Thadomal Shahani Engineering College

Bandra (W.), Mumbai- 400 050.

CERTIFICATE

Certify that Mr./~~Miss~~ OM ANIRUDHA SHETE of COMPUTER Department, Semester V with Roll No. 2103163 has completed a course of the necessary experiments in the subject Theory of Computer Scienc under my supervision in the **Thadomal Shahani Engineering College** Laboratory in the year 2023 - 2024

27/10/23

Teacher In-Charge

Head of the Department

Date 27-10-2023

Principal

THADOMAL SHAHANI TSEC ENGINEERING COLLEGE

CONTENTS

Om Shete
THADOMAL SHAHANI TSEC ENGINEERING COLLEGE
TE C32
2103163

DFA (A)

Assignment No : 1

Q.1 Design DFA to determine whether ternary number (base 3) is divisible by 5.

⇒ Sol^ :-

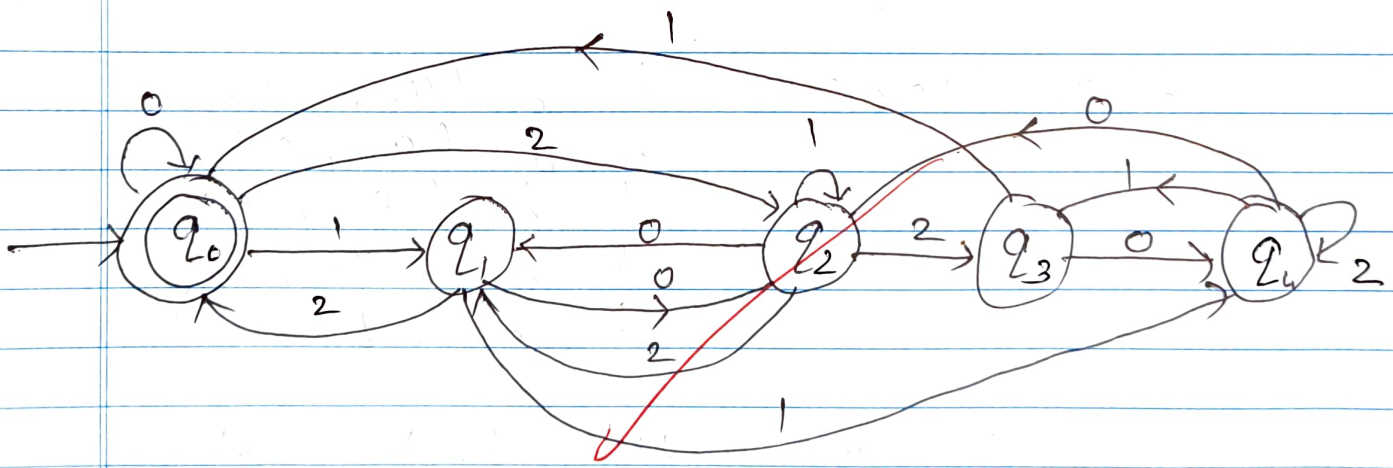
A ternary system has three alphabets

$$\Sigma = \{0, 1, 2\}$$

Base of ternary number is 3.

The running remainder could be :

$(0)_3 = 0 \rightarrow$ associated state, q_0
$(1)_3 = 1 \rightarrow$ associated state, q_1
$(2)_3 = 2 \rightarrow$ associated state, q_2
$(10)_3 = 3 \rightarrow$ associated state, q_3
$(11)_3 = 4 \rightarrow$ associated state, q_4



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

$Q = \{q_0, q_1, q_2, q_3, q_4\}$

$\Sigma = \{0, 1, 2\}$

$q_0 = q_0$

$F = q_0$

- δ (Transition table) :

Q \ Σ	0	1	2
q_0	q_0	q_1	q_2
q_1	q_3	q_4	q_0
q_2	q_1	q_2	q_3
q_3	q_4	q_0	q_1
q_4	q_2	q_3	q_4

- <u>Simulation</u>

Input string : 12021

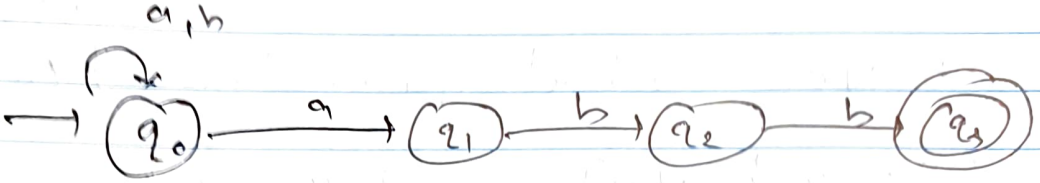
$\delta(q_0, 12021) \vdash \delta(q_1, 2021)$
$\vdash \delta(q_0, 021)$
$\vdash \delta(q_0, 21)$
$\vdash \delta(q_2, 1)$
$\vdash \delta(q_1, \varepsilon)$

$\therefore q_1$ is not a final state.
$\therefore$ 12021 is not accepted by DFA.

THADOMAL SHAHANI TSEC ENGINEERING COLLEGE

Q.2 Construct NFA that accepts set of all string over {a, b} ending with 'abb'. Convert this NFA to equivalent DFA.

→ Soln :-



NFA can be defined as-

$M = \{Q, \Sigma, \delta, q_0, F\}$

$Q = \{q_0, q_1, q_2, q_3\}$

$q_0 = q_0$

$F = \{q_3\}$

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

$\delta \Rightarrow$

	a	b
q_0	$\{q_0, q_1\}$	$\{q_0\}$
q_1	$\emptyset$	$\{q_2\}$
q_2	$\emptyset$	$\{q_3\}$
q_3	$\emptyset$	$\emptyset$

- NFA to DFA

⇒

Step I : Take $\{q_0\}$ as the initial state

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

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

Step 2: New subset generated $\{q_0, q_1\}$

$\delta(\{q_0, q_1\} \cup a) = \{q_0, q_1\}$
$\delta(\{q_0, q_1\} \cup b) = \{q_0, q_2\}$ → new state

Step 3: New subset generated $\{q_0, q_2\}$

$\delta(\{q_0, q_2\} \cup a) = \{q_0, q_1\}$
$\delta(\{q_0, q_2\} \cup b) = \{q_0, q_3\}$

Step 4: $\{q_0, q_3\}$

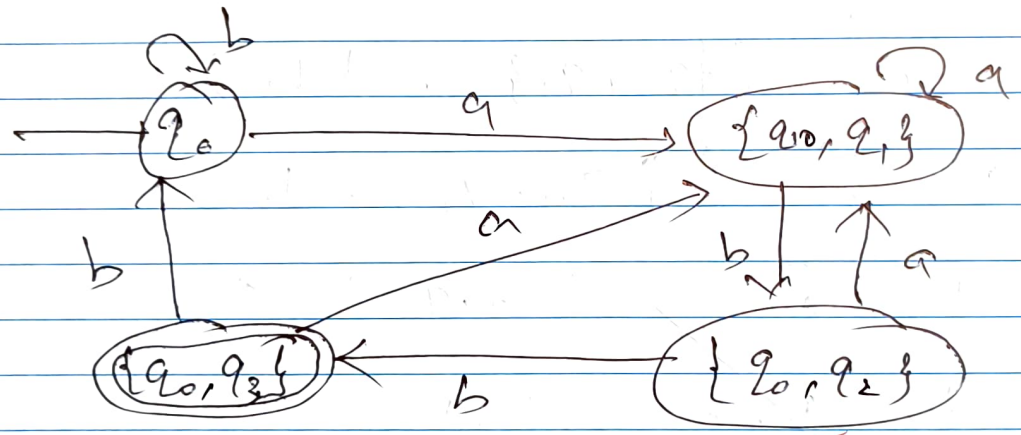
$\delta(\{q_0, q_3\} \cup a) = \{q_0, q_1\}$
$\delta(\{q_0, q_3\} \cup b) = \{q_0\}$

No new state generated

Step 5: New transition table.

Q \ Σ	a	b
q_0	$\{q_0, q_1\}$	$\{q_0\}$
$\{q_0, q_1\}$	$\{q_0, q_1\}$	$\{q_0, q_2\}$
$\{q_0, q_2\}$	$\{q_0, q_1\}$	$\{q_0\}$
$\{q_0, q_3\}$	$\{q_0, q_1\}$	$\{q_0, q_2\}$

Transition diagram:



Above DFA can be represented as -

$M' = \{Q', \Sigma', \delta', q_0', F'\}$

$Q' = \{\{q_0\}, \{q_0, q_1\}, \{q_0, q_2\}, \{q_0, q_3\}\}$

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

$q_0' = q_0$

$F' = \{q_0, q_3\}$

- Simulation

⇒ a ababb

$\delta'(q_0, aababb)$

$\vdash \delta'(\{q_0, q_1\}, ababb)$

$\vdash \delta'(\{q_0, q_1\}, babb)$

$\vdash \delta'(\{q_0, q_2\}, abb)$

$\vdash \delta'(\{q_0, q_1\}, bb)$

$\vdash \delta'(\{q_0, q_2\}, b)$

$\vdash \delta'(\{q_0, q_3\}, \in)$

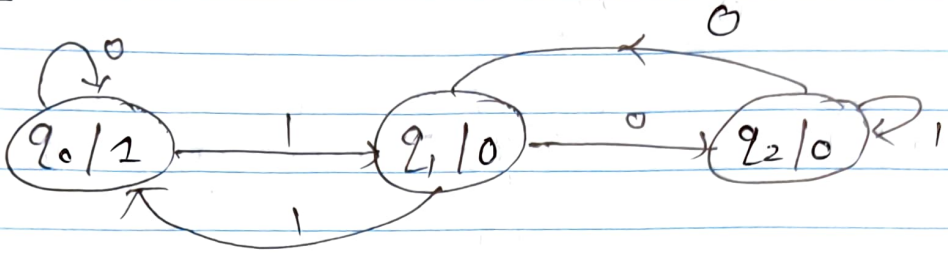
$\vdash$ Final state.

Hence, input string accepted.

THADOMAL SHAHANI
TSEC
ENGINEERING COLLEGE

Q.3 Construct Moore Machine to find out the residue modulo-3 binary numbers.

⇒ Soln :-



- State q_0 is the running remainder as 0.
- State q_1 is the running remainder as 1
- State q_2 is for the running remainder as 2.

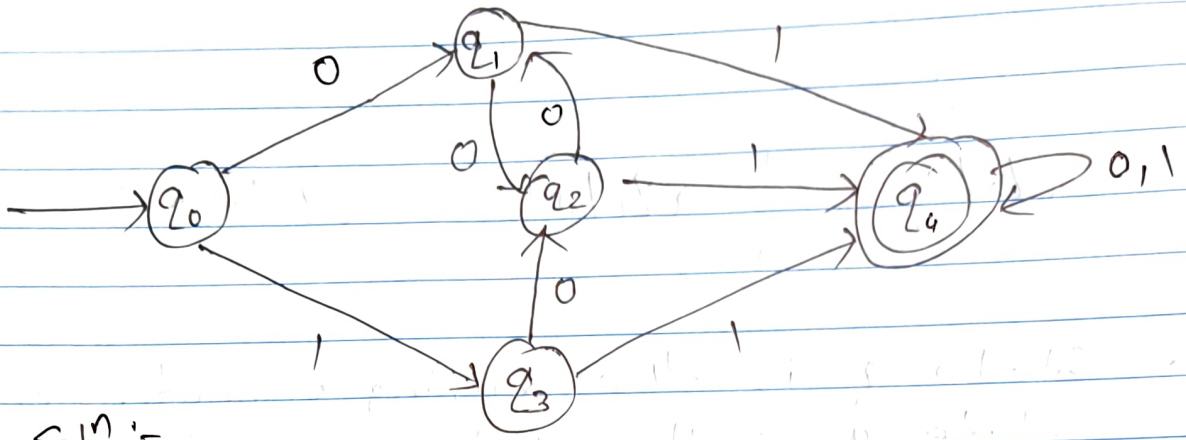
Output 1 indicates divisibility by 3
Output 0 indicates that the number is not divisible by 3.

∴ Required regular expression -

⇒

$$(0 + (1+01)^* 00)^*$$

Q.4 Minimise following DFA.

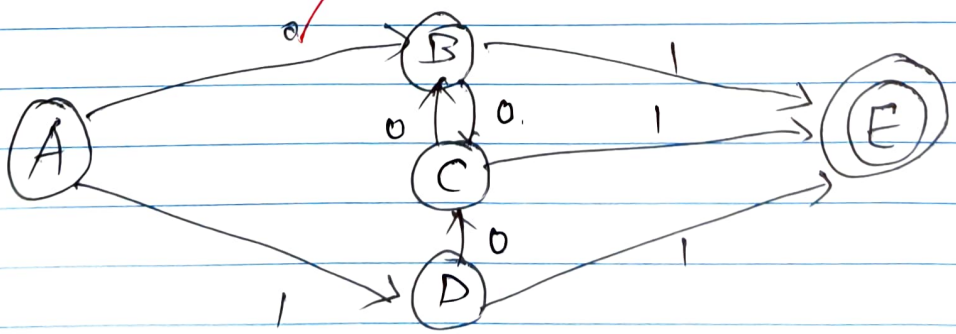


⇒ Soln :-

Step I : Draw Transition Table

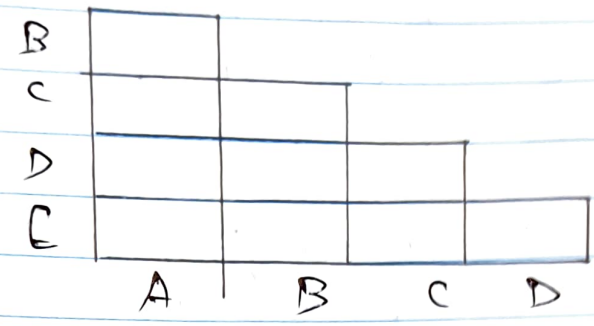
δ	0	1
q_0	q_1	q_3
q_1	q_2	q_4
q_2	q_1	q_4
q_3	q_2	q_4
q_4	q_4	q_4

- Consider $q_0 = A$, $q_1 = B$, $q_2 = C$, $q_3 = D$, $q_4 = E$
- The diagram is -



THADOMAL SHAHANI TSEC ENGINEERING COLLEGE

Step 2:



Mark all final state & non final.
(B, E), (C, E), (D, E).

Step 3: Process all the states.

1) (A, E)

⇒ $\delta(A, 0) = B$, $\delta(E, 0) = E$
$\delta(A, 1) = E$, $\delta(E, 1) = E$

2) (B, E)

⇒ $\delta(B, 0) = C$, $\delta(E, 0) = E$
$\delta(B, 1) = E$ $\delta(E, 1) = E$

Not equivalent

3) (C, E)

⇒ $\delta(C, 0) = B$, $\delta(E, 0) = E$
$\delta(C, 1) = E$, $\delta(E, 1) = E$

4) $\delta(D, E)$

i) $\delta(D, 0) = C$, $\delta(E, 0) = E$

$\delta(D, 1) = E$, $\delta(E, 1) = E$

5) (A, D)

i) $\delta(A, 0) = B$, $\delta(D, 0) = D$

$\delta(A, 1) = C$, $\delta(D, 1) = E$

6) (B, D)

i) $\delta(B, 0) = \{C, E\}$, $\delta(D, 0) = E$

$\delta(B, 1) = C$, $\delta(D, 1) = E$

Equivalent

7) (E, D)

i) $\delta(C, 0) = B$, $\delta(D, 0) = E$

$\delta(C, 1) = C$, $\delta(D, 1) = E$

Equivalent

8) (E, A)

i) $\delta(C, 0) = B$, $\delta(C, 0) = E$

$\delta(C, 1) = B$, $\delta(C, 1) = E$

9) (B, C)

i) $\delta(B, 0) = E$, $\delta(C, 0) = E$

$\delta(B, 1) = B$, $\delta(C, 1) = E$

10) (A, B)

i)

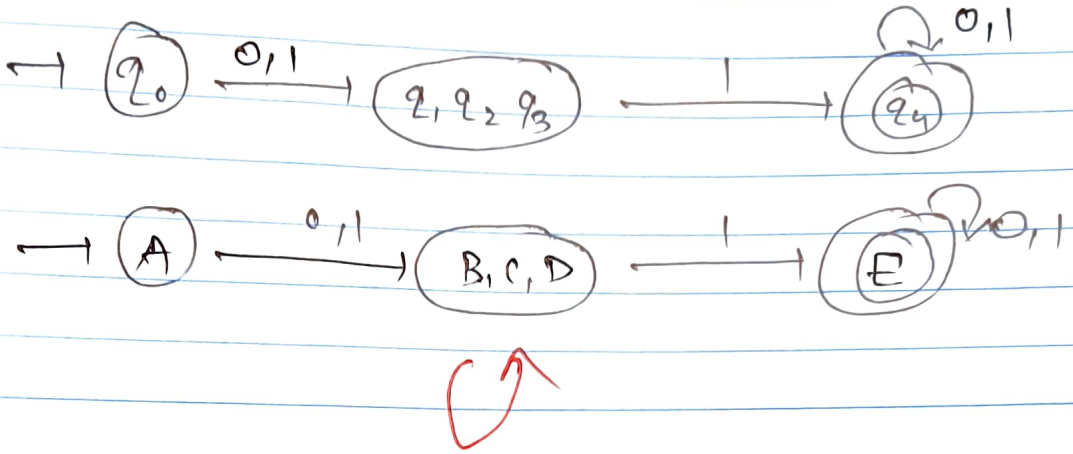
$\delta(A, 0) = B$, $\delta(B, 0) = E$

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

THADOMAL SHAHANI
TSEC
ENGINEERING COLLEGE

- Minimised DFA

⇒



Om Shete
THADOMAL SHAHANI TSEC ENGINEERING COLLEGE

TCS 163

VA BT

Assignment No. 2

Q.1. Give regular expression for
a. Set of all strings over {0,1} that end with 1 has no substring 00
b. Set of all strings over {0,1} with even number's followed by an odd number.

⇒ Soln :-

a) Strings with end 1 and no substring '00'
⇒

$$R.E. = (0+1)^* (1+01)^*$$

$(0+1)^*$ ⇒ matches any no. of 0's and 1's
$(1+01)^*$ ⇒ allows for 1 or 01 to occur any no. of times, ensuring there's no substring '00' within the string.

b) Strings containing even no. followed by odd no.
⇒

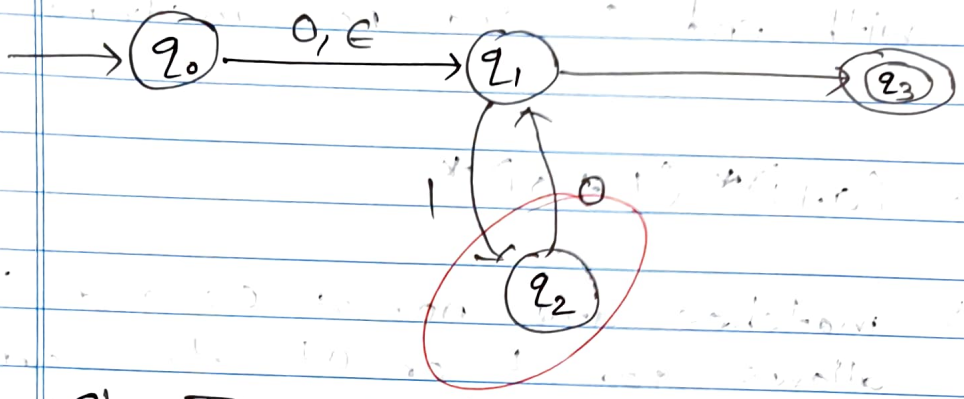
$$R.E. = (01+10)^* 0$$

$(01+10)^*$ ⇒ Matches pairs of 01 or 10 any no. of times, representing an even no. of 1's.
(0) ⇒ ensures that string ends with an odd no. of 0's.

Q.2 Convert $(0+\epsilon)(10)^*(\epsilon+1)$ into NFA with ϵ-moves and hence obtain DFA.

⇒ Sol^n :-

Step I: NFA for the given expression

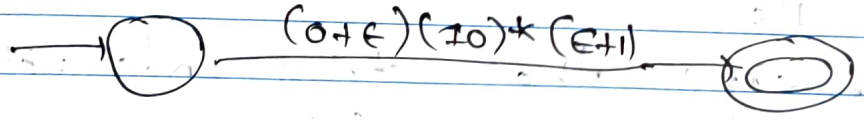


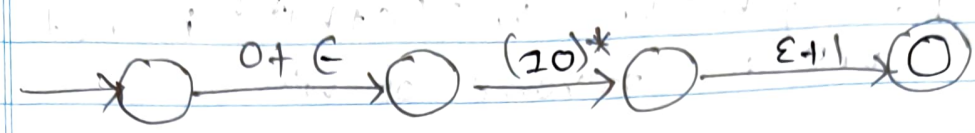
Step II: ϵ-closure of states

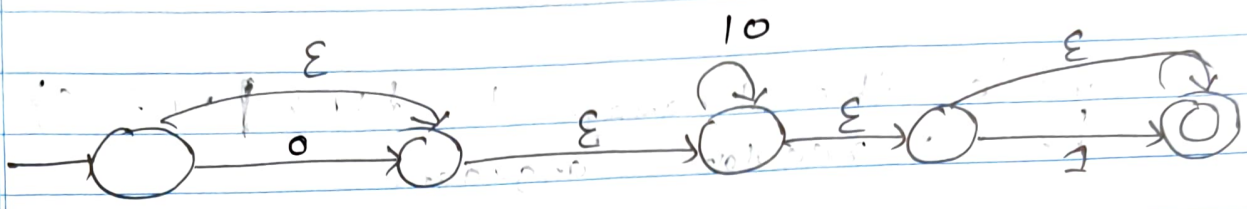
$q_0 \rightarrow \{q_0, q_1, q_3\}$
$q_1 \rightarrow \{q_1, q_3\}$
$q_2 \rightarrow \{q_2\}$
$q_3 \rightarrow \{q_3\}$

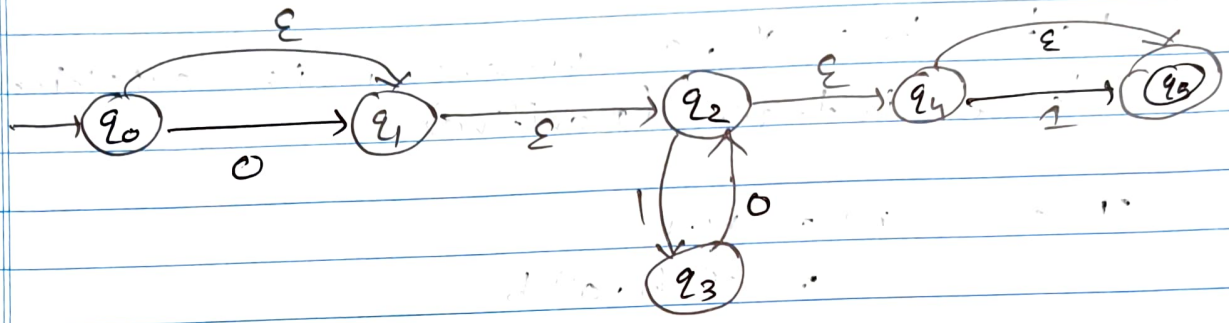
Step I in detail:
RE to NFA

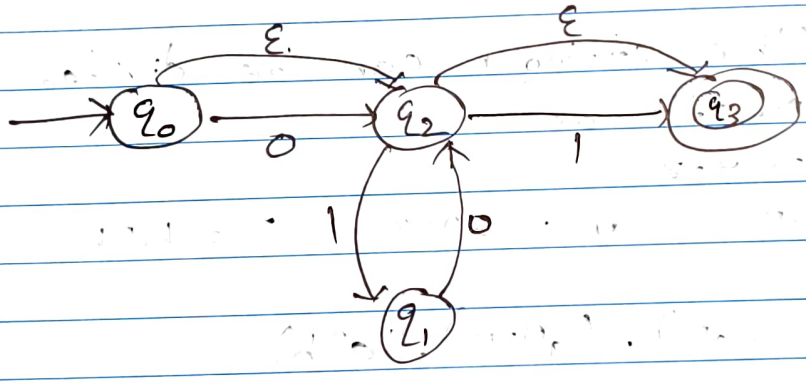
⇒



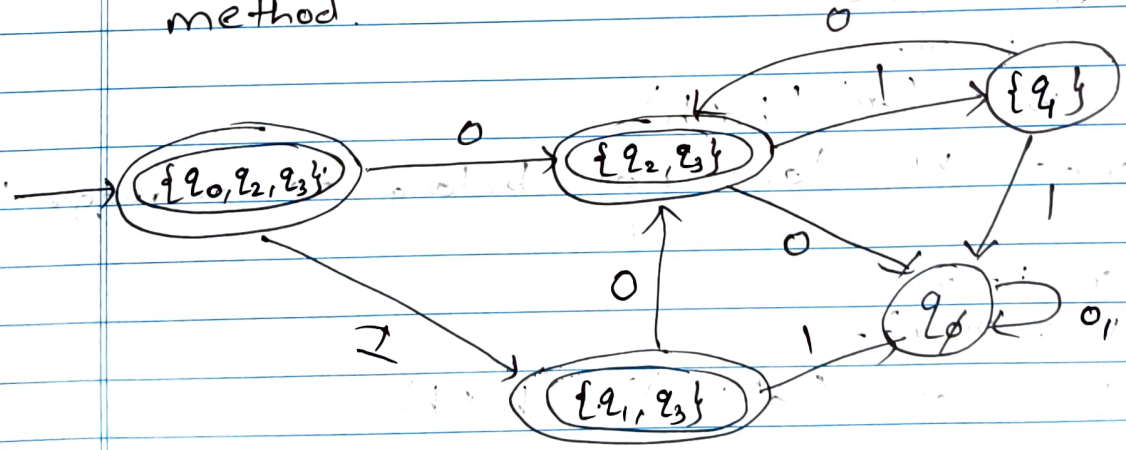








Step III : NFA to DFA using direct method



Q.3 Prove that $\{ W C W^R \mid W \in (a+b)^* \}$ is not regular where W^R is reverse of W

⇒ Soln :-

Consider the language $L = \{ W, W \mid W \in C \}$ be a regular language.

Let $w = a^n . a$ where n is sufficient large integer (positive).

Then $m = w.w$
$= a^n . b . a^n b$

Case 1 : The string n can be divided into three paste as

$m = x.y.z$

$x = a^m$, $y = a^{n-m}$, $z = b a^n b$

Hence,

$w = (a^m) . (a^{n-m}) . (b a^n b)$

∴ By pumming lemma $w = xyz$ where $i >= 0$

when $i = 0$

$w = x.z$

$w = (a^m)(b a^n b)$

Hence w does not belong to language L

when $i = 2$

$w = x y^2 z$

$\therefore w = a^m (a^{n-m})^2 . (b a^n b)$

THADOMAL SHAHANI TSEC ENGINEERING COLLEGE

$W = (a^m)\ (a^{2n-2m})\ (b\, a^n b)$

$= a^{2n-m} \cdot b \cdot a^n b$

∴ w does not belong to language L

Case II :

$w = x\,y\,z$

let $x = a^n$, $y = 1$, & $z = a^n b$

Hence $w = (a^n)\ (b)\ (a^n b)$

By pumming lemma, $w = x\,y^i\,z$, where $i >= 0$

when $i = 0$

$w = x \cdot z$

$w = a^n \cdot (a^n b)$

Hence w does not belong to language L

when $i = 2$

$w = x\,y^2\,z$

$= a^n\ (b)^2\ (a^n b)$

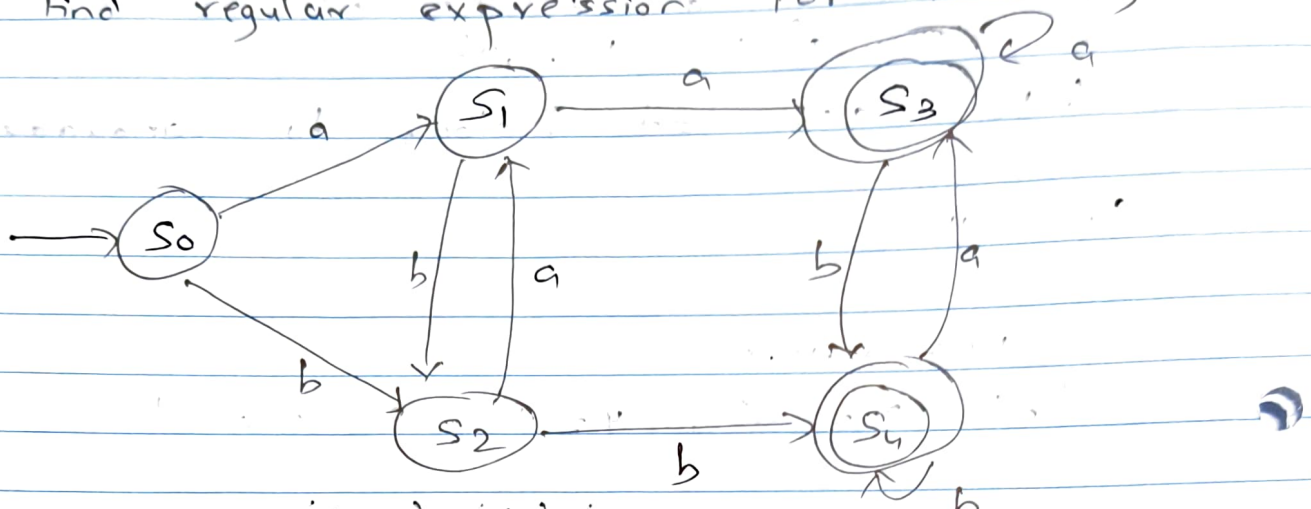
$= a^n b^2 a^n b$

Hence w does not belong to L.

∴ From case I and II, we can say language L is regular is not true.

∴ Given language L is not a regular language.

Q.4 Find regular expression for following FA



⇒ Soln :-

Equations can be written as:

$$S_0 = \varepsilon \quad -(1)$$
$$S_1 = S_0 \cdot a + S_2 \cdot a \quad -(2)$$
$$S_2 = S_0 \cdot b + S_1 \cdot b \quad -(3)$$
$$S_3 = S_1 \cdot a + S_4 \cdot a + S_3 \cdot a \quad -(4)$$
$$S_4 = S_2 \cdot b + S_4 \cdot b + S_3 \cdot b \quad -(5)$$

Putting $S_0 = \varepsilon$ in eq (1) and (2) we get

$$S_1 = \varepsilon \cdot a + S_2 \cdot a = S_2 \cdot a + \varepsilon \cdot a \quad -(6)$$
$$S_2 = \varepsilon \cdot b + S_1 \cdot b = S_1 \cdot b + \varepsilon \cdot b \quad -(7)$$

- Given DFA can be described as -

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

where

$Q = \{S_0, S_1, S_2, S_3, S_4\}$

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

$q_0 = \{S_0\}$

$F = \{S_3, S_4\}$

- Transition function can be written as -

$\delta(S_0, a) = S_1$, $\delta(S_0, b) = S_2$

$\delta(S_1, a) = S_3$, $\delta(S_1, b) = S_2$

$\delta(S_2, a) = S_1$, $\delta(S_2, b) = S_4$

$\delta(S_3, a) = S_3$, $\delta(S_3, b) = S_4$

$\delta(S_4, a) = S_3$, $\delta(S_4, b) = S_4$

- Grammar can be mapped as $G(V, T, S, P)$

$V = \{S_0, S_1, S_2, S_3, S_4\}$

$T = \{a, b\}$

$S = S_0$

- Production rules can be written as -

$S_0 \rightarrow aS_1$ $S_0 \rightarrow bS_2$

$S_1 \rightarrow a$ $S_1 \rightarrow bS_2$

$S_2 \rightarrow aS_1$ $S_2 \rightarrow b$

$S_3 \rightarrow a$ $S_3 \rightarrow b$

$S_4 \rightarrow a$ $S_4 \rightarrow b$

∴ Final Production rules :-

$S_0 \rightarrow aS_1 \mid bS_2$
$S_1 \rightarrow a \mid bS_2$
$S_2 \rightarrow aS_1 \mid b$
$S_3 \rightarrow a \mid b$
$S_4 \rightarrow a \mid b$

27/09/2023

Om Shete
THADOMAL SHAHANI TE C3
TSEC 2103163
ENGINEERING COLLEGE

28/10

Assignment No. 3

Q.1 Explain chomskey Hierarchy in detail.

⇒ • A grammer can be classified on the basis of production rules.

• Chomskey classified grammers into the following types :

1. Type 3 : Regular Grammer
2. Type 2 : Context Free Grammer
3. Type 1 : Context Sensitive Grammer
4. Type 0 : Unrestricted Grammer.

• Type 3 OR Regular Grammer

⇒ A grammer is called type 3 or regular grammer if all its production are of the following forms :

$$A \rightarrow \varepsilon$$
$$A \rightarrow a$$
$$A \rightarrow aB$$
$$B \rightarrow Ba$$

Where, $a \in \Sigma$ and $A, B \in V$

→ A language generated by Type 3 grammer is known as regular language.

• Type 2 OR Context Free Grammer

⇒ A grammer is called Type 2 or context free gramma if all its production are of following form

$$A \rightarrow \alpha$$

Where , $A \in V$ and $\alpha \in (V \cup T)^*$

V is set of variables and T is set of terminals.

→ The language generated by Type 2 grammer is called as a context Free language, a regular language but not the reverse.

• Type 1 OR Context Sensitive Grammer

⇒ A grammer is called a Type 1 or context sensitive grammer if all its production are of the following form –

$$\alpha \rightarrow \beta ,$$

Where, β is atleast as long as α

- Type 0 or Unrestricted Grammer

⇒ Productions can be written without any restriction in a unrestricted grammer.

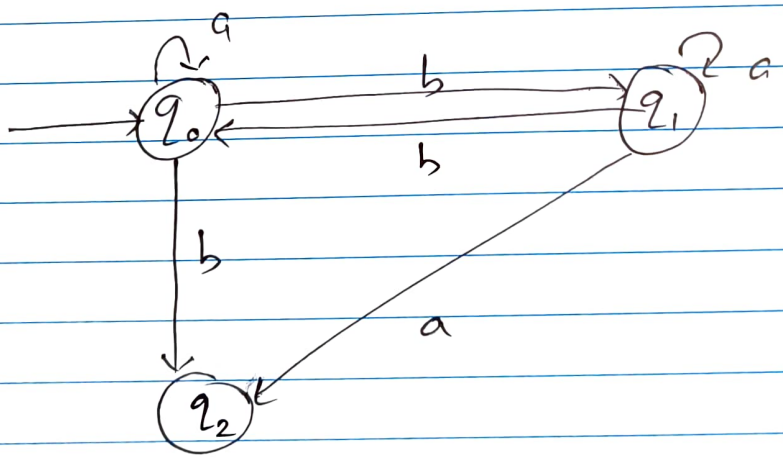
- If there is production of the $\alpha \to \beta$, the length of α could be more than length of β.
- Every grammer also is a Type 0 Grammer.
- A Type 2 Grammer is also a Type 1 Grammer
- A Type 3 Grammer is also a Type 2 Grammer.

Q.2 Construct Finite Automata recognize L(G) where G is grammer given by -

$S \to aS \mid bA \mid b$

$A \to aA \mid bS \mid a$

⇒ Soln :-



We can write Automata as -

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

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

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

$q_0 = q_0$

$F = q_2$

Transition table (δ)

⇒

	a	b
q_0	$\{q_0\}$	$\{q_1, q_2\}$
q_1	$\{q_1, q_2\}$	$\{q_0\}$
q_2	$\emptyset$	$\emptyset$

THADOMAL SHAHANI TSEC ENGINEERING COLLEGE

Q.3 Consider the following grammar :

S → icts | ictses | a

C → b

⇒ Soln :-

For string 'ibtibtaea' find following -

i) LMD

ii) RMD

iii) Parse Tree

iv) Check if above grammar is abiguous

⇒ Soln :-

i) LMD

⇒

S → icts .... [S → icts]

S → ibts .... [C → b]

S → ibticts es .... [S → ictses]

S → ibtibtses .... [C → b]

S → ibtibtaes .... [S → a]

S → ibtibtaea .... [S → a]

ii) RMD

⇒

S → icts .... [S → icts]

S → ictictses .... [S → ictses]

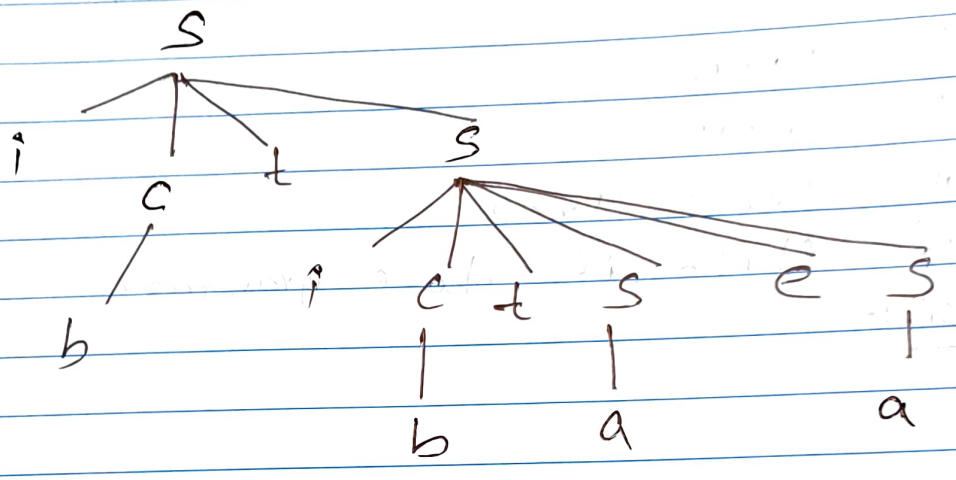
S → ictictsea .... [S → a]

S → ictictaea .... [S → a]

S → iCtebtaea ..... [C → b]
S → ibtebtaea ..... [C → b]

(iii) Parse Tree

⇒



(iv) The above grammar is ambiguous grammar due to laughing if problem.

THADOMAL SHAHANI TSEC ENGINEERING COLLEGE

Q.4 Convert the following Grammar to CNF form :

$S \rightarrow ABA$

$A \rightarrow aA \mid bA \mid \epsilon$

$B \rightarrow bB \mid aA \mid \epsilon$

Soln :-

1. The non-terminals $\{S, A, B\}$ are nullable. Null productions are removed. The resulting grammer is :

$S \rightarrow ABA \mid BA \mid AB \mid AA \mid A \mid B$

$A \rightarrow aA \mid bA \mid a \mid b$

$B \rightarrow bB \mid aA \mid b \mid a$

2. Removing unit productions, we get

$S \rightarrow ABA \mid BA \mid AB \mid AA \mid aA \mid bA \mid a \mid b \mid bB \mid aA$

$A \rightarrow aA \mid bA \mid a \mid b$

$B \rightarrow bB \mid aA \mid b \mid a$

3. Every symbol in α, in production of the form $A \rightarrow \alpha$ where $|\alpha| \geq 2$ should be a variable. This can be done by adding two productions :

$C_a \rightarrow a$

$C_b \rightarrow b$

The set of productions after above changes is :

$S \rightarrow ABA \mid BA \mid AB \mid AA \mid C_aA \mid C_bA \mid a \mid b \mid C_bB \mid C_aA$
$A \rightarrow C_aA \mid C_bA \mid a \mid b$
$B \rightarrow C_bA \mid C_aA \mid b \mid a$
$C_a \rightarrow a, \quad C_b \rightarrow b$

4. Finding an equivalent CNF :

Original Production	Equivalent production in CNF
$S \rightarrow ABA$	$S \rightarrow AC_1, C_1 \rightarrow BA$
$S \rightarrow BA \mid AB \mid AA \mid C_aA \mid C_bA \mid a \mid b \mid C_bB \mid C_aA$	$S \rightarrow BA \mid AB \mid AA \mid C_aA \mid C_bA \mid a \mid b \mid C_bB \mid C_aA$
$A \rightarrow C_aA \mid C_bA \mid a \mid b$	$A \rightarrow C_aA \mid C_bA \mid a \mid b$
$B \rightarrow C_bB \mid C_aA \mid b \mid a$	$B \rightarrow C_bB \mid C_aA \mid b \mid a$
$C_a \rightarrow a$	$C_a \rightarrow a$
$C_b \rightarrow b$	$C_b \rightarrow b$

19/10/23

Om Shete
THADOMAL SHAHANI TSEC ENGINEERING COLLEGE
TE C3
2103163

Assignment No 4

26/10

Q.1 Explain DPDA and NPDA with help of example.

⇒

- DPAD (Deterministic Pushdown Automata)

⇒

- In a DPAD there is only one move in every situation.
- A DPDA is less powerful than NPDA. Every context free language cannot be accepted by a DPDA.
- For example, a string of the form WW^R cannot be processed by a DPDA.
- The class of a language a DPDA can accepts lies in between a regular language and CFL.
- A DPDA can be defined as -

$$M = (Q, \Sigma, \Gamma, \delta, q_0, z_0, F)$$

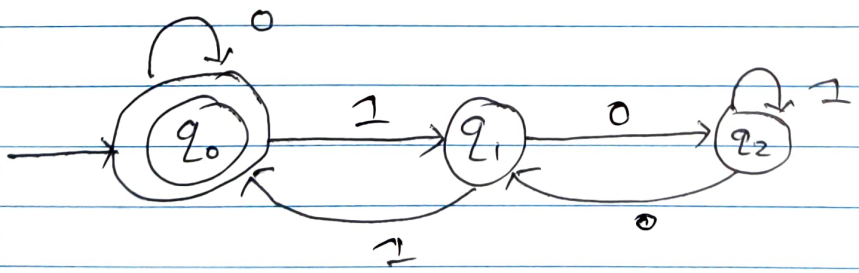
where,

$\delta(q, a, X)$ has one move for any $q \in Q$, $X \in \Gamma$ and $a \in \Sigma$.

- For e.g., DPDA for binary number divisible by 3.

⇒

- Transition diagram:



∴ DPDA is given by -

$M = [\{q_0, q_1, q_2\}, \{0, 1\}, \{0, 1, z_0\}, \delta, q_0, z_0, \{q_0\})$.

$\delta \Rightarrow$

δ / Σ	0	1
q_0	q_0	q_1
q_1	q_2	q_0
q_2	q_1	q_2

- **NPDA** (Non-deterministic PDA)

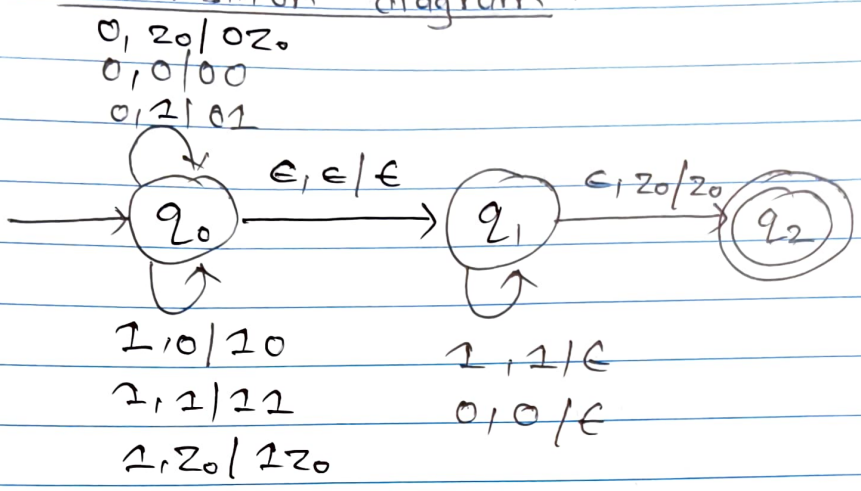
⇒

- A NPDA provides non-determinism to PDA.
- In DPDA, there is only one move in every situation. Where as, in case of NPDA there could be multiple moves under a situation.
- Every context free language cannot be recognized by a DPDA but it can be recognized by NPDA.
- The class of language a DPDA can accept lies in between a regular language and CFL.
- A palindrome can be accepted by NPDA but it can not be accepted by a DPDA.

Q. 2 Construct PDA accepting languages of Palindrome

⇒ Soln :-

- Transition diagram :



- Simulation :

I/P string : 10101

$$
\begin{aligned}
(q_0, 10101) &\vdash (q_0, 0101) \\
&\vdash (q_0, 0101) \\
&\vdash (q_0, 101) \\
&\vdash (q_0, 01) \\
&\vdash (q_0, 1) \\
&\vdash (q_0, \epsilon) \\
&\vdash (q_1, \epsilon) \\
&\vdash (q_2, \epsilon)
\end{aligned}
$$

∴ The 10101 string accepted.

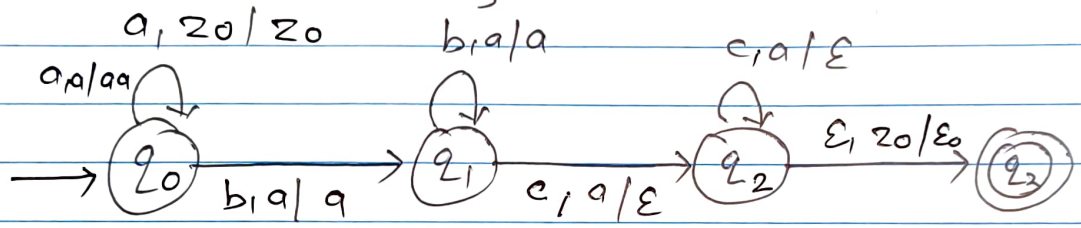
- PDA can be defined as -

$M = (\{q_0, q_1, q_2\}, \{0, 1\}, \{0, 1, Z_0\}, \delta, q_0, Z_0, \{q_2\})$.

Q.3 Construct PDA accepting the following languages $L = \{ a^n b^m c^n \mid m, n >= 1 \}$

⇒ Soln :-

- Transition diagram :



- Transition function :

$\delta(q_0, a, Z_0) = (q_0, aZ_0)$
$\delta(q_0, a, a) = (q_0, aa)$
$\delta(q_0, b, a) = (q_1, a)$
$\delta(q_1, b, a) = (q_1, a)$
$\delta(q_1, c, a) = (q_2, \varepsilon)$
$\delta(q_2, c, a) = (q_2, \varepsilon)$
$\delta(q_2, \varepsilon, Z_0) = (q_3, \varepsilon)$

- PDA can be defined as -

$M = (\{q_0, q_1, q_2, q_3\}, \{a, b, c\}, \{a, z_0\}, q_0, z_0, \{q_3\}, \delta)$

- Simulation :

$(q_0, aabbcc) \vdash (q_0, abbcc)$
$\vdash (q_0, bbcc)$
$\vdash (q_1, bcc)$
$\vdash (q_1, cc)$
$\vdash (q_2, c)$
$\vdash (q_2, \epsilon)$
$\vdash (q_3, \epsilon)$

∴ I/P string is accepted.

Q.4 Design PDA for the following CFG. Show acceptance of one of valid string

$S \rightarrow aAA$
$A \rightarrow bS$
$A \rightarrow aS$
$S \rightarrow a$

⇒ Soln :-

The equivalent PDA, M is given by

$M = (\{q\}, \{a, b\}, \{a, b, S, A\}, \delta, q, S, \phi)$

where δ is given by -

$\delta(q, \varepsilon, S) = \{(q, aAA)\}$
$\delta(q, \varepsilon, A) = \{(q, aS), (q, bS), (q, a)\}$

$\delta(q, a, a) \Rightarrow \{(q, \varepsilon)\}$
$\delta(q, b, b) \Rightarrow \{(q, \varepsilon)\}$

- Acceptance of aba^4 by M

$\delta(q, abaaaa, S) \vdash (q, baaaa, aAA)$
$\vdash (q, baaaa, AA)$
$\vdash (q, aaaa, SA)$
$\vdash (q, aaaa, SA)$
$\vdash (q, aaaa, OAAA)$
$\vdash (q, aaa, AAA)$
$\vdash (q, aaa, aAA)$
$\vdash (q, aa, AA)$
$\vdash (q, aa, OA)$
$\vdash (q, a, A)$
$\vdash (q, a, a)$
$\vdash (q, \varepsilon, \varepsilon)$.

Thus, the string aba^4 is accepted by M using an empty stack.

$\therefore aba^4 \in L$.

19/10/23

Assignment No. 5

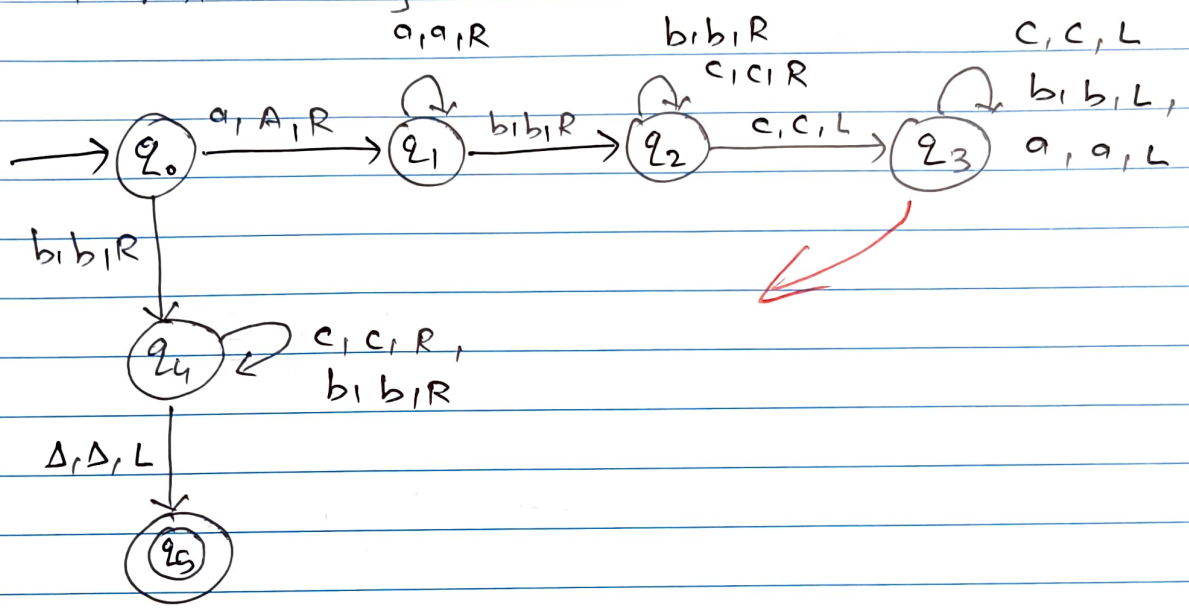
Om Shete
TE C32
2103163

Q.1 Design Turing Machine for recognizing the following language

$L = \{ a^n b^m c^n \mid m, n >= 1 \}$

⇒ Soln :-

- Transition diagram :



- Transition table :

	a	b	c	A	C	Δ
q_0	$\{q_1, A, R\}$	$\{q_4, b, R\}$				
q_1	$\{q_1, a, R\}$	$\{q_2, b, R\}$				
q_2		$\{q_2, b, R\}$	$\{q_2, c, L\}$		$\{q_2, c, R\}$	
q_3	$\{q_3, a, L\}$	$\{q_3, b, L\}$		$\{q_0, A, R\}$	$\{q_3, c, L\}$	
q_4		$\{q_2, b, R\}$			$\{q_4, c, R\}$	$\{q_5, a, L\}$
q_5						

Shete
C3

$M = \{Q, q_0, F, \Gamma, \delta, \Delta\}$

$= (\{q_0, q_1, q_2, q_3, q_4, q_5\}, q_0, \{q_5\}, \{a, b, c\}, \{a, b, c, A, C\}, \delta, \Delta)$.

- Simulation :

a a b c c
↑
q_0

$\vdash$ A a b c c
↑
q_1

$\vdash$ A a b C c
↑
q_3

$\vdash$ A a b C c
↑
q_3

$\vdash$ A A b C c
↑
q_2

$\vdash$ A A b C C
↑
q_3

$\vdash$ A A b C C
↑
q_3

$\vdash$ A A b C C
↑
q_1

$\vdash$ A A b C C
↑
q_4

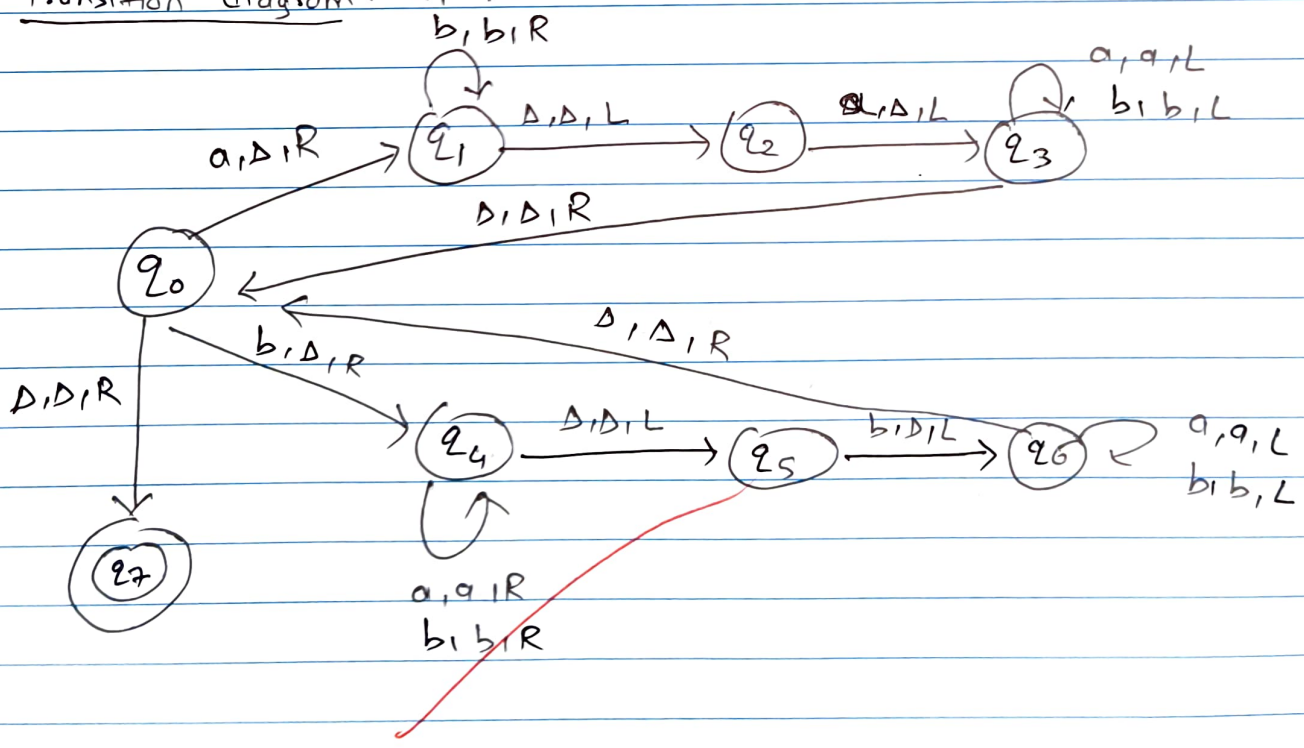
$\vdash$ A A b c c Δ
↑
q_4
$\vdash$ A A b C C Δ
↑
q_4
$\vdash$ A A b CC Δ
↑
q_5

∴ I/P string is accepted.

Q.2 Construct turing machine for checking even palindrome string over $\Sigma = \{0, 1\}$.

⇒ Soln :-

• Transition diagram:



- Transition table:

	a	b	Δ
q_0	$\{q_1, \Delta, R\}$	$\{q_4, \Delta, R\}$	$\{q_7, \Delta, R\}$
q_1	$\{q_1, a, R\}$	$\{q_1, b, R\}$	$\{q_2, \Delta, L\}$
q_2	$\{q_3, \Delta, L\}$		
q_3	$\{q_3, a, L\}$	$\{q_3, b, L\}$	$\{q_0, \Delta, R\}$
q_4	$\{q_4, a, R\}$	$\{q_4, b, R\}$	$\{q_4, \Delta, L\}$
q_5		$\{q_6, \Delta, L\}$	
q_6	$\{q_6, a, L\}$	$\{q_6, b, L\}$	$\{q_0, \Delta, R\}$
q_7			

- Simulation :

$M = \{Q, \Sigma, \Gamma, \delta, q_0, \Delta, F\}$

$= (\{q_0, q_1, q_2, q_3, q_4, q_5, q_6, q_7\}, \{a, b\}, \{a, b\}, \delta, q_0, \Delta, \{q_7\})$.

abbaΔ $\vdash$ (q_0) abba Δ

$\vdash$ Δ (q_1) bba Δ

$\vdash$ Δ b (q_1) ba Δ

$\vdash$ Δ bb (q_1) a Δ

$\vdash$ Δ bba (q_2) Δ

$\vdash$ Δ bb (q_2) a Δ

$\vdash$ Δ b (q_3) b Δ Δ

$\vdash$ Δ (q_3) b b Δ Δ

$\vdash$ Δ (q_3) Δ bb Δ Δ

$\vdash$ Δ (q_0) bb Δ Δ

$\vdash$ $\Delta\Delta$ (q_4) b Δ Δ

$\vdash$ $\Delta\Delta$ b (q_4) Δ Δ

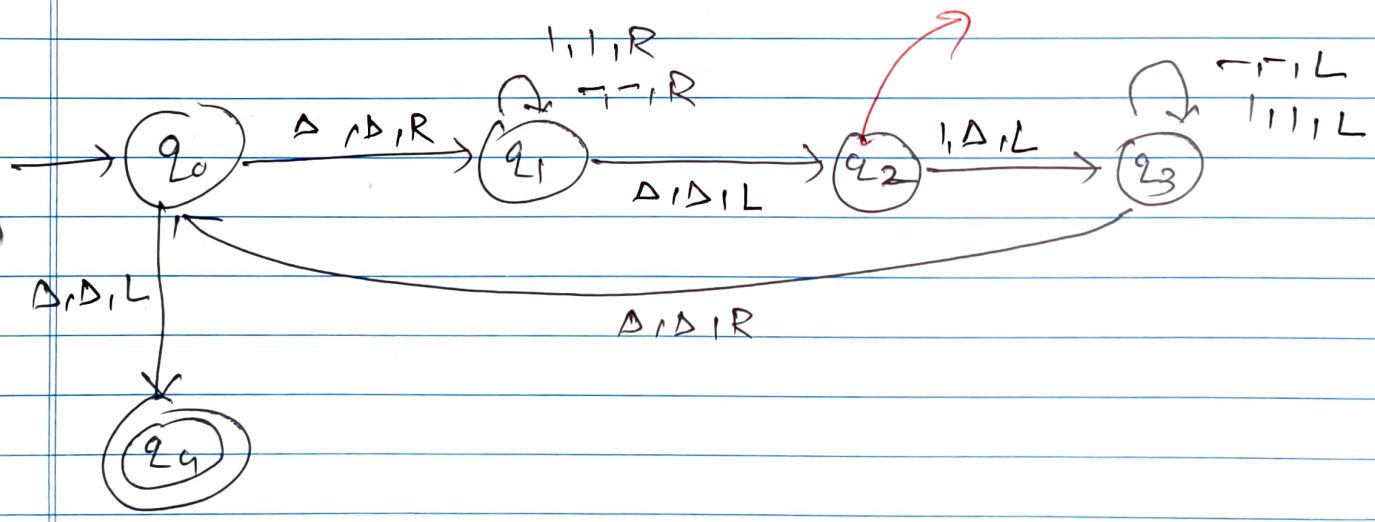
$\vdash \Delta\Delta(q_5)b\Delta\Delta$
$\vdash \Delta(q_6)\Delta b\Delta\Delta$
$\vdash \Delta\Delta(q_0)b\Delta\Delta$
$\vdash \Delta\Delta(q_0)\Delta\Delta\Delta$
$\vdash \Delta\Delta(q_7)\Delta\Delta\Delta$

$\therefore$ I/P string is accepted.

Q.3 Design TM to substract two unary numbers assume $m > n$

$\Rightarrow$ Sol^n:-

- Transition diagram :



- Transition table:

	1	-	Δ
q_0	$\{q_1, \Delta, R\}$		$\{q_4, \Delta, R\}$
q_1	$\{q_1, 1, R\}$	$\{q_1, -, R\}$	$\{q_2, \Delta, L\}$
q_2	$\{q_3, \Delta, L\}$		
q_3	$\{q_3, 1, L\}$	$\{q_3, -, L\}$	
q_4			

- Simulation:

$(q_0)111-11\Delta$

$\vdash (q_0)111-11\Delta$
$\vdash \Delta(q_1)11-11\Delta$
$\vdash \Delta 1(q_1)1-11\Delta$
$\vdash \Delta 11(q_1)-11\Delta$
$\vdash \Delta 11-1(q_1)1\Delta$
$\vdash \Delta 11-11(q_1)\Delta$
$\vdash \Delta 11-1(q_2)1\Delta$
$\vdash \Delta 11-(q_2)\Delta\Delta$
$\vdash \Delta 11(q_2)-1\Delta\Delta$
$\vdash \Delta 1(q_2)1-1\Delta\Delta$
$\vdash \Delta(q_2)11-1\Delta\Delta$
$\vdash (q_2)\Delta 11-1\Delta\Delta$
$\vdash \Delta(q_0)11-1\Delta\Delta$
$\vdash \Delta\Delta(q_1)1-1\Delta\Delta$
$\vdash \Delta\Delta 1(q_1)-1\Delta\Delta$
$\vdash \Delta\Delta 1-(q_1)1\Delta\Delta$
$\vdash \Delta\Delta 1-(q_2)1\Delta\Delta$
$\vdash \Delta\Delta 1-(q_1)\Delta\Delta\Delta$

$\vdash \Delta\Delta \vdash (q_2)\Delta\Delta$.

∴ TM can be defined as –

$M = \{Q, \Sigma, \delta, \Gamma, q_0, \Delta, F\}$

$M = (\{q_0, q_1, q_2, q_3, q_4\}, \{1, -\}, \delta, \{1, -\}, q_0, \Delta, \{q_4\})$.

Q.4 Define turing machine and explain different variants of turing machine.

⇒

- Turing machine is an abstract mathematical model of computation that was introduced by mathematical and compute scientist Alan Turing in 1930's.
- It can solve any mathematical problem.
- The prime components of turing machine are :

1) TAPE : An inficiently any tape divided into cells and holds a symbol from finite alphabet, including Δ symbol.

2) READ/WRITE head : A head that can read the symbol at its current position on tape and write a new symbol.

- Types of turing machine:

1) Standard Turing Machine

⇒ This is the original and most basic form of turing machine. It can read, write, and erase symbols on the tape, and its rules are defined by a transition function that maps the current state and the symbol under the head to a new state, new symbol to write and a direction in which to move the head.

2) Nondeterministic Turing Machine (NTM)

⇒ Unlike a standard TM, an NTM can be in multiple states at once and explore various computational paths simultaneously.

3) Universal Turing Machine (UTM)

⇒ A UTM is a turing machine capable of simulating any other turing machine.

- It has a special program on its tape that encodes the description of the machine it is simulating.
- A UTM can execute the steps of the simulated machine, effectively making it a universal computer.

19/10/23

Om Shete
THADOMAL SHAHANI TSEC ENGINEERING COLLEGE
TE C3
2103163

Assignment No. 6

Q.1 Write short note on : Recursive and Recursively Enumerable languages.

⇒

- There is a difference between recursively enumerable (Turing acceptable) and recursive (Turing Decidable) language.
- Following statements are equivalent :
  1) The language L is Turing acceptable.
  2) The language L is recursively enumerable
- Following statements are Equivalent :
  1) The language L is Turing decidable.
  2) The language L is recursive.
  3) There is an algorithm for recognizing L.
- Every turing decidable language is turing acceptable.
- Every turing acceptable language need not be turing decidable.

- **Turning Acceptable Language :**

- A language $L \subseteq \Sigma^*$ is said to be a Turing language if there is a turing machine M which halts on every $w \in L$ with an answer 'YES'.
- However, if $w \notin L$ then M may not halt.

• Turing Decidable Language:

- A language $L \subseteq \Sigma^*$ is said to be turing being decidable if there is a turing machine M which always halts on every $w \in \Sigma^*$.
- If $w \in L$ then ~~sto~~ M halts, with answer 'YES'; and if $w \notin L$ then M halts with answer 'NO'.

- A set of solutions for any problem defines a language.
- A problem P is said to be decidable / solvable if the language $L \subseteq \Sigma^*$ representing the problem is turing decidable.
- If P is solvable / decidable then there is an algorithm for recognizing L, representing the problem. It may be noted that an algorithm terminates on all inputs.
- Following statements are equivalent:

1) The language L is Turing decidable.
2) The language L is recursive.
3) There is an algorithm for recognizing L.

Q.2 What is Halting Problem ? Explain in detail.

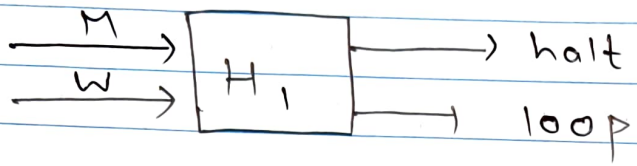
=>

- The halting problem of a turing machine states :
Given a Turing Machine M and an input w to the machine M, determine if the machine M will eventually halt when it is given input w.
- Halting problem of a turing machine is unsolvable.
- Proof :

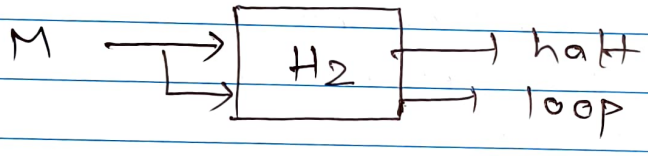
- Moves of a turing machine can be represented using a binary number. Thus, a turing machine can be represented using a string over Σ^* (0, 1). This concept has already been explained in the chapter.
- Insolvability of halting problem of a turing machine can be proved through the method of contradiction.

- Step 1 : Let us assume that the halting problem of a Turing machine is solvable. There exists a machine H (say). H takes two inputs :
  1) A string describing M.
  2) An input w for machine M.

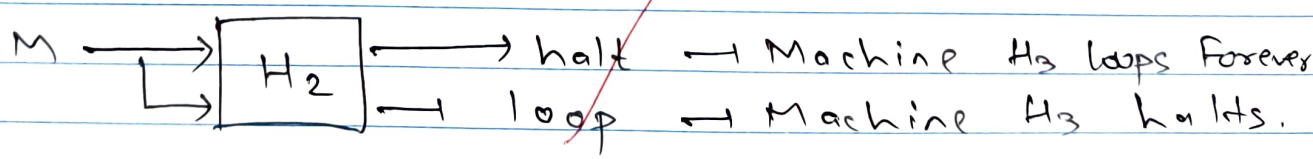
- H_1 generates an output "halt" if H_1 determines that M stops on input w; otherwise H outputs "loop". Working of the machine H_1 is shown below.



- Step 2 : Let us revise the machine H_1 as H_2 to take both inputs and H_2 should be able to determine if M will halt on M as its input.

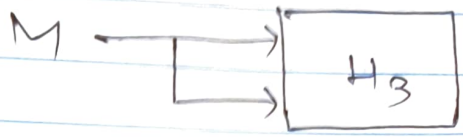


- Step 3: Let us construct a new turing machine H_3 that takes output of H_2 as input and does the following:
  1) If the output of H_2 is "loop" than H_3 halts.
  2) If the output of H_2 is "halt" than H_3 will loop forever.



H_3 will do opposite of output of H_2.

Step 4: Let us give H_3 itself as inputs to H_3.



- If H_3 halts on H_3 as input then H_3 would loop.
- If H_3 loops forever on H_3 as input H_3 halts.
- In either case, the result is wrong.
- Hence, H_3 does not exist.
- If H_3 does not exist than H_2 does not exist.
- If H_2 does not exist than H_1 does not exist.

Q. 3 Explain detailed explanation on Rice's theorem

⇒

- Every property that is satisfied by some but not all recursively enumerable language is un-decidable.
- Any property that is satisfied by some recursively enumerable language but not all is known as nontrivial property.
- We have seen many properties of R.E languages that are un-decidable. These properties include:

1) Given a TM M, is L(M) nonempty?
2) Given a TM M, is L(M) finite?
3) Given a TM M, is L(M) regular?
4) Given a TM M, is L(M) recursive?

- The rice's theorem can be proved by reducing some other unsolvable problem to nontrivial property of recursively enumerable language.
- Non-trivial Property:

⇒ A property is considered "non-trivial" if it holds for some turing machines but not for others.

- For eg., the property "contains at least one palindrome" is non-trivial because some turing machines recognizes languages with palindromes, while other do not.

- No Algorithm

⇒ Rice's theorem asserts that there's no general algorithm that can determine whether an arbitary turing machine recognizes a language with non-specific trivial property.

- Implications

⇒ This theorem has significant consequences. It shows that many interesting question about turing machines' languages are undecidable.

- Limations

⇒ Rice's theorem doesn't simplify that all questions about turing machines are undecidable.

Q.4 Define post correspondance problem. Prove that PCP with two lists $x = \{ b, bab^3, ba \}$ and $y = \{ b^3, ba, a \}$ have a solution.

⇒

- Let A and B be two non-empty lists of strings over Σ A and B are given as below:

$$A = \{ x_1, x_2, x_3, \ldots, x_k \}$$
$$B = \{ y_1, y_2, y_3, \ldots, y_k \}$$

- We say, there is a post correspondence between A and B if there is sequence of one or more integers $i, j, k, \ldots m$ such that: The string $x_i x_j \ldots x_m$ is equal to $y_i y_j \ldots y_m$.

- $X = \{ b, bab^3, ba \}$
  $Y = \{ b^3, ba, a \}$

$\Rightarrow$

We will have to find a sequence using which when the elements of A and B are listed, will produce identical string.

$\therefore$ The required string is (2, 1, 1, 3).

$\therefore$ $x_2 x_1 x_1 x_3$
$= bab^3 \cdot b \cdot b \cdot ba$
$= b^7 a^2$

$\therefore$ $y_2 y_1 y_1 y_3$
$= ba \cdot b^3 \cdot b^3 \cdot a$
$= b^7 \cdot a^2$

Thus, the PCP has solution.