1

Aayush Garg
2K17/MC/003

Assignment No. 1

Theory of Computation

Q1). A finite automata is a 5-tuple

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

→ Q is a finite set of states
→ Σ is an input alphabet
→ $\delta : Q \times \Sigma \to Q$, the transistion function
$q_0 \in Q$ is the initial state
$F \subseteq Q$ is the finite set of final state

M accepted as string w if M will, by starting in states q_0 and reading the characteristics of w, end up in a ~~full~~ final state. So a final finite automate is a language sacrifice

A transistion system is a tuple (T) where configuration
T is a set of ~~confisuation~~

T ∈ T is a set of terminal

A finite automata can be seen as a labelled transistion system whose configuration as its states, whose label set id the input alphabet, whose terminal corresponds to the transistion function.

Transistion system that fails to be finite automata. The set of configurational may be infinite, as may the set of labels and transistion relation may cease to be deterministic.

②

②

DFA

1). It is a 5-tuple $(q, \Sigma, \delta, q_0, F)$ where δ is the transition function mapping from $q \times \Sigma$ to q.

2). It stands for deterministic finite automata which means on a single it can only go to a single output / have a single next state.

3). It cannot use empty string transitions

4). DFA can be visualized as one machine.

5). DFA is a complete system.

③.

NDFA

1). It is a 5-tuple $(q, \Sigma, \delta, q_0, F)$ where δ is the transition function mapping from $q \times \Sigma$ into 2^q which is the power set of q the set of all subsets of q.

2). It stands for non-deterministic finite automata which means on any input it can go to multiple next states.

3). It can use empty string transitions.

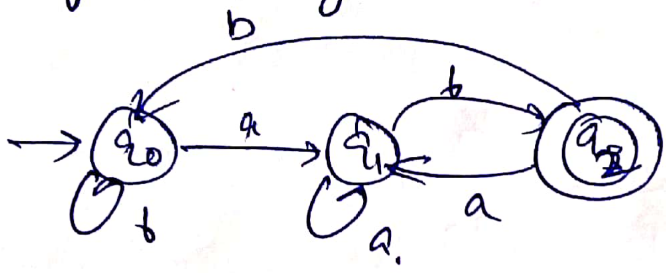
4) NDFA can be understood as multiple machines computing at same time.

5). NDFA need not be completed.

Q3

DFA accepting language {a, b} that have set of all strings that end with ab.

Ans.



State	a	b
$\rightarrow q_0$	q_1	q_0
q_1	q_1	q_2
q_2	q_1	q_0

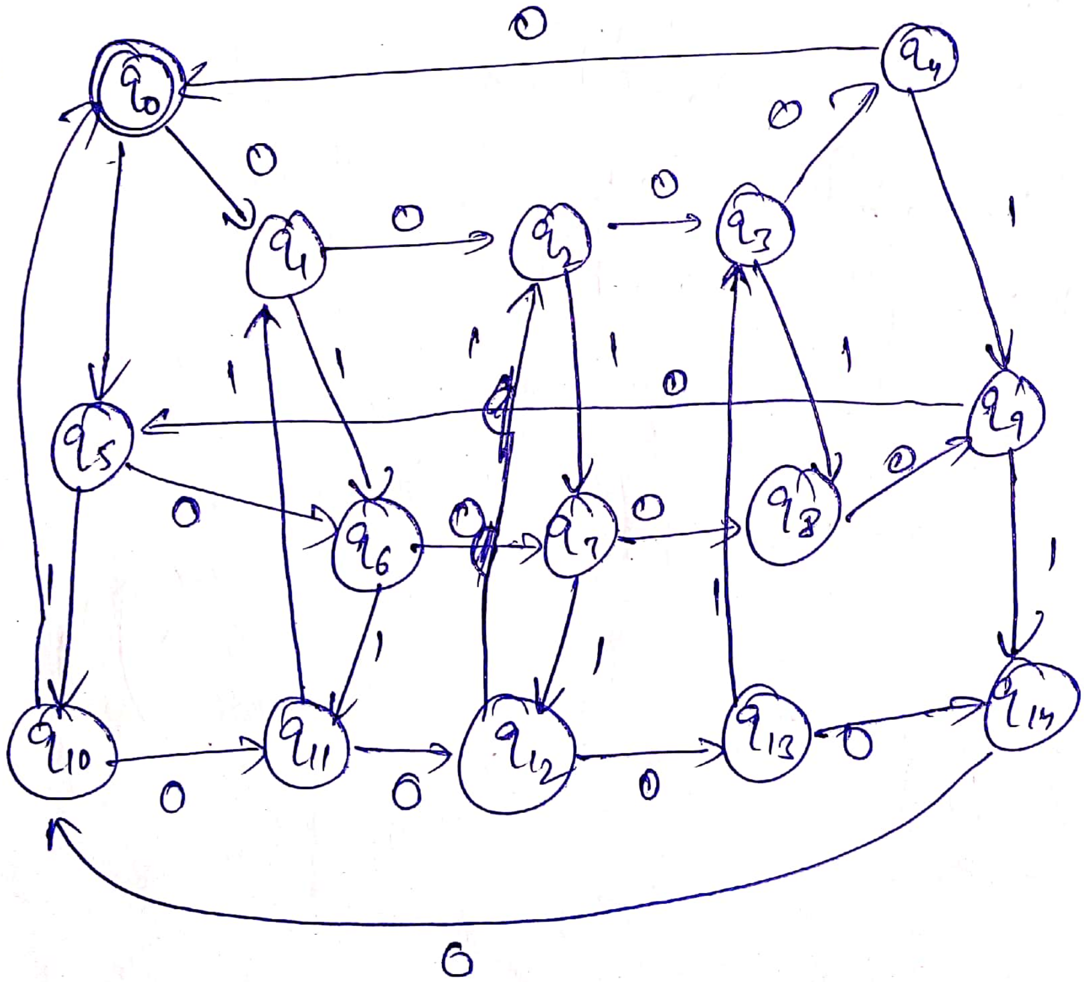
No. of states = 3

Initial state = q_0

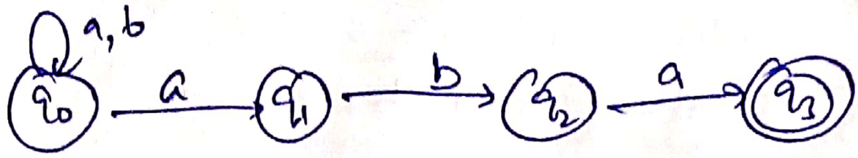
Final state = q_2

L = {ab, aab, bab, aaab, bbab, ...}

4).



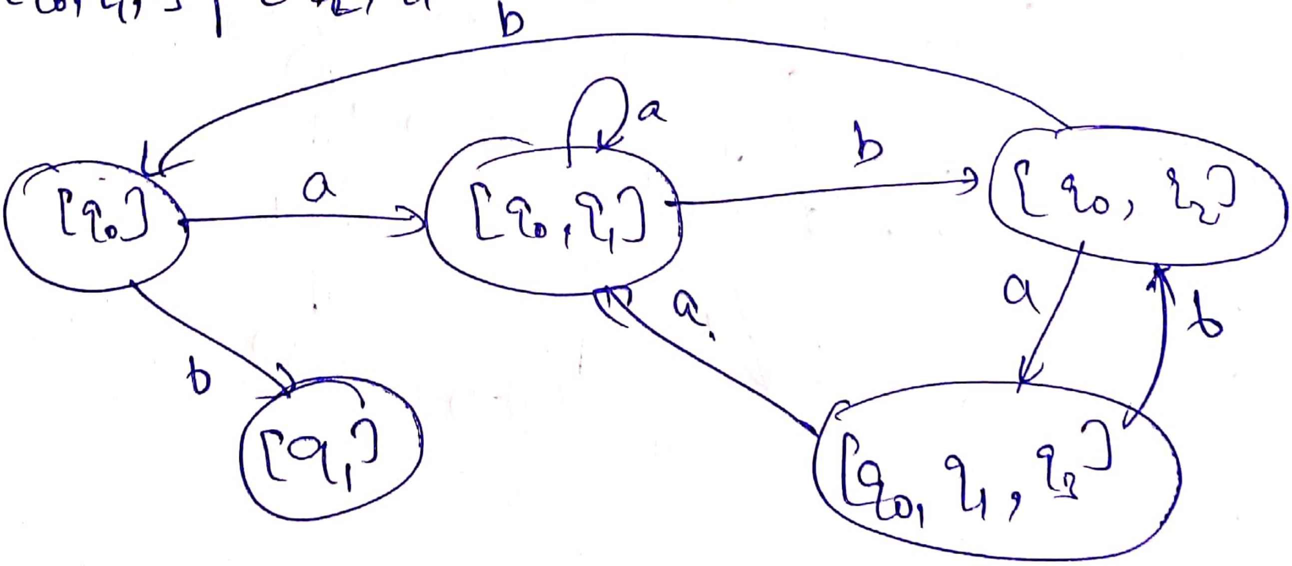
5).



State/Σ	a	b
q_0	q_0, q_1	q_0
q_1	–	q_2
q_2	q_3	–
q_3	–	–

Converting this to DFA.

State/Σ	a	b
$[q_0]$	$[q_0, q_1]$	$[q_1]$
$[q_0, q_1]$	$[q_0, q_1]$	$[q_0, q_2]$
$[q_0, q_2]$	$[q_0, q_1, q_3]$	$[q_0]$
$[q_0, q_1, q_3]$	$[q_0, q_1]$	$[q_1, q_2]$



6)

Present State	Next State a=0	Next State a=1	Output
→ q_0	q_1	q_2	1
q_1	q_3	q_2	0
q_2	q_2	q_1	1
q_3	q_0	q_3	1

Mealey

Present State	Next State a=0 State	a=0 Output	a=1 State	a=1 Output
→ q_0	q_1	0	q_2	1
q_1	q_3	1	q_2	1
q_2	q_2	1	q_1	0
q_3	q_0	1	q_3	1

7). Construct a Moore Machine, equivalent to mealy Machine.

Present State	Next State a=0 State	a=0 Output	a=1 State	a=1 Output
→ q_1	q_1	1	q_2	0
q_2	q_4	1	q_4	1
q_3	q_2	1	q_3	1
q_4	q_3	0	q_1	1

(8)

Present State	Next State			
	a = 0		a = 1	
	State	Output	State	Output
q_1	q_1	1	q_{20}	0
q_{20}	q_4	1	q_4	1
q_{21}	q_4	1	q_4	1
q_{30}	q_{21}	1	q_{31}	1
q_{31}	q_{21}	1	q_{21}	1
q_4	q_{30}	0	q_1	1

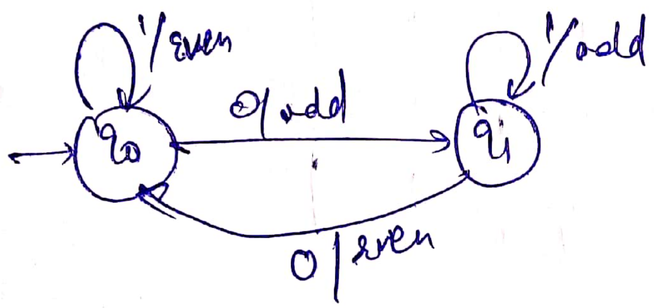
State Table.

Present State	Next State		Output
	a = 0	a = 1	
$\to q_1$	q_1	q_{20}	1
q_{20}	q_4	q_4	0
q_{21}	q_4	q_4	1
q_{30}	q_{21}	q_{31}	0
q_{31}	q_{21}	q_{31}	1
q_4	q_{30}	q_1	1

Revised State Table.

8). Input alphabet = {0, 1}

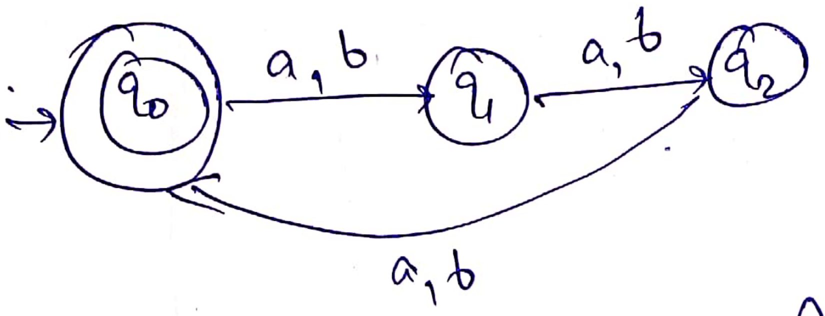
Output alphabet = {Even, odd}



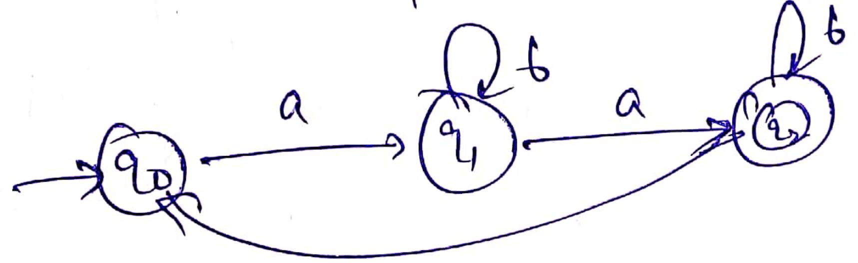
Transition table

Present state	Next state			
	a = 0		a = 1	
	State	Output	State	Output
q_0	q_1	odd	q_0	even
q_1	q_0	even	q_1	odd

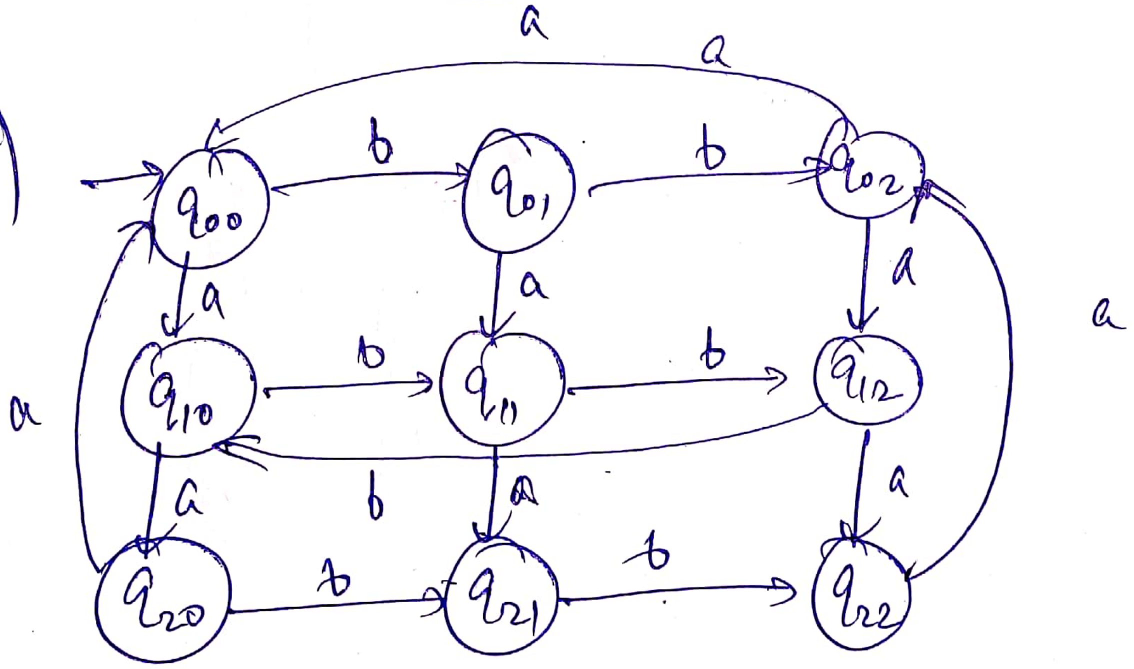
9) a)



b).



c)



d).