarrow SaSaSaSaS.$	$\Rightarrow basas.$
$\Rightarrow basasasas$	$\Rightarrow basasasas$
$\Rightarrow babasasas$	$\Rightarrow babasasas$
$\Rightarrow bababasas$	$\Rightarrow bababasas.$
$\Rightarrow babababas$	$\Rightarrow babababas$
$\Rightarrow babababab$	$\Rightarrow babababab.$

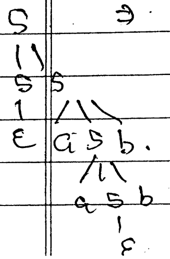


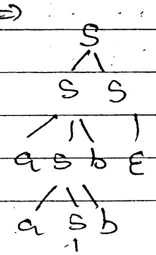


String babababab can be derived by two leftmost transition. Hence, it is ambiguous

3. Check whether grammar is ambiguous
$S \rightarrow aSb \mid SS \mid \epsilon.$

$S \Rightarrow SS$	$S \Rightarrow \cancel{aSb}.\ SS$
$\Rightarrow \cancel{aSbS}\ \cancel{S}S$	$\Rightarrow aSbS$
$\Rightarrow \cancel{a\epsilon b}\ asb.\ asb$	$\Rightarrow aaSbb$
$\Rightarrow \cancel{abab}\ aasbb.$	$\Rightarrow aabb$
aabb.	





papergrid

Date: / /

1. Construct the CFG for following language:
$L = \{a^n \mid n>0\}.$

$L = \{a, aa, aaa, \ldots\}.$

$G = (V, T, P, S)$

$P = \{S \rightarrow aS \mid a\}.$

$V = \{S\}.$

$T = \{a\}.$

2. Construct CFG: $L = \{a^{2n} \mid n>0\}.$

$G = (V, T, P, S)$ $\quad L = \{aa, aaaa, \ldots\}$

$V = \{S\}.$

$T = \{a\}$

$P = \{S \rightarrow aa \mid aaS\}.$

3. $L = \{a^n b^n \mid n>0\}.$

$G = (V, T, P, S)$

$V = \{S\}.$

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

$P = \{S \rightarrow ab \mid aSb\}$

4. $L = \{(ab)^n \mid n>0\}.$

$G = (V, T, P, S)$

$V = \{S\}.$

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

$P = \{S \rightarrow ab \mid abS \mid \cancel{Sab}\}$

5. $L = \{a^n b^m \mid m,n>0\}$.

$G = \{V, T, P, S\}$, $V = \{S, A, B\}$, $T = \{a, b\}$

$P = \{ S \rightarrow aAbB$ // $P = \{S \rightarrow aS \mid aA$, $A \rightarrow bA \mid b\}$

$A \rightarrow aA \mid \varepsilon$

$B \rightarrow bB \mid \varepsilon \}$

6. $L = \{a^i b^j \mid i \leq 2j\}$. $i, j \geq 0$.

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

$L = \{\varepsilon, b, bb, abb, aab, bbbb, abbbb, aabbbb, \ldots\}$

$P = \{ S \rightarrow \cancel{\varepsilon \mid bbS \mid a \mid aSbb}$

$\cancel{\varepsilon \mid Sb \mid aSb}$

$\varepsilon \mid Sb \mid aaSb \mid aSb \}$

7. $L = \{wcw^r \mid w \in (a,b)^*\}$ r: reverse

$L = \{c, aca, bcb, aacaa, abcba, bacab, bbcbb \ldots\}$

$P = \{S \rightarrow c \mid aSa \mid bSb\}$

8. $L = \{a^n b^n c^m \mid m,n>0\}$

$L = \{abc, abcc, aabbc, aabbcc, \ldots\}$

$P = \{ S \rightarrow abc \mid aAb\cancel{A}cB$ // $\{S \rightarrow AB$

$A \rightarrow \varepsilon \mid aAb \mid \cancel{a \mid b \mid bA}$ $A \rightarrow aAb \mid ab$

$B \rightarrow cB \mid \varepsilon \}$ $B \rightarrow cB \mid c\}$

papergrid
Date: / /

9. $L = \{a^i b^j c^k \mid j = i+k\}$. $i, j, k \geq 0$

$L = \{\varepsilon, ab, bc, abbc, aabb, bbcc, \ldots\}$

$P = \{ S \rightarrow \varepsilon \mid \cancel{b}aS1bS2 \mid \cancel{bS2}\ \cancel{bS1b}S2c$

$S1 \rightarrow aS1b \mid \varepsilon$

$S2 \rightarrow bS2c \mid \varepsilon$

10. $L = \{0^i 1^j 0^k \mid j > i+k\}$.

$S \rightarrow ABC$

$A \rightarrow 0A1 \mid \cancel{0}\varepsilon$

$C \rightarrow 1C0 \mid \varepsilon$

$B \rightarrow 1B \mid 1$

11. $L = \{a^n b^n c^m d^m \mid m,n>0\}$

$S \rightarrow AB$

$A \rightarrow \cancel{aA}aAb \mid ab$

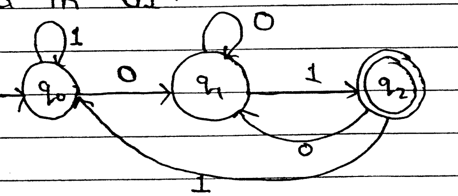
$B \rightarrow cBd \mid cd$

14.Nov.18

If L & M are regular so is L∪M.

Proof: Since, L&M are regular they have the regular expressions say $L = L(R)$ and $M = M(S)$ then $L \cup M = L(R+S)$.
By the definition of + operator for R.E.

1. Construct a DFA that accepts all the strings that end in 01.



Date: / /

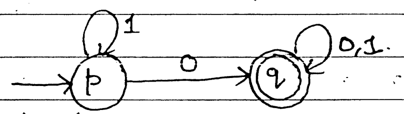
→ Steps for constructing DFA for R.L. :

1. Reverse all the arcs in transition diag. of original DFA.

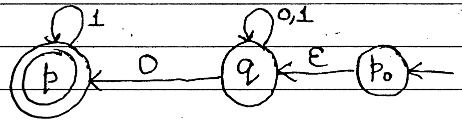
2. Make start state of original DFA be only accepting state for new automata.

3. Create a new start state p_0 with transition on ϵ to all the accepting state of DFA.

Q.



Construct the DFA for reversal of the given lang.



Corol1. If $L = L(A)$ for some DFA A, then there is a regular expression R such that $L = L(R)$.

Date: 23/11/18

→ CFG to CNF (Chomsky Normal Form)

Steps: 1. Elimination of ϵ-productions.

2. Elimination of unit productions.

3. Elimination of useless symbols.

4. Convert to CNF.

Sentential form : Let $G = (V, T, P, S)$ be a CFG then any string α in $(V \cup T)^*$ such that $S \overset{*}{\Rightarrow} \alpha$ (multiple ~~transitions~~ production) then α is a sentential form.

- If $S \overset{*}{\underset{lm}{\Rightarrow}} \alpha$, then α is a left sentential form
- If $S \overset{*}{\underset{rm}{\Rightarrow}} \alpha$, then α is a right sentential form.

→ Elimination of useless symbols from CFG :-

When we say that a symbol X is useful, there are two steps to determine whether X is useful:

i) We say that X is generating if $X \overset{*}{\Rightarrow} w$ for some terminal string w.

ii) We say that X is reachable if there is a derivation $S \overset{*}{\Rightarrow} \alpha X \beta$ for some strings α & β.

1. Obtain the grammar by removing the useless symbol from following CFG.

$S \rightarrow aSb \mid A \mid \epsilon$

$A \rightarrow aA$

Soln: The start symbol S is always generating and reachable. So, it is useful.

Next, we will check whether A is useful or not:

$A \Rightarrow aA$

with this production it is not possible to generate any strings.

Hence, A is neither generating nor reachable.

Grammar without useless symbol is

$G' = (V', T', P', S)$

where,

$V' = \{S\}$

$T' = \{a, b\}$

$P' = \{S \rightarrow aSb \mid \varepsilon\}$

2. Obtain the grammar G' from following CFG after eliminating useless symbols.

$S \rightarrow A$

$A \rightarrow aA \mid \varepsilon$

$B \rightarrow bA$.

Solⁿ: The start symbol S is always generating and reachable. So, it is useful.

- For A, A is generating as production goes to ε.

$S \Rightarrow A$

$\Rightarrow aA \rightarrow \varepsilon$.

Hence, A is reachable. $\alpha = a, \beta = \varepsilon$

- For B, B is ~~neither~~ generating because, $B \Rightarrow bA$

$\Rightarrow b\varepsilon$

$\cancel{B \Rightarrow bA}$. $S \Rightarrow A$, $S \Rightarrow aA$.

Hence, B is not reachable. (no production of form $S \not\Rightarrow \alpha X B$)

Hence, it is not useful

Grammar without useless symbol

$G' = (V', T', P', S)$.

$V' = \{S, A\}$

$T' = \{a\}$

$P' = \{S \rightarrow A$

$A \rightarrow aA \mid \varepsilon\}$.

3. Obtain grammar G' from CFG by eliminating useless symbols.

$S \rightarrow AB \mid CA$.

$A \rightarrow a$.

$B \rightarrow BC \mid AB$.

$C \rightarrow aB \mid b$.

Solⁿ: The start symbol S is always generating and reachable. So, it is useful.

- For, A, $A \Rightarrow a$

Hence, it is generating.

$S \Rightarrow AB$.

$\Rightarrow \cancel{aB\,ABC}$

Hence, it is reachable, $\alpha = \varepsilon$, $\beta = B$.

So, it is useful.

- For B, $B \Rightarrow BC$

$\Rightarrow ABC$

$\Rightarrow aBb$.

It is not generating as no string can be generated.

Hence, it is neither generating nor reachable.

So, it is not useful.

- For C, $C \Rightarrow b$.

It is generating.

$S \Rightarrow CA$.

$\cancel{\Rightarrow CA}$

$\alpha = \varepsilon$, $\beta = A$.

It is reachable.

Hence, it is useful.

$G' = \{V', T', P', S\}$

$V' = \{S, A, C\}$

$T' = \{a, b\}$

$P' = \{\ S \rightarrow \cancel{AB} \, CA$

$A \rightarrow a$

$C \rightarrow \cancel{ab} \mid b$

$\}$

4. $S \rightarrow aAa$

$A \rightarrow Sb \mid bCC \mid DaA$

$C \rightarrow abb \mid DD$

$D \rightarrow aDA$

$E \rightarrow aC$

Sol^n:

The start symbol S is always generating and reachable. So, it is useful.

- A : A is ~~not~~ generating any string ~~on~~ applying productions. Hence, it is ~~not~~ useful.

$C \Rightarrow abb$

- C : C is ~~not~~ generating any string. Hence, it is ~~not~~ useful. $S \Rightarrow aAa$ $\Rightarrow abCCa$

- D : D is not generating any string. Hence, it is not useful.

- E : $E \Rightarrow aC$ It is generating but not reachable. Hence, it is not useful

$G' = (V', T', P', S)$

$V' = \{S, A, C\}$

$T' = \{a, b\}$

$P' = \{\ S \rightarrow aAa$

$A \rightarrow Sb \mid bCC \mid \cancel{DaA}$

$C \rightarrow abb \mid \cancel{DD}$

$\}$

Elimination of ε-productions :-

A production of the form $A \rightarrow \varepsilon$ is called an ε-production. Also, A can be called as nullable variable if $A \overset{*}{\Rightarrow} \varepsilon$.

1. Eliminate the ε-prod. from grammar G to obtain G'.

$P = \{\ S \rightarrow aS \mid AB$ $\quad G = (V, T, P, S)$

$A \rightarrow \varepsilon$

$B \rightarrow \varepsilon$

$D \rightarrow b$

$\}$

Sol^n:

$S \Rightarrow AB$

$\Rightarrow \varepsilon B$

$\Rightarrow \varepsilon\varepsilon$

$\Rightarrow \varepsilon$

$V_n = \{A, B, S\}$

$G' = (V', T', P', S)$

$P' = \{\ S \rightarrow aS \mid a$

$D \rightarrow b\}$

$V' = \{S, D\}$

$T' = \{a, b\}$

2. Obtain grammar G' after eliminating ϵ-prod. from following grammar G.

$$P = \{ S \rightarrow a \mid Xb \mid aYa$$
$$X \rightarrow Y \mid \epsilon$$
$$Y \rightarrow b \mid X$$
$$\}.$$

Solⁿ: $V_n = \{X, Y\}$.

$G' = (V', T', P', S)$

$$P' = \{ S \rightarrow a \mid Xb \mid aYa \mid b \mid aa$$
$$X \rightarrow \cancel{b}Y$$
$$Y \rightarrow b \mid X$$
$$\}$$

$V' = \{S, X, Y\}$.
$T' = \{a, b\}$

3. $G = (V, T, P, S)$

$$P = \{ S \rightarrow XY$$
$$X \rightarrow Zb$$
$$Y \rightarrow bW$$
$$W \rightarrow Z$$
$$A \rightarrow aA \mid bA \mid \epsilon$$
$$B \rightarrow Ba \mid Bb \mid \epsilon.$$
$$Z \rightarrow AB$$
$$\}.$$

$Z \Rightarrow AB$
$\Rightarrow \epsilon B.$
$\Rightarrow \epsilon\epsilon$
$\Rightarrow \epsilon.$

$V_n = \{A, B, Z, W\}$

$W \Rightarrow Z$
$\Rightarrow \epsilon$

$$P' = \{ S \rightarrow XY$$
$$X \rightarrow Zb \mid b$$
$$Y \rightarrow bW \mid b$$
$$W \rightarrow Z$$
$$\cancel{A} \rightarrow aA \mid bA \mid a \mid b$$
$$B \rightarrow Ba \mid Bb \mid a \mid b$$
$$Z \rightarrow AB \mid A \mid B$$
$$\}$$

$V' = \{S, X, Y, W, A, B, Z\}$
$T' = \{a, b\}$

→ Elimination of unit productions:
A production of the form $A \rightarrow B$ or $X \rightarrow Y$ is a unit production.

Concept of unit pair:
A pair (A, A) is called as a unit pair for any variable A i.e. $A \overset{*}{\Rightarrow} A$ by zero step. If (A, B) is a unit pair and $B \rightarrow C$ is a production where C is a variable then (A, C) is a unit pair.

1. Obtain grammar G' from the following grammar by eliminating unit productions.

$$P = \{ S \rightarrow A \mid bb$$
$$A \rightarrow B \mid b$$
$$B \rightarrow S \mid a$$
$$\}$$

Solⁿ: (S, S) and the production $S \rightarrow A$ gives the unit pair (S, A).
(S, A) and the production $A \rightarrow B$ gives unit pair (S, B)
(S, B) and the production $B \rightarrow S$ gives unit pair (S, S)
(A, S) & $S \rightarrow A$ gives unit pair (A, A)
(A, A) and $A \rightarrow B$ gives unit pair (A, B)
(A, B) and $B \rightarrow S$ gives unit pair (A, S)
(B, B) and $B \rightarrow S$ gives unit pair (B, S)

(B,S) & ~~B~~ S → ~~A~~ gives unit pair (B,A)
(B,A) & A → B gives unit pair (B,B)

Pair	Production
1. (S,S)	S → bb
2. (S,A)	S → b
3. (S,B)	S → a
4. (A,A)	A → b
5. (A,B)	A → a
6. (A,S)	A → bb
7. (B,B)	B → a
8. (B,S)	B → bb
9. (B,A)	B → b

P' = { S → bb|b|a
A → bb|b|a
B → bb|b|a
}.

V' = {S,A,B}.
T' = {a,b}

2.

P = { S → AB
B → C|b
D → E
A → a
C → D
E → a
}

B → C → D → E

~~(S,S)~~ (B,B) & B → C gives unit pair (B,C)
(B,C) & C → D gives unit pair (B,D)
(B,D) & D → E gives unit pair (B,E)

(C,C) & C → D gives (C,D)
(C,D) & D → E gives (C,E)

(D,D) & D → E gives (D,E)

Pair	Production
~~(S~~ (B,B)	B → b
(B,C)	~~B→~~ —
(B,D)	—
(C,C)	—
(C,D)	—
~~(D,D)~~ (C,E)	C → a
(D,D)	—
(D,E)	D → a
~~(B,E)~~	B → a
(S,S)	S → AB
(A,A)	A → a
(E,E)	E → a

P' = { S → AB
A → a
B → b|a
C → a
D → a
E → a
}

V' = {S,A,B,C,D,E}.
T' = {a,b}

3. $P = \{ \quad E \to T \mid E+T$

$T \to F \mid T*F$

$\cancel{F \to I \mid (E)}$

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

$\}$.

(E,E) & $E \to T$ gives (E,T)

(E,T) & $T \to F$ gives (E,F)

(E,F) & $F \to I$ gives (E,I)

(T,T) & $T \to F$ gives (T,F)

(T,F) & $F \to I$ gives (T,I)

(F,F) & $F \to I$ gives (F,I)

Pair	Production.
(E,E)	$E \to E+T$
(E,T)	$E \to \cancel{E*T}\ T*F$
(E,F)	$E \to (E)$
(E,I)	$E \to a \mid b \mid Ia \mid Ib \mid I0 \mid I1.\} \; Z$
(T,T)	$T \to T*F.$
(T,F)	$T \to (E)$
(T,I)	$T \to a \mid b \mid Ia \mid Ib \mid I0 \mid I1$
(F,F)	$F \to (E)$
(F,I)	$F \to a \mid b \mid Ia \mid Ib \mid I0 \mid I1$
(I,I)	$I \to a \mid b \mid Ia \mid Ib \mid I0 \mid I1$

$P' = \{ E \to E+T \mid T*F \mid (E) \mid Z$ → Write the complete grammar

$T \to T*F \mid (E) \mid Z$

$F \to (E) \mid Z$

$I \to Z$

$\}$

$V' = \{E, T, F, I\}$

$T' = \{a, b, 0, 1, (,), +, *\}$

→ Convert the following CFG to CNF.

A grammar G is said to be in CNF if it has the productions of following type.

$A \to BC$

$A \to a$

1. $P = \{ S \to aAD$

$A \to aB \mid bAB.$

$B \to b$

$D \to d$

$\}$

Sol^n:

- $S \to aAD$

Introduce variable C_1.

$C_1 \to a$

$S \to C_1 AD$

Introduce variable C_2.

$C_2 \to AD.$

Rewrite

$S \to C_1 C_2.$

- $A \to aB$

Rewrite, $A \to C_1 B.$

- $A \to bAB.$

Introduce variable C_3.

$C_3 \to b.$

$A \to C_3 AB.$

Introduce variable C_4.

$C_4 \to AB.$

Rewrite, $A \to C_3 C_4.$

$P' = \{ \; S \to C_1 C_2$

$C_1 \to a$

$C_2 \to AD$

$A \to C_1 B \mid C_3 C_4.$

$C_3 \to b$

$C_4 \to AB$

$B \to b$

$D \to d$

$\}.$

$V' = \{S, A, B, D, C_1, C_2, C_3, C_4\}.$

$T' = \{a, b, d\}.$

2. $P = \{ \; S \to aSa \mid bSb \mid a \mid b \mid aa \mid bb \}.$

$C_1 \to a$

$C_2 \to b$

$S \to C_1, \quad S \to C_2, \quad S \to C_1C_1, \quad S \to C_2C_2$

$C_3 \to Sa, \quad C_4 \to Sb$

$S \to aC_3 \mid S \to bC_4.$

$S \to C_1C_3 \mid S \to C_2C_4.$

$P' = \{ \; S \to C_1C_3 \mid C_2C_4 \mid C_1 \mid C_2 \mid C_1C_1 \mid C_2C_2 \}.$

$C_1 \to a, \; C_2 \to b, \; C_3 \to Sa, \; C_4 \to Sb$

$V' = \{S, C_1, C_2, C_3, C_4\}$

$T' = \{a, b\}$

....One day work is ~~lef~~ incomplete.

29/Nov/18

→ **Pushdown Automata (PDA):**

It has seven components $P = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$

where, Q is finite set of state.

Σ is set of input symbols.

Γ is finite stack alphabet.

δ is transition function.

q_0 is start state.

Z_0 is start symbol in stack

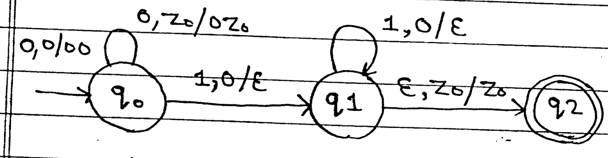
F is set of final or accepting states.

The transition function δ takes as argument $\delta(q, a, X)$ where q is a state in Q, a is input symbol in Σ or $a = \varepsilon$ & X is stack symbol. The output of δ is a finite set of pair (p, γ) where p is new state & γ is string of stack symbol that replaces X at the top of stack.

1. Design a PDA that accepts the following language.

$L = \{ 0^n 1^n \mid n \geq 1 \}$

Sol^n:



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

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

$\Gamma = \{0 \, Z_0\}$

$\delta(q_0, 0, Z_0) = (q_0, 0Z_0)$

$\delta(q_0, 0, 0) = (q_0, 00)$

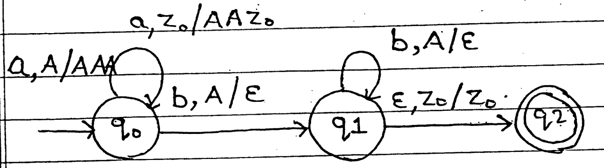
$\delta(q_0, 1, 0) = (q_1, \varepsilon)$

$\delta(q_1, 1, 0) = (q_1, \varepsilon)$

$\delta(q_1, \varepsilon, Z_0) = (q_2, Z_0)$

1. Construct PDA for following lang.

$L = \{ a^n b^{2n} \mid n \geq 1 \}$



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

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

$\tau = \{A\}$

$\delta(q_0, a, Z_0) = (q_0, AAZ_0)$

$\delta(q_0, a, A) = (q_0, AA\cancel{A})$

$\delta(q_0, b, A) = (q_1, \varepsilon)$

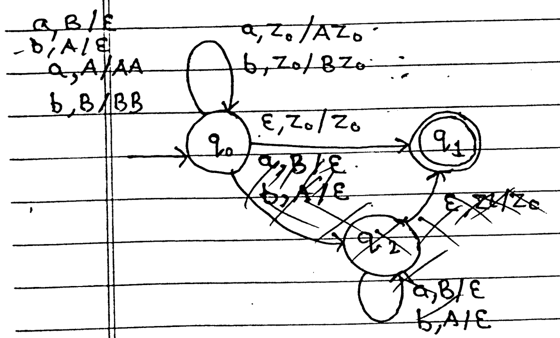
$\delta(q_1, b, A) = (q_1, \varepsilon)$

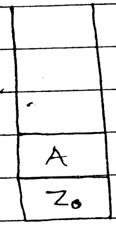
$\delta(q_1, \varepsilon, Z_0) = (q_2, Z_0)$

Start state is q_0.

$F = \{q_2\}$.

2. $L = \{ n_a(\omega) = n_b(\omega) \}$





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

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

$\tau = \{A, B\}$

$\delta(q_0, a, Z_0) = (q_0, AZ_0)$

$\delta(q_0, b, Z_0) = (q_0, BZ_0)$

$\delta(q_0, a, B) = (q_0, \varepsilon)$

$\delta(q_0, b, A) = (q_0, \varepsilon)$

$\delta(q_0, a, A) = (q_0, AA)$

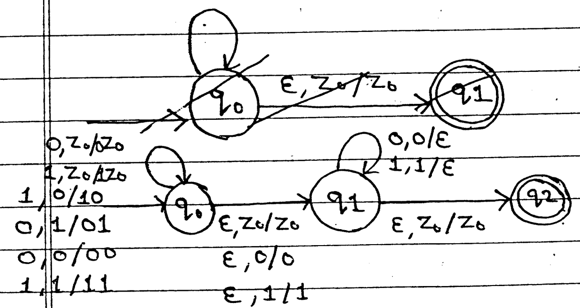
$\delta(q_0, b, B) = (q_0, BB)$.

$\delta(q_0, \varepsilon, Z_0) = (q_1, Z_0)$

03. Dec. 18 $F = \{q_1\}$

3. $L = \{ \omega\omega^R \mid \omega \text{ is in } (0+1)^* \}$

$L = \{ \varepsilon, 00, 11, 0110, 0000, 1001, \ldots \}$



$Q = \{q_0, q_1, q_2\}$. $\Sigma = \{0, 1\}$. , $\tau = \{0, 1\}$.

$\delta(q_0, 0, Z_0) = (q_0, 0Z_0)$

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

$\delta(q_0, 1, 0) = (q_0, 10)$

$\delta(q_0, 0, 1) = (q_0, 01)$

$\delta(q_0, 0, 0) = (q_0, 00)$

$\delta(q_0, 1, 1) = (q_0, 11)$

$\delta(q_0, \varepsilon, Z_0) = (q_1, Z_0)$

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

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

$\delta(q_1, 0, 0) = (q_1, \varepsilon)$

$\delta(q_1, \varepsilon, Z_0) = (q_2, Z_0)$

$F = \{q_2\}$

papergrid

Date: / /

⇒ Instantaneous Description of PDA – (ID) :-

~~We shall represent the~~ configuration of PDA by triple (q, w, γ) where q is a state, w is remaining input & γ is stack contents.
We show the top of stack at left end of γ and bottom at right end. Such a triple is called as instantaneous description of PDA.

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

Here, by consuming input 'a' & replacing x by α, we can go from 'q' to state 'P'.

1. For given PDA, write ID for following input string.
~~$(q_0, 111, z_0)$~~

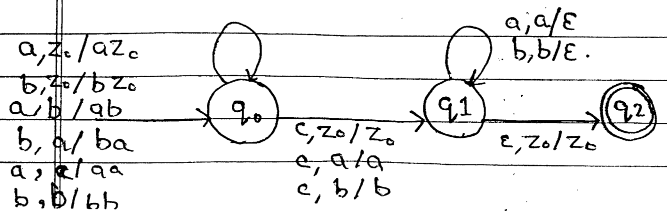
$L = \{ ww^R \mid w \text{ is in } (0+1)^* \}$.

$w = 1111$

Sol^n:

$(q_0, 1111, z_0) \vdash (q_0, 111, 1z_0)$
$\vdash (q_0, 11, 11z_0)$
$\vdash (q_1, 11, 11z_0)$
$\vdash (q_1, 1, 1z_0)$
$\vdash (q_1, \varepsilon, z_0)$
$\vdash (q_2, \varepsilon, z_0)$

2. $L = \{ wcw^R \mid w \text{ is in } (a+b)^* \}$
$L = \{ \varepsilon, aca, bcb, abcba, \ldots \}$



papergrid

Date: 04/12/18

3. Write down the ID for aabcbaa

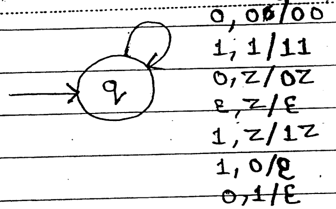
$(q_0, aabcbaa, z_0) \vdash (q_0, abcbaa, az_0)$
$\vdash (q_0, bcbaa, aaz_0)$
$\vdash (q_0, cbaa, baaz_0)$
$\vdash (q_1, baa, baaz_0)$
$\vdash (q_1, aa, aaz_0)$
$\vdash (q_1, a, az_0)$
$\vdash (q_1, \varepsilon, z_0)$
$\vdash (q_2, \varepsilon, z_0)$

Languages of a PDA –
1) Acceptance by final state.
2) Acceptance by empty stack.

1. Acceptance by final state : Let, $P = (Q, \Sigma, \Gamma, \delta, q_0, z_0, F)$ be a PDA then $L(P)$ is the language accepted by the PDA P. $L(P) = \{ w \mid (q_0, w, z_0) \overset{*}{\vdash}_P (q, \varepsilon, \alpha) \}$ for some state 'q' in 'F' & any stack string α.
i.e. the starting in the initial ID with w as the remaining input. PDA 'P' consumes 'w' from the i/p & enters in an accepting state. The contents of stack at that time is irrelevant.

2. Acceptance by empty stack: For each PDA 'P' = $(Q, \Sigma, \Gamma, \delta, q_0, z_0, F)$ we define $N(P) = \{ w \mid (q_0, w, z_0) \overset{*}{\vdash}_P (q, \varepsilon, \varepsilon) \}$ for any state 'q' i.e. $N(P)$ is set of inputs 'w' that PDA 'P' can consume and at the same time empty its stack.

1\. Construct a PDA to accept a lang. of strings with equal no. of zeroe's and one's by acceptance by empty stack.

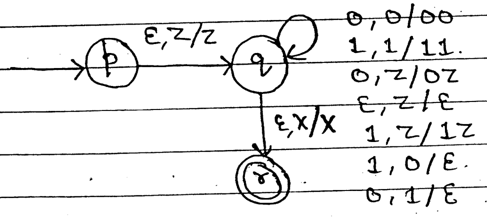


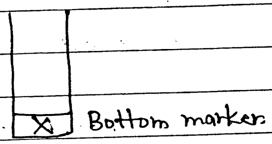
N : empty stack.

$P_N = (\{q\}, \{0,1\}, \{Z, 0, 1\}, \delta_N, q, Z)$

$\delta_N(q, 0, Z) = (q, 0Z)$
$\delta_N(q, 0, 1) = (q, \varepsilon)$
$\delta_N(q, 1, 0) = (q, \varepsilon)$
$\delta_N(q, 1, Z) = (q, 1Z)$
$\delta_N(q, 1, 1) = (q, 11)$
$\delta_N(q, 0, 0) = (q, 00)$
$\delta_N(q, \varepsilon, Z) = (q, \varepsilon)$

2\. Construct a PDA that accepts string of lang. equal no. of 0's & 1's by acceptance by final state.





$P_F = (\{p, q, r\}, \{0,1\}, \{Z, 0, 1, X\}, \delta_F, p, X, \{r\})$

papergrid
Date: / /

→ Equivalence b/w CFG's & PDA's :

1\. Conversion of CFG to equivalent PDA.

Let, $G = (V, T, P, S)$ be a CFG.

Construct the PDA 'P' that accepts the L(G) by empty stack as follows. $P = (\{q\}, T, V \cup T, \delta, q, S)$ where

δ is defined by -

i) for each variable A, $\delta(q, \varepsilon, A) = \{(q, \beta) \mid A \to \beta$ is a production of P$\}$.

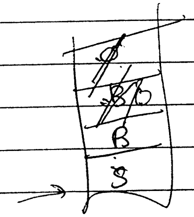
ii) for each terminal a, $\delta(q, a, a) = \{(q, \varepsilon)\}$

Q-1. Convert the following CFG to its equivalent PDA.

$P = \{ S \to 0BB$
$B \to 0S \mid 1S \mid 0$
$\}$

Sol^n:

$\delta(q, \varepsilon, S) = (q, 0BB)$
$\delta(q, \varepsilon, B) = \{(q, 0S), (q, 1S), (q, 0)\}$



$\delta(q, 0, 0) = \{(q, \varepsilon)\}$
$\delta(q, 1, 1) = \{(q, \varepsilon)\}$

$P = (\{q\}, \{0,1\}, \{S, B, 0, 1\}, \delta, q, S)$

Q2. Construct the following CFG ~~PDA~~ ~~for given CFG~~ to PDA.

$\{ S \to aABB \mid aAA.$
$A \to aBB \mid a$
$B \to bBB \mid A.$
$\}.$

Sol^n;

$\delta(q, \varepsilon, S) = \{(q, aABB), (q, aAA)\}.$
$\delta(q, \varepsilon, A) = \{(q, aBB), (q, a)\}.$
$\delta(q, \varepsilon, B) = \{(q, bBB), (q, A)\}.$

$\delta(q, a, a) = \{(q, \varepsilon)\}$ $\quad P = (\{q\}, \{a, b\}, \{S, A, B, a, b\}, \delta, q, S)$
$\delta(q, b, b) = \{(q, \varepsilon)\}$

Q3. $\{ I \rightarrow a \mid b \mid Ia \mid Ib \mid I0 \mid I1$

$E \rightarrow I \mid E*E \mid E+E \mid (E)$

$\}$.

Sol^n: $\delta(q,\varepsilon,I) = \{(q,a),(q,b),(q,Ia),(q,Ib),(q,I0),(q,I1)\}$

$\delta(q,\varepsilon,E) = \{(q,I),(q,E*E),(q,E+E),(q,(E))\}$.

$\delta(q,b,b) = \{q,\varepsilon\}$

$\delta(q,a,a) = \{q,\varepsilon\}$

$\delta(q,0,0) = \{q,\varepsilon\}$

$\delta(q,1,1) = \{q,\varepsilon\}$

$\delta(q,*,*) = \{q,\varepsilon\}$

$\delta(q,+,+) = \{q,\varepsilon\}$

$\delta(q,(,() = \{q,\varepsilon\}$

$\delta(q,),)) = \{q,\varepsilon\}$

Q4. Convert following CFG to CNF:

$\{ S \rightarrow 0A0 \mid 1B1 \mid BB$

$A \rightarrow C$

$B \rightarrow S \mid A$

$C \rightarrow S \mid \varepsilon$

$\}$

Sol^n: Since, S is start symbol and is always generating and reachable.
So, it is useful.

$A \Rightarrow C$
$\Rightarrow \varepsilon$

A is reachable ~~but not~~ and generating.

$B \Rightarrow A$
$\Rightarrow C$
$\Rightarrow \varepsilon$

~~A~~ B is generating & reachable.

papergrid
Date: / /

$C \Rightarrow \varepsilon$.

C is generating but ~~not~~ reachable.
~~Hence, it is not useful.~~

$P' = \{ S \rightarrow 0A0 \mid 1B1 \mid BB$

$A \rightarrow C$

$B \rightarrow S \mid A$.

$C \rightarrow S \mid \varepsilon$

$\}$.

Eliminating ε-Productions:

$P' = \{ S \rightarrow 0A0 \mid 1B1 \mid BB$ $\quad V_n = \{A,B,C,S\}$.

$A \rightarrow S$

$B \rightarrow S \mid A$.

$C \rightarrow S$

$\}$

Eliminating unit productions:

(S,S) ~~an~~ is an unit production.

(A,A) and $A \rightarrow S$ gives (A,S)

(B,B) and $B \rightarrow S$ gives (B,S)

(B,B) and $B \rightarrow A$ gives (B,A)

~~$B \rightarrow A$~~

(C,C) and $C \rightarrow S$ gives (C,S).

$P' = \{ S \rightarrow 0A0 \mid 1B1 \mid BB$.

$A \rightarrow$

papergrid

Date: / /

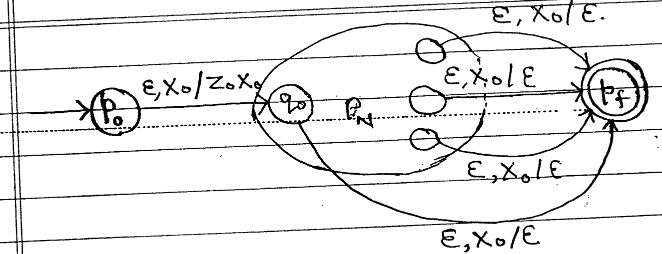
1. $\{ S \to aA \mid a \mid B \mid C$
$A \to aB \mid \varepsilon$
$B \to aA.$
$C \to cC$
$D \to ddd$
$\}$

2) $\{ S \to BAAB$
$A \to 0A2 \mid 2A0 \mid \varepsilon$
$B \to AB \mid 1B \mid \varepsilon$
$\}$

3. $\{ S \to ABA$
$A \to aA.$
$B \to bB \mid \varepsilon$
$\}$

4) $\{ S \to AaA \mid cA \mid Bab$
$A \to aaBa \mid cDa \mid aa \mid DC$
$B \to bB \mid bAB \mid bb \mid aS$
$C \to Ca \mid bC \mid D$
$D \to bD \mid \varepsilon$
$\}.$

Date: / /



P_F simulates P_N and accepts if P_N empties its stack.

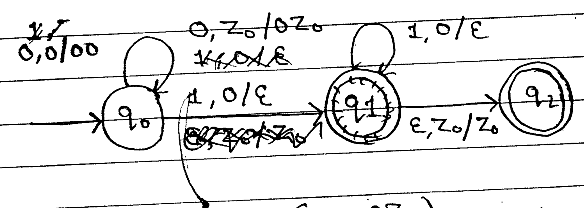
→ DETERMINISTIC PDA :-
We define PDA, $P = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$ to be deterministic if and only if following conditions are made :-

i) $\delta(q, a, X)$ has atmost 1 member for any 'q' in 'Q', 'a' in Σ or $a = \varepsilon$ and 'X' in Γ.

ii) if $\delta(q, a, X)$ is non-empty for some 'a' in Σ then $\delta(q, \varepsilon, X)$ should be empty.

Q1. Construct DPDA for following language :
$L = \{ 0^n 1^n \mid n \geq 1 \}$

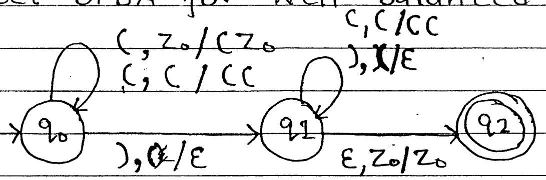
$0^n 1^n$:



$\delta(q_0, 0, Z_0) = (q_0, 0Z_0)$
$\delta(q_0, 0, 0) = (q_0, 00)$
$\delta(q_0, 1, 0) = (q_1, \varepsilon)$
$\delta(q_1, 1, 0) = (q_1, \varepsilon)$
$\delta(q_1, \varepsilon, Z_0) = (q_2, Z_0)$

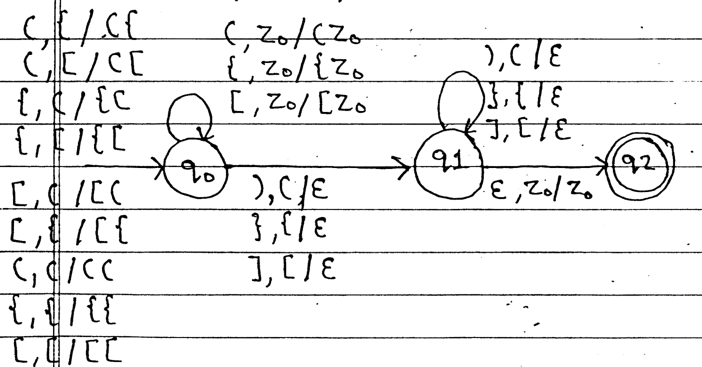
$\Sigma = \{0, 1\}$
$Q = \{q_0, q_1, q_2\}$
$\Gamma = \{Z_0\}$

2. Construct DPDA for well-balanced parenthesis.



$Q = \{q_0, q_1, q_2\}$
$\Sigma = \{(,)\}$
$\delta(q_0, (, Z_0) = (q_0, (Z_0)$
$\delta(q_0, (, () = (q_0, (()$
$\delta(q_0,), () = (q_1, \varepsilon)$
$\delta(q_1,), () = (q_1, \varepsilon)$
$\delta(q_1, \varepsilon, Z_0) = (q_2, Z_0)$

3. DPDA for well balanced paranthesis for all 3 types of bracket. ⟨(, {, [⟩



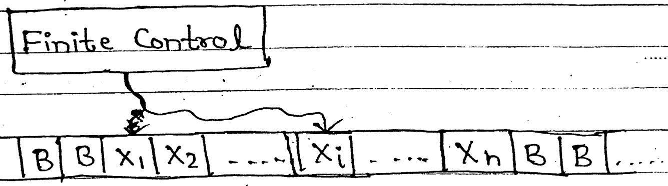
// Can be made using 2 states

4. $L = \{ a^i b^j c^k \mid i = j \,\|\, j = k \}$

5. $L = \{ 0^n 1^m 0^n \mid n, m \geq 1 \}$

06/Dec/2018

TURING MACHINE



We describe a turing machine by $M = (Q, \Sigma, \Gamma, \delta, q_0, B, F)$ where;

- Q = Finite set of states of finite control.
- Σ = Finite set of input symbols.
- Γ = Complete set of tape symbols $\Sigma \subset \Gamma$
- δ = Transition function.

The arguments of $\delta(q, X)$ are state 'q' & tape symbol X.

The value of $\delta(q, X)$ is define by a triple (P, Y, D) where P is the next state in Q, Y is symbol in Γ, ~~return~~ written in the cell being scanned, replacing whatever symbol was there. D is the dirⁿ, either left or right telling dirⁿ in which head moves.

q_0 is the start state in member Q.

B = Blank symbol which is in Γ but not in Σ.
The blank appears initially in all but the finite no. of initial cells that hold input symbols.

F = set of final states or accepting states.

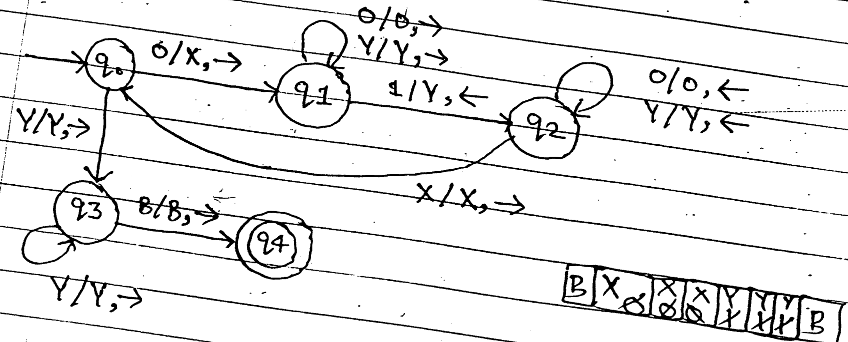
→ Construct the turing machine for following lang.
$L = \{ 0^n 1^n \mid n \geq 1 \}$

Solⁿ:

In the state q_0, if we get a zero, replace it with X' & move to state q1 and move tape head to right.

state q1, if we get 1 replace it with Y & move to the state q2 and move the tape head to l

2. In state q2, if we get X, replace it with X and move to the state and move the tape head to.



δ	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	—	—	—	—	—

$\delta(q_0, 0) = (P, Y, D)$

$Q = \{q_0, q_1, q_2, q_3, q_4\}$.

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

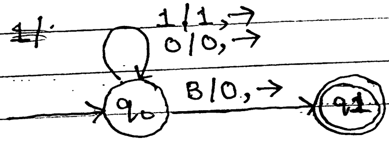
$\Gamma = \{0, 1, X, Y, B\}$

$q_0 = \{q_0\}$

$F = \{q_4\}$

$L = \{0^n 1^n 2^n \mid n \geq 1\}$.

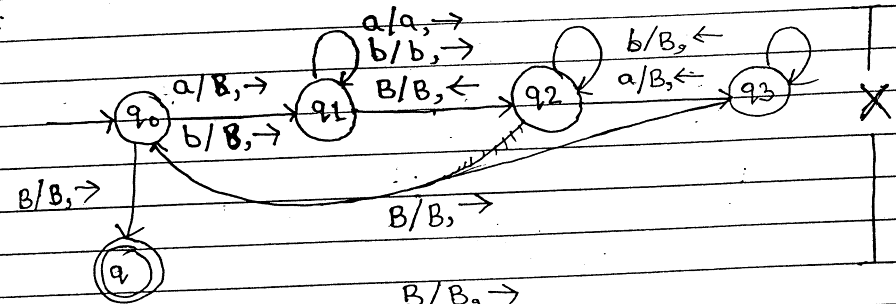
to double the number.

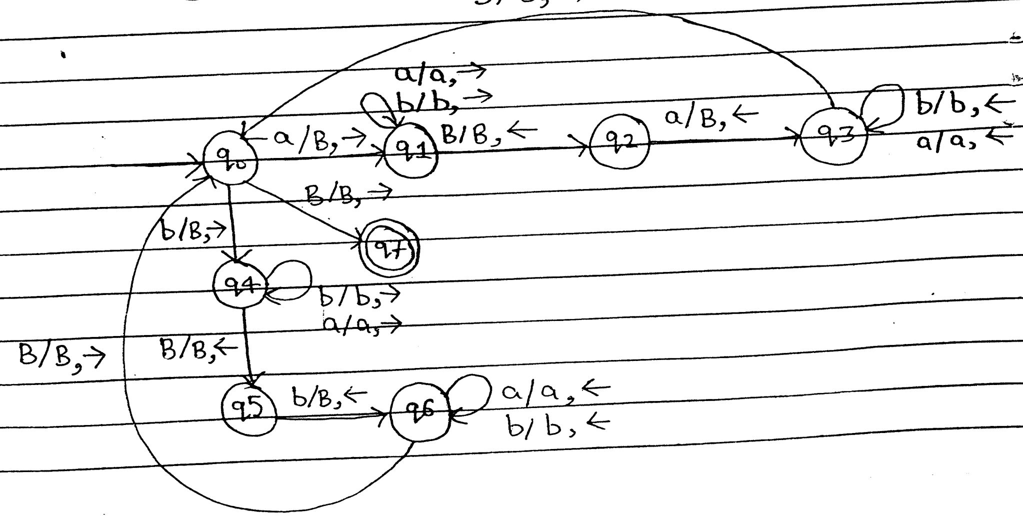


δ	0	1	B
q_0	$(q_0, 0, R)$	$(q_0, 1, R)$	(q_1, B, R)
q_1	—	—	—

for the following lang.

$L = \{ww^R \mid w \text{ is in } (a+b)^*\}$.

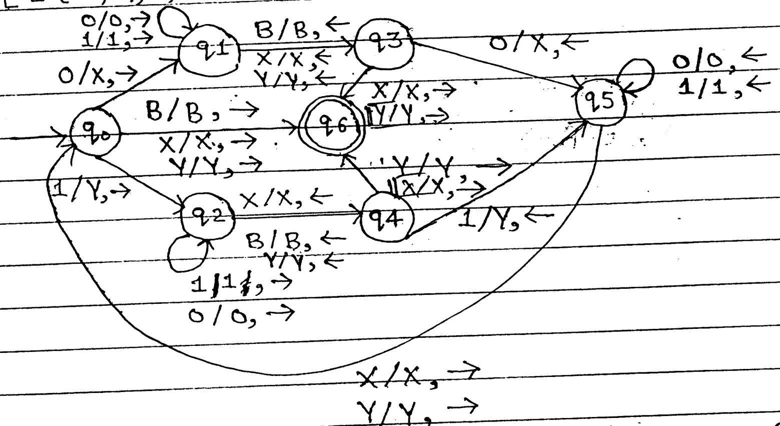




δ	a	b	B
q_0			
q_1			
q_2			
q_3			
q_4			
q_5			
q_6			
q_7.			

11.Dec.18 Turing machine for set of all palindromes over 0's and 1's.

$L = \{\varepsilon, 0, 1, 00, 11, 000, \ldots\ldots\}$



δ	0	1	X	Y	B
q_0					
q_1					
q_2					
q_3					
q_4					
q_5					
q_6					

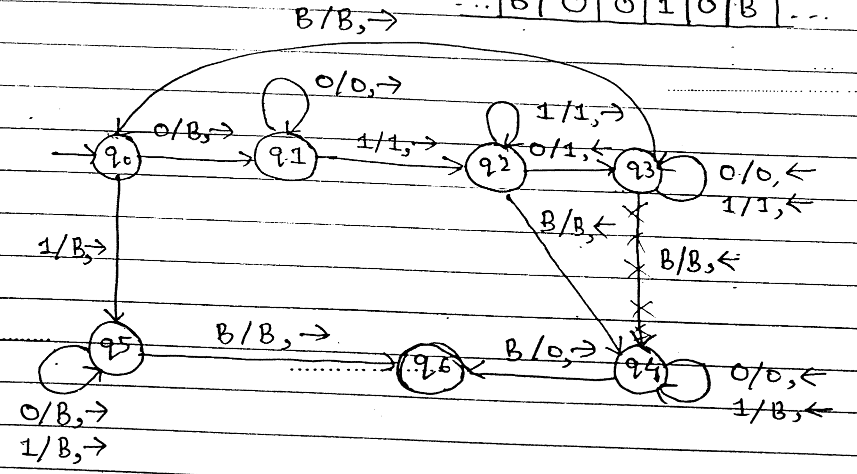
Proper Subtraction (Monus) :

$m \dot{-} n = \max(m-n, 0)$

$m>n \rightarrow [m-n]$
$m<n \Rightarrow 0$

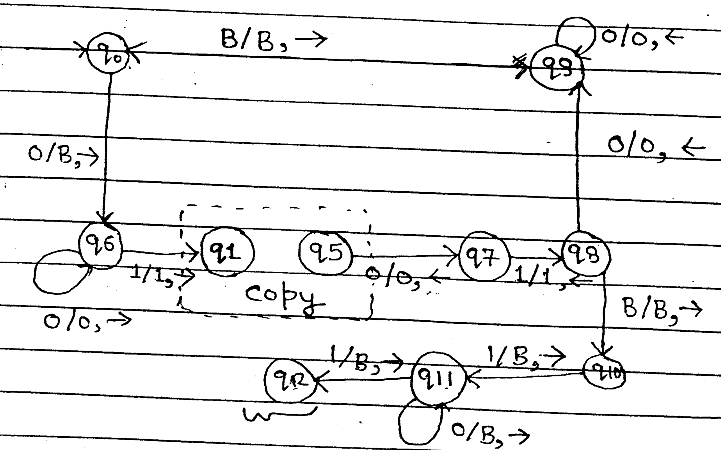
$0^m 1 0^n$: $m=2$, $n=1$.

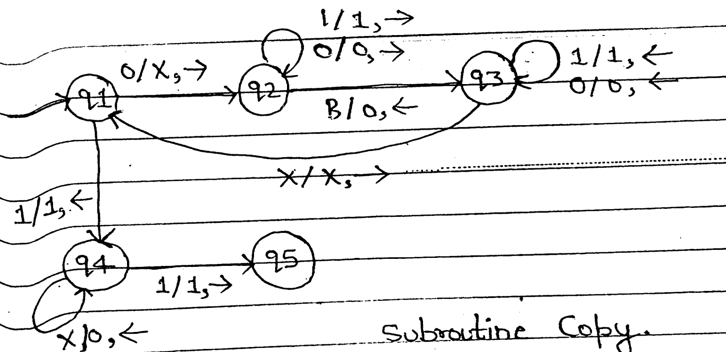
... | B | 0 | 0 | 1 | 0 | B | ...



12.12.18 Turing Machine : Subroutine

Multiply





Subroutine Copy.

Multiply two nos. m & n using concept of subroutine design turing machine.

eg- $0^m 1 0^n 1$: m = 1 , n = 1

...	B	0	1	0	1	B	...

m = 2 , n = 3

...	B	B	B	B	B	B	B	B	0	0	0	B	0	0

→ ID of a Turing Machine :
We denote the moves of turing machine by $\vdash^*$.
Suppose, $\delta(q, X_i) = (P, Y, L)$ i.e. the next move is leftward $X_1 X_2 \dots X_{i-1} q X_i X_{i+1} \dots X_n \vdash X_1 X_2 \dots X_{i-2}\, p X_{i-1} Y X_{i+1} \dots X_n$.

Notice, how this move reflects the change to state p & tape head is positioned at cell (i-1).

Suppose, $\delta(q, X_i) = (P, Y, R)$

$X_1 X_2 \dots X_{i-1} q X_i X_{i+1} \dots X_n \vdash X_1 X_2 \dots X_{i-1} Y p X_{i+1} \dots X_n$

13 12 18

* Instantaneous Description (ID) of Turing Machine:

$\delta(q, X_i) = (P, Y, L)$

→ Write down the ID for input string "00" for given turing machine [i.e. Palindrome].

$q_0 00 \vdash X q_1 0 \vdash X 0 q_1 \vdash X q_3 0 \vdash q_5 X X \vdash X q_0 X \vdash X X q_6$.

For 1001 :-

$q_0 1001 \vdash Y q_2 001 \vdash Y 0 q_2 01 \vdash Y 0 0 q_2 1 \vdash Y 0 0 1 q_2$

$\vdash Y 0 0 q_4 1 \vdash Y 0 q_5 0 Y \vdash Y q_5 0 0 Y \vdash q_5 Y 0 0 Y$

$\vdash Y q_0 0 0 Y \vdash Y X q_1 0 Y \vdash Y X 0 q_1 Y \vdash Y X 0 q_3 Y$

$\vdash Y X q_3 0 Y \vdash$ ~~Y X q X~~ $Y q_5 X X Y \vdash Y q_0 X X Y$

$\vdash Y X q_6 X Y$.

* Post's Correspondance Problem (PCP) :-
An instance of post correspondance problem consist of two list of strings over some alphabet Σ, two list must be of equal length.
We generally refer to the list as A & B where,
A = $w_1 w_2 \dots w_k$.
B = $x_1 x_2 \dots x_k$ for some integer k.
For each 'i', the pair (w_i, x_i) is said to be a corresponding pair. We say this instance of PCP has a solution if there is a sequence of one or more integers $i_1, i_2 \dots i_n$ when interpreted as indexes for strings in A & B lists yield same string.

i.e. $w_{i_1} w_{i_2} \dots w_{i_m} = x_{i_1} x_{i_2} \dots x_{i_m}$
The sequence $i_1, i_2 \dots i_m$ is solⁿ to the instance of PCP.

1.

i	List A w_i	List B x_i
1	110	110110
2	0011	00
3	0110	110

} equal length = 11

Let, m = 3
If $i_1 = 1$ is selected, ~~the~~ we are unable to get equal strings.

then, $i_1 = 2$ W = 00110110110
$i_2 = 3$ x = 00110110110
$i_3 = 1$

As we have the sequence of integers (2, 3, 1).
So, this is the solⁿ to the instance of PCP.

2.

i	List A w_i	List B x_i
1	011	101
2	11	011
3	1101	110

W = ~~11010~~ 1
x = ~~1101001~~

W = 1101 011 1101 ~~1101~~
x = ~~110 101 101 101~~

There is no solution for this PCP.

3.

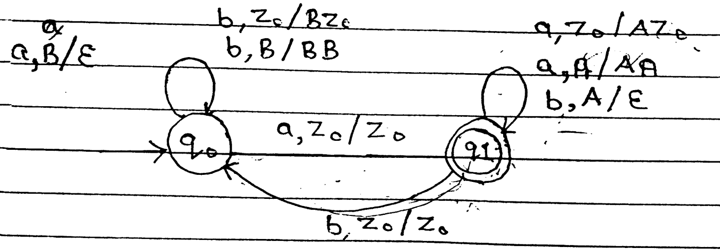
i	List A w_i	List B x_i
1	11	111
2	100	001
3	111	11

W = 11 111 m = 2
x = 111 11

(1, 3) is the solution.
(1, 2, 3) " "

→ Problems on PDA:
1. Construct a PDA for
$L = \{ x \in (a,b)^* \mid n_a(x) > n_b(x) \}$.

Solⁿ: L = {a, aab, aba, baa, ...}. ab abaaa

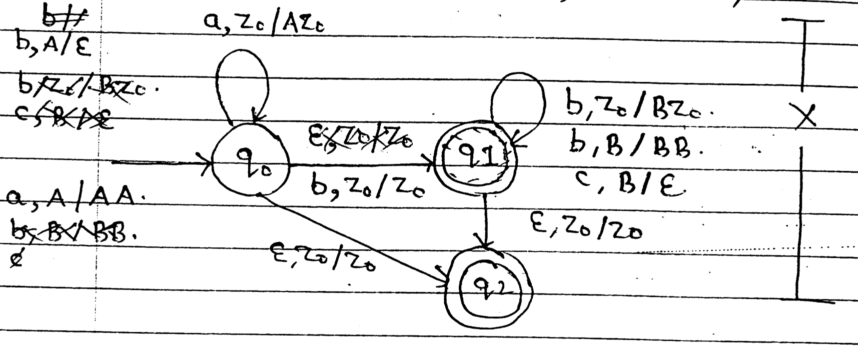


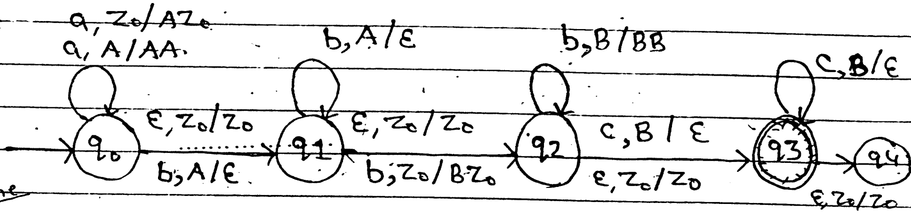
ID: abaabab

$(q_0, abaabab) \vdash (q_1, baabab, Z_0)$
$\vdash (q_0, aabab, Z_0)$
$\vdash (q_1, abab, Z_0)$
$\vdash (q_1, bab, AZ_0)$
$\vdash (q_1, ab, Z_0)$
$\vdash (q_1, b, AZ_0)$
$\vdash (q_1, \varepsilon, Z_0)$

2. $L = \{ a^n b^{n+m} c^m \mid n, m \geq 0 \}$.

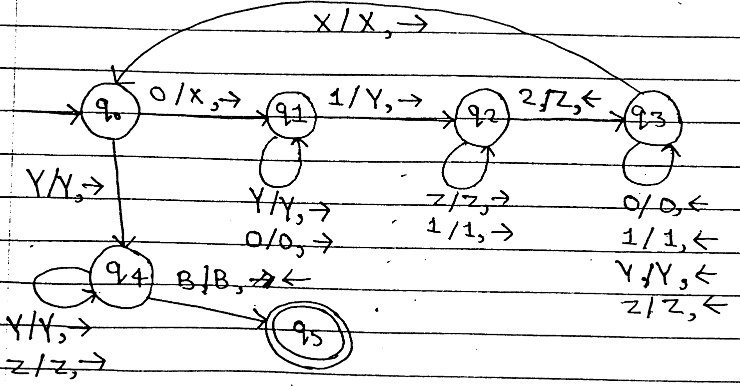
$L = \{ \varepsilon, bc, ab, abbc, abbbcc, aabbbc, \ldots \}$.



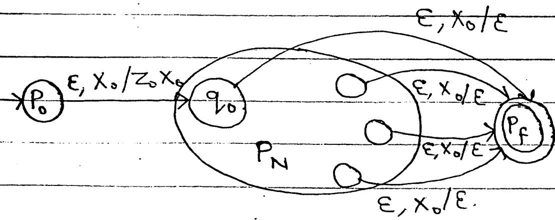


Turing Machine

Q. $L = \{ 0^n 1^n 2^n \mid n \geq 1 \}$



1.

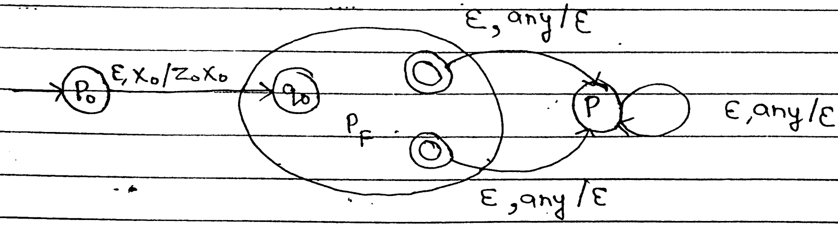


P_F simulates P_N

$P_F = (Q \cup \{P_0, P_F\}, \Sigma, \Gamma \cup \{X_0\}, \delta_F, P_0, X_0, \{P_F\})$

⇒ If $L = N(P_N)$ for some PDA, $P_N = (Q, \Sigma, \Gamma, \delta_N, q_0, Z_0)$ then there is a PDA P_F such that $L = L(P_F)$.

2.



P_N simulates P_F.

1. $(P_0, w, X_0) \vdash (q_0, w, Z_0X_0) \overset{*}{\underset{P_F}{\vdash}} (q, \varepsilon, X_0) \underset{P_F}{\vdash} (P_f, \varepsilon, \varepsilon)$

2. Let, $L = L(P_F)$ for some PDA, $P_F = (Q, \Sigma, \Gamma, \delta_F, q_0, Z_0, F)$ then there is a PDA P_N such that $L = N(P_N)$.

$P_N = (Q \cup \{P_0, P\}, \Sigma, \Gamma \cup \{X_0\}, \delta_N, P_0, X_0)$

$(P_0, w, X_0) \vdash (q_0, wZ_0X_0) \overset{*}{\underset{P_N}{\vdash}} (q, \varepsilon, \alpha X_0) \overset{*}{\underset{P_N}{\vdash}} (P, \varepsilon, \varepsilon)$

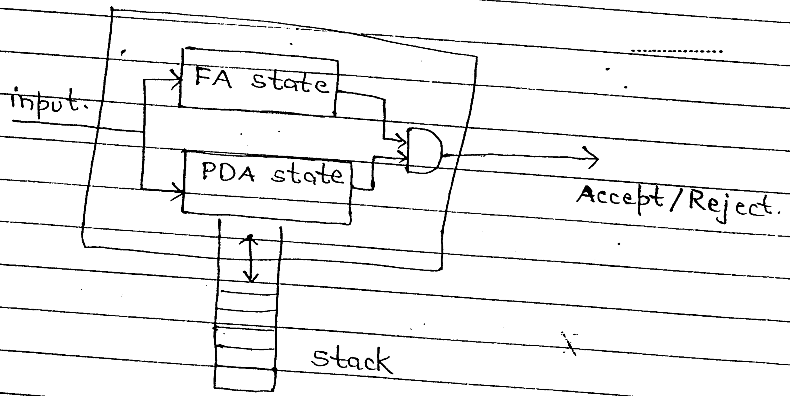
Context Free Language (CFL) :

1. If L is a CFL, then so is L^R.

Let, $L = L(G)$ for some CFG, $G = (V, T, P, S)$. Construct $G^R = (V, T, P^R, S)$ where P^R is reverse of each production in P. i.e. If $A \to \alpha$ is a production of G then.

2. If L is a CFL & R is a regular lang. then $L \cap R$ is a CFL.

roof :



$P = (Q_P, \Sigma, \Gamma, \delta_P, q_P, Z_0, F_P)$: CFL

$A = (Q_A, \Sigma, \delta_A, q_A, F_A)$: Regular Lang.

$P' = (Q_P \times Q_A, \Sigma, \Gamma, \delta, (q_P, q_A), Z_0, (F_P \times F_A))$: $L \cap R$

$(q_P, w, Z_0) \overset{*}{\underset{P}{\vdash}} (q, \varepsilon, \nu)$ if and only if

$((q_P, q_A), w, Z_0) \overset{*}{\underset{P'}{\vdash}} ((q, p), \varepsilon, \nu)$ where $p = \hat{\delta}_A(p_A, w)$ q_A