Arpan Adhikari
231309
TOC UT Question Ans.

q1a) soln

Given language, $L = \{w \in \{0,1\}^* : w$ contains '0110' or '1001' as substring$\}$

Let, the NFA be $(Q, \Sigma, \delta, q_0, F)$

$Q = \{q_0, q_1, q_2, q_3, q_4, q_5, q_6, q_7\}$
$\Sigma = \{0,1\}$
$q_0 = q_0$
$F = \{q_4\}$
~~δ: Q|Σ 0 1~~

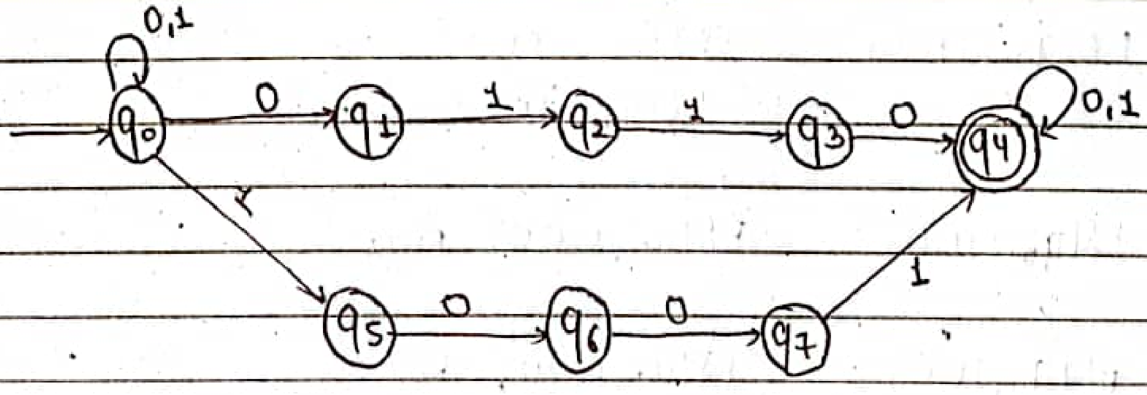


Fig. NFA State diagram

δ:

Q/Σ	0	1
q_0	$\{q_0, q_1\}$	$\{q_0, q_5\}$
q_1	-	q_2
q_2	-	q_3
q_3	q_4	-
q_4	q_4	q_4
q_5	q_6	-
q_6	q_7	-
q_7	-	q_4

Testing:

For 01011100,

$\delta(q_0, 01011100)$

$\vdash \delta(q_0, 1011100)$ $\quad \vdash \delta(q_1, 1011100)$

$\vdash \delta(q_2, 011100)$ Reject

$\vdash \delta(q_0, 011100)$ $\quad \vdash \delta(q_5, 011100)$

$\vdash \delta(q_6, 11100)$ Reject

$\vdash \delta(q_0, 11100)$ $\quad \vdash \delta(q_1, 11100)$

$\vdash \delta(q_2, 1100)$ Reject

$\vdash \delta(q_0, 1100)$ $\quad \vdash \delta(q_5, 1100)$ Reject

$\vdash \delta(q_0, 100)$ $\quad \vdash \delta(q_5, 1100)$ Reject

$\vdash \delta(q_0, 00)$ $\quad \vdash \delta(q_5, 00)$

$\vdash \delta(q_6, 00)$

~~Reject~~

$\vdash \delta(q_0, 0)$ $\quad \vdash \delta(q_1, 0)$ $\quad \vdash \delta(q_7, 0)$ Reject

$\vdash \delta(q_0, \epsilon)$ $\quad$ Reject

Reject $\quad \vdash \delta(q_1, \epsilon)$ Reject

Hence, the string is rejected.

For 1010011,

$\delta(q_0, 1010011)$

$\vdash \delta(q_0, 010011)$

$\vdash \delta(q_0, 10011)$

$\vdash \delta(q_5, 0011)$

$\vdash \delta(q_6, 011)$

$\vdash \delta(q_7, 11)$

$\vdash \delta(q_4, 1)$

$\vdash \delta(q_4, \epsilon)$ $\quad$ Accept Ans

Q1b)

Conversion of NFA to DFA:

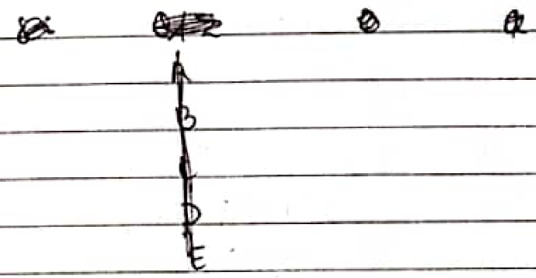
Given NFA is $(Q, \Sigma, \delta, q_0, F)$

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

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

$q_0 = A$

$F = \{C, E\}$



Finding ε-closure,

$E(A) = \{A, B, D\}$

$E(B) = \{B, D\}$

$E(C) = \{C, E\}$

Let start state of equivalent DFA is E-closure of start state of NFA
i.e. {A,B,D}

For {A,B,D},

$\delta'(\{A,B,D\},0)$
$= E(D) \cup E(C) \cup E(E)$
$= \{C,D,E\}$ N.S.

$\delta'(\{A,B,D\},1)$
$= E(A) \cup E(D) \cup E(C) \cup \phi$
$= \{A,B,D\} \cup \{D\} \cup \{C,E\} \cup \phi$
$= \{A,B,C,D,E\}$ N.S

For {C,D,E},

$\delta'(\{C,D,E\},0)$
$= \phi \cup E(E) \cup E(C)$
$= \{C,E\}$ N.S.

$\delta'(\{C,D,E\},1)$
$= \phi \cup \phi \cup E(C)$
$= \{C,E\}$

For {A,B,C,D,E},

$\delta'(\{A,B,C,D,E\},0)$
$= E(D) \cup E(C) \cup \phi \cup E(E) \cup E(C)$
$= \{D\} \cup \{C,E\} \cup \{E\} \cup \{C,E\}$
$= \{C,D,E\}$

$\delta'(\{A,B,C,D,E\},1)$
$= E(A) \cup E(D) \cup E(C) \cup \phi \cup \phi \cup E(C)$
$= \{A,B,D\} \cup \{D\} \cup \{C,E\} \cup \{C,E\}$
$= \{A,B,C,D,E\}$

For {C,E},

$\delta'(\{C,E\},0)$
$= \phi \cup E(C)$
$= \{C,E\}$

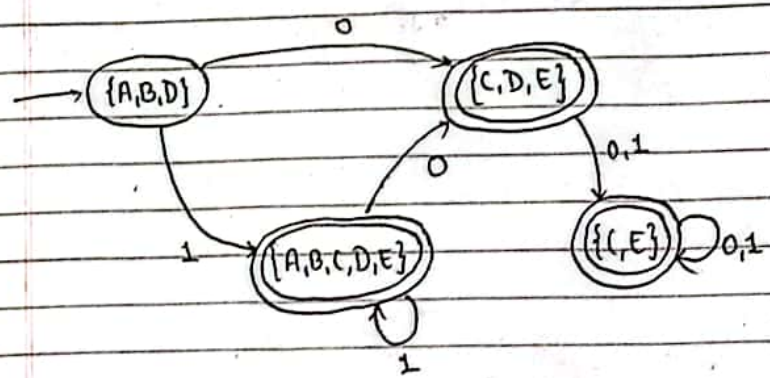
$\delta'(\{C,E\},1)$
$= \phi \cup E(C)$
$= \{C,E\}$

No more new states. so, process terminates.

Now, equivalent DFA be $D = (Q', \Sigma', \delta', q_0', F')$
where,
$Q' = \{\{A,B,D\},\{C,D,E\},\{A,B,C,D,E\},\{C,E\}\}$
$\Sigma' = \{0,1\}$
$q_0' = \{A,B,D\}$
$F' = \{\{C,D,E\},\{A,B,C,D,E\},\{C,E\}\}$

δ':

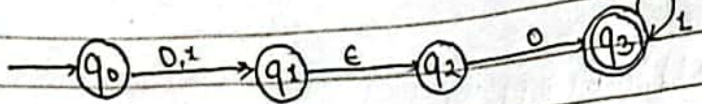
Q'/Σ'	0	1
{A,B,D}	{C,D,E}	{A,B,C,D,E}
{C,D,E}	{C,E}	{C,E}
{A,B,C,D,E}	{C,D,E}	{A,B,C,D,E}
{C,E}	{C,E}	{C,E}



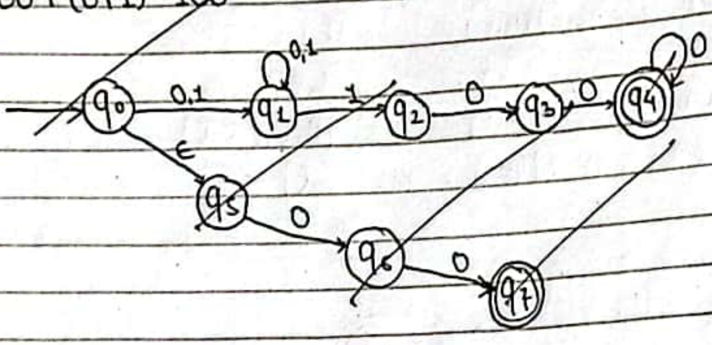
state diagram

Q2a) Conver RE to E-NFA.

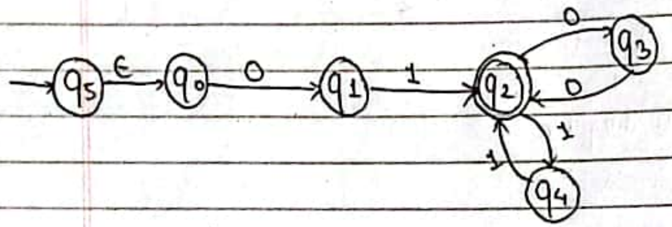
(i) $(0+1)01^*$



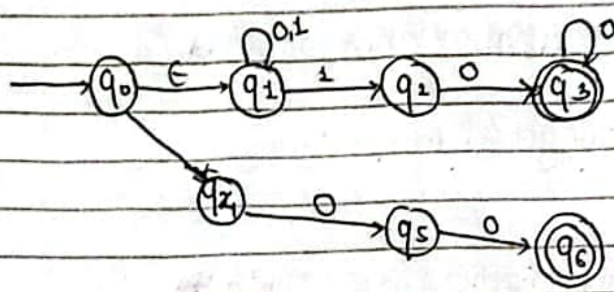
(ii) $\cancel{00+(0+1)^*100^*}$



(iii) $01(00+11)^*$



(ii) $00+(0+1)^*100^*$



2b) Here,

$A = \epsilon$ —— (i)

$B = A.a + D.a$ —— (ii)

$C = B.b + D.b$ —— (iii)

$D = A.b + B.a + C.a + C.b$ —— (iv)

Putting values of B & C in (iv),

$D = A.b + Aaa + Daa + B.ba + Dba + Bbb + D.bb$

or, $D = b + aa + B.ba + Bbb + D.aa + Dba + Dbb$ [using (i)]

or, $D = b + aa + Aaba + Daba + Aabb + Dabb + Daa + Dba + Dbb$

[using (ii)]

or, $\underset{R}{D} = \underbrace{(b+aa+aba+abb)}_{Q} + \underset{R}{D}\,\underbrace{(aba+abb+aa+ba+bb)}_{P}$

Using Ardens theorem, comparing with $R = Q + RP$,

$R = D$

$Q = (b+aa+aba+aba)$

$P = (aba+abb+aa+ba+bb)$

We get, $R = QP^*$

$\therefore D = (b+aa+aba+abb)(aba+abb+aa+ba+bb)^*$ —— (v)

For (ii), Using (i) & (v) in (ii), we get,

$$B = a + (b+aa+aba+abb)(aba+abb+aa+ba+bb)^*a \quad \text{— (vi)}$$

Hence, From (v) & (vi), we get (iii) as,

$$C = B.b + D.b$$

$$RE = (a+(b+aa+aba+abb)(aba+abb+aa+ba+bb^*a)b$$
$$(a+(b+aa+aba+abb)(aba+abb+aa+ba+bb)^*ab + (b+aa+aba+abb)(aba+abb+aa+ba+bb)$$

Ans

2c) soln

Closure properties of regular language:

- Regular language exhibit structural properties known as closure properties, meaning that applying specific operations to regular language produces another regular language.
- The class of language occupied by finite automata is closed under:

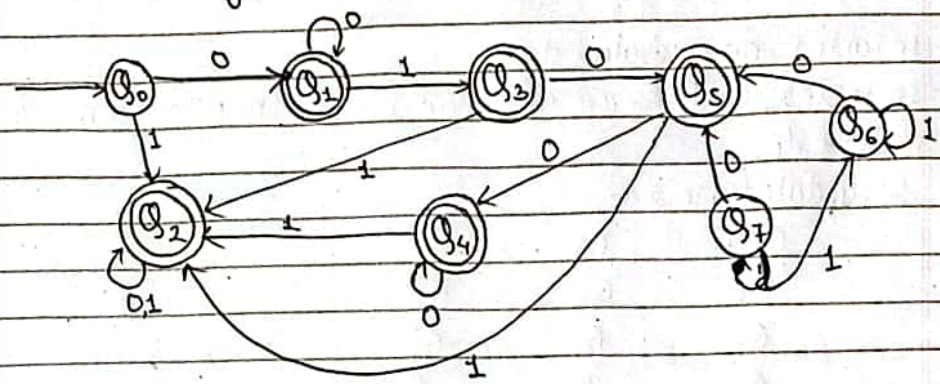
(i) Union
The union of two regular languages L_1 and L_2 denoted by $L_1 \cup L_2$ is the set of strings belonging to L_1, L_2 or both.

(ii) Concatenation
The concatenation of L_1 and L_2 denoted by $L_1.L_2$ is the set of strings formed by appending any string from L_2 to any string from L_1.

(iii) Kleene Closure
The Kleene closure of a language L, denoted L^*, is the set of all strings formed by concatenating zero or more strings from L (including the empty string ε).

Q3a) soln

The initial diagram is as,



State diagram

Here, q_7 and q_6 are unreachable states. So, remove them, constructing transition table of remaining states,

Q/Σ	0	1
q_0	q_1	q_2
q_1	q_1	q_3
q_2	q_2	q_2
q_3	q_5	q_2
q_4	q_4	q_2
q_5	q_4	q_2

Dividing above table into two table as follows:

Q/Σ	0	1
q_0	q_1	q_2

Table-1

Q/Σ	0	1
q_1	q_1	q_3
q_2	q_2	q_2
q_3	q_5	q_2
q_4	q_4	q_2
q_5	q_4	q_2

Table-2

- In Table 1, no equivalent states.
- In Table 2, q_4 & q_5 are equivalent because they have same o/p to same i/p.

So, update table 2 as,

Q/Σ	0	1
q_1	q_1	q_3
q_2	q_2	q_2
q_3	q_4	q_2
q_4	q_4	q_2

Table-3

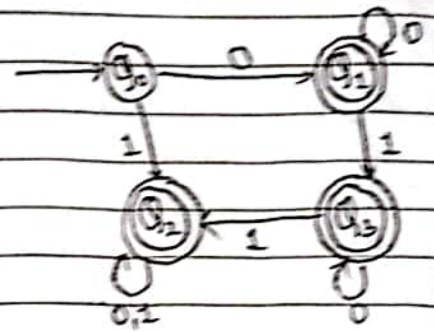
- In Table-3, q_3 & q_4 are equivalent. So, updated table is,

Q/Σ	0	1
q_1	q_1	q_3
q_2	q_2	q_2
q_3	q_3	q_2

Hence, Final table is (after combining):

Q/Σ	0	1
$\rightarrow q_0$	q_1	q_2
$*q_1$	q_1	q_3
$*q_2$	q_2	q_2
$*q_3$	q_3	q_2

Minimized diagram is,



Q.3b)

Finite Automata is the model of computation that accepts/rejects language.

Given language is $L = \{w \in \{a,b\}^* : w$ contain 'bb' or 'bab' as substring$\}$

Let DFA be $D = (Q, \Sigma, \delta, q_0, F)$

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

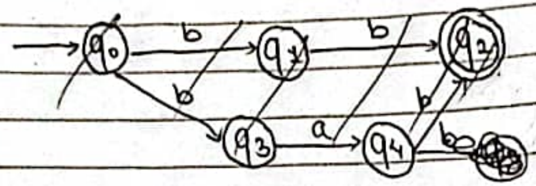
$\Sigma = \{a, b\}$

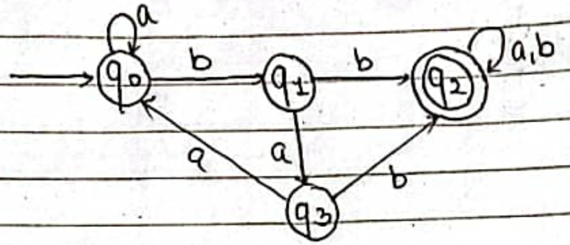
$q_0 = q_0$

$F = \{q_2\}$

δ:

Q|Σ	a	b
→q_0	q_0	q_1
q_1	q_3	q_2
*q_2	q_2	q_2
q_3	q_1	q_2





Test:

(i) abaabb

$\delta(q_0, abaabb)$
$\vdash \delta(q_0, baabb)$
$\vdash \delta(q_1, aabb)$
$\vdash \delta(q_3, abb)$
$\vdash \delta(q_0, bb)$
$\vdash \delta(q_1, b)$
$\vdash \delta(q_2, \epsilon)$
Accept

(ii) baaba

$\delta(q_0, baaba)$
$\vdash \delta(q_1, aaba)$
$\vdash \delta(q_3, aba)$
$\vdash \delta(q_0, ba)$
$\vdash \delta(q_1, a)$
$\vdash \delta(q_3, \epsilon)$
Reject

Q40) Is NFA more powerful than DFA?

soln

NFAs and DFAs are equally powerful in terms of computational capability as both recognize exactly the same class of languages (regular languages). This equivalence is proven by the fact that every NFA can be converted into an equivalent DFA, though the DFA may require exponentially more states.

1. Formal Equivalence Proof
Any NFA $M = (Q, \Sigma, \delta, q_0, F)$ can be converted to DFA $M' = (Q', \Sigma', \delta', q_0, F')$ where,
$Q'_D = 2^{Q_N}$ (all subsets of NFA states)

2. NFA Features Do not add power
While NFAs allow:
- ε-transitions
- Multiple transitions for a single input,

These features donot expand the set of recognizable languages beyond what DFAs can handle.

Examples illustrating equivalence:

~~Example 1~~: NFA:
Consider an NFA accepting strings ending in "01":
$Q = \{q_0, q_1, q_2\}$
$\Sigma = \{0, 1\}$
$q_0 = q_0$
$F = \{q_2\}$

δ:

Q/Σ	0	1
q_0	$\{q_0, q_1\}$	$\{q_0\}$
q_1	ϕ	$\{q_2\}$
q_2	ϕ	ϕ

Equivalent DFA:

We obtain the DFA for above NFA as, $D = (Q', \Sigma, \delta', q_0', F')$

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

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

$q_0' = q_0$

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

δ':

Q'/Σ	0	1
$\{q_0\}$	$\{q_0, q_1\}$	$\{q_0\}$
$\{q_0, q_1\}$	$\{q_0, q_1\}$	$\{q_0, q_2\}$
$\{q_0, q_2\}$	$\{q_0, q_1\}$	$\{q_0\}$

This DFA accepts the same language as original NFA.

Q4b)

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

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

$T = \{a, b\}$

$S = \{S\}$

$P = \{ S \to AB$

$A \to aA$

$A \to a$

$B \to bB$

$B \to b \}$

Using Left Most Derivation,

$S \Rightarrow AB$

$\Rightarrow aAB$

$\Rightarrow aaAB$

$\Rightarrow aaaAB$

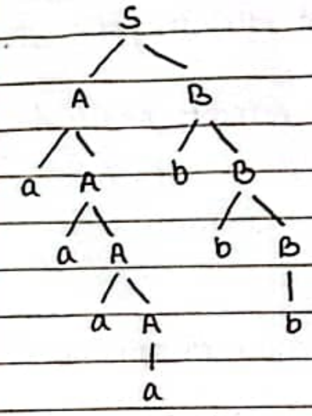
$\Rightarrow aaaaB$

$\Rightarrow aaaabB$

$\Rightarrow aaaabbB$

$\Rightarrow aaaabbb$

Parse tree:



Using Right Most Derivation,

$S \Rightarrow AB$

$\Rightarrow AbB$

$\Rightarrow AbbB$

$\Rightarrow Abbb$

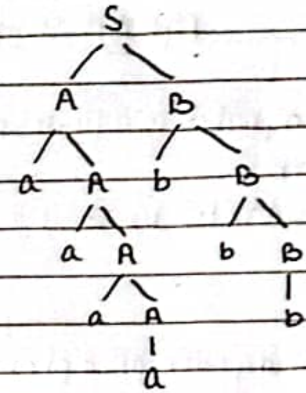
$\Rightarrow aAbbb$

$\Rightarrow aaAbbb$

$\Rightarrow aaaAbbb$

$\Rightarrow aaaabbb$

Parse tree:



Q5b)

Pumping Lemma for regular language

- Pumping Lemma says that all sufficient long words, in a regular language may be pumped i.e. have a middle section of word repeated an arbitary no. of times to produce new word also lies within same language.

Statement of pumping lemma:

- Let L be a regular language.
- w be any string, $w \in L$ with $|w| \geq n$; where n be integer constant
- There are strings x, y, z such that

$$w = xyz \quad \text{where } |xy| \leq n$$
$$|y| > 0$$

then, $xy^iz \in L$ for all $i \geq 0$

Here, substring y is pumped.

5a) Alphabet, String, Kleene Closure, Positive Closure & Language

- An alphabet is a finite, non-empty set whose elements are called symbols.
- It is denoted by Σ. e.g. $\Sigma = \{0,1\}$

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

- A string over an alphabet Σ is a finite sequence of symbols where each symbol is an element of Σ.
- It is denoted by w.

- The set of all strings over an alphabet Σ is called Kleene closure of Σ.

- The set of all strings over an alphabet Σ except empty string is called positive closure of Σ.

- Any set of strings over an alphabet Σ given some conditions is called language.