Aayush Garg Assignment No 2 (1)
2K17/MC/003 Theory of Computation

1). Chomsky classified the grammars into four types in terms of productions (0-3).

To define the grammars, we define

In a production of the form $\phi A \psi \rightarrow \phi \alpha \psi$, where A is a variable, ϕ is called the left context, ψ is the right context and $\phi \alpha \psi$ is the replacement string

Type 0

Let G be be Type 0 grammar. Then we can find an equivalent grammar G_1 in which each production is either of the form $\alpha \rightarrow \beta$, where α and β are strings of variables only, or of the form $A \rightarrow a$ where A is variable and a is terminal

Type 1

A grammar is called type 1 or context sensitive if all its productions are type 1 productions. The production $S \rightarrow \Lambda$ is also allowed in type 1 grammar, but in this case S does not appear in the right hand side of any production.

Eg $A \rightarrow abA$ is a type 1 production
Both left and right contexts are Λ

Type 2

A grammar is called a type 2 grammar if it contains only type 2 productions. It is also called a context free grammar.

as A can be replaced by α in any context.

A language created by a context free grammar is called type 2 language or context free language.

A type 2 production is a production of the form $A \rightarrow \alpha$ where $A \in V_N$ and $\alpha \in (V_N \cup \Sigma)^*$

In other words L.H.S has no left context or right context.

Eg $S \rightarrow Aa$, $A \rightarrow a$, $B \rightarrow abc$

Type 3

A production of the form $A \rightarrow a$ or $A \rightarrow aB$ where $A, B \in V_N$ and $a \in \Sigma$ is called a type 3 production.

A grammar is called a type 3 or regular grammar if all its productions are type 3 productions.

Eg $S \rightarrow \Lambda$ production, $S \rightarrow aS$

In this case S does not appear in R.H.S of any production

$w_1 = \{ 0S1, 0A, 0, 1B, 1 \}$

$w_2 = \{ 0S1, 00, 00A, 0, 11B, 11, 1 \}$

as

2) To test whether

$001100 \in L(G)$

$w = 001100 \Rightarrow |w| = 6$

$w_0 = \{S\}$

$w_1 = \{ 0S1, 0A, 0, S, 1B, 1 \}$

$w_? = \{ 0, 1, 0S1, S, 0A, 1B \}$

$w_2 = \{ 00S1, 00A1, 001, 01B1, 011, 00A, 00, 11B, 11, 1 \}$

$w_3 = \{ 000S11, 000A1, 001B1, 0011, 001, 000A1, 011B, 011, 000A, 00, 111B, 111, 11, 1 \}$

$w_4 = \{ 000011, 000111, 00001, 001111, 00111, 0011, 001, 011111, 01111, 011, 00000A, 00, 11111B, 00000, 11111, 1111, 111, 11, 1 \}$

$w_5 = \{000011, 000111, 000001, 001111, 00111, 0011,$ (4)
$001, 011111, 01111, 011, 00000A, 00,$
$11111B, 00000, 11111, 1111, 111, 11, 1\}$

$w_6 = \{000011, 000111, 000001, 001111, 00111, 0011,$
$001, 011111, 01111, 011, 000000, 00, 111111,$
$00000, 11111, 1111, 111, 11, 1\}$

$w_7 = \{w_6\}$

$\because$ $001100 \notin w_6$ & $001010 \notin w_6$

also $01010 \notin w_6$

$\Rightarrow$ $001100, 001010$ & 01010
are not generated by the given grammar

3) Type 2 grammar are used as context free grammar (as A can be replaced by α in any context from the product $A \to \alpha$)
where $A \in V_N$ and $\alpha \in (V_N \cup \Sigma)^*$.

Grammar rule for generating,

Scanned with CamScanner

One possible context free grammar containing twice as many zeros as ones can be, ⑤

$S \to 0S0S1S|\ 0S1|S0S|\ 1S0S0S.$

4). Let $M = (\{q_1, q_2, --- q_n\}, \Sigma, \delta, q_1, F)$

The construction set that we give can be better understood in terms of the state diagrams of M. If a string $w \in \Sigma^*$ is accepted by M then ∃ a path from q_1 to some final state of M with path value w. So to each final state say q_j, there corresponds a subset of Σ^* consisting of path values from q_0 to q_j..

As T(M) is the union of such subsets of Σ^* it is enough to represent them by regular expressions. So the main part of the proof lies in the construction of subsets of path values of path from the state q_i to the state q_j.

Let P_{ij}^k denote the set of path values of paths from q_i to q_j whose intermediate vertices lie in $\{q_1, --- q_k\}$. We construct P_{ij}^k for $k = 0, 1, --- n$ recursively as follows:

$$P_{ij}^0 = \{a \in \Sigma \mid \delta(q_i, a) = q_j\} \quad — ①$$

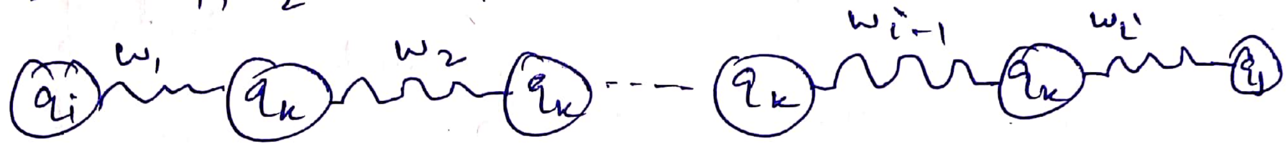
$$P_{ii}^0 = \{a \in \Sigma \mid \delta(q_i, a) = q_i\} \cup \{\Lambda\} \quad — ②$$

$$P_{ij}^k = P_{ik}^{k-1}(P_{kk}^{k-1})^* P_{kj}^{k-1} \cup P_{ij}^{k-1} \quad — ③$$

In terms of the state diagram, the construction can be understood better. P_{ij}^0 simply denotes the set of path values of edges from q_i to q_j. In P_{ii}^0 we include a Λ in addition to labels of self-loops from q_i. This explains (1) & (2)

Consider a path from q_i to q_j which has intermediate vertices be in $\{q_1, q_2, \ldots q_k\}$. If the path does not pass through q_k, then its path values lies in P_{ij}^{k-1}. Otherwise, the path passes through q_k possibly more than once.

The path can be split into several path with path values $w_1, w_2, \ldots w_i$.



$w = w_1 w_2 \ldots w_i$. w_1 is the path value of the path from q_i to q_k (i.e. q_k is not an intermediate vertex). $w_2, \ldots w_{i-1}$ are the path values of the paths from q_k to itself without passing through q_k. w_i is the path value of the path from q_k to q_j without passing through q_k. So, w_1 is in P_{ik}^{k-1}, $w_2 \ldots w_{i-1}$ are in (P_{kk}^{k-1}) & w_i is in P_{kj}^{k-1}.

This explains (3).

We prove that the sets introduced by ① - ③ are represented by regular expression by induction on k ($\forall$ i and j). P_{ij}^{0} is a finite subset of Σ, say $\{a_1, \ldots a_i\}$. Then P_{ij}^{0} is represented by $P_{ij}^{0} = a_1 + a_2 + \ldots a_i$ (union). $\|$ly, we can construct P_{ij}^{0} representing P_{ii}^{0}. Thus, there is a basis for induction.

Let us assume the result holds for (k-1), ie P_{ij}^{k-1} is represented by a r.e. P_{ij}^{k-1} $\forall$ i, j. From ③

$$P_{ij}^{k} = P_{ik}^{k-1} (P_{kk}^{k-1})^* P_{kj}^{k-1} \cup P_{ij}^{k-1}.$$

$\therefore$ the result is true for all k. By principle of induction the sets constructed by ① - ③ are represented by r.e.

As $Q = \{q_1, \ldots q_n\}$, P_{ij}^{n} denotes the set of path value of all paths from q_i to q_j. If $F = \{q_{f_1}, \ldots q_{f_n}\}$ then

$T(M) = \bigcup_{j=1}^{n} P_{1f_j}^{n}$. So, T(M) is represented by the r.e

$P_{1f_1}^{n} + \ldots + P_{1f_m}^{m}$ is represented by a r.e

Q) 5) a) $\{a^2, a^5, a^8 \ldots\} = aa(aaa)^*$ (8)

b). $\{w \in \{a,b\}^* \mid$ ~~where~~ has only one a$\} = b^* a b^*$

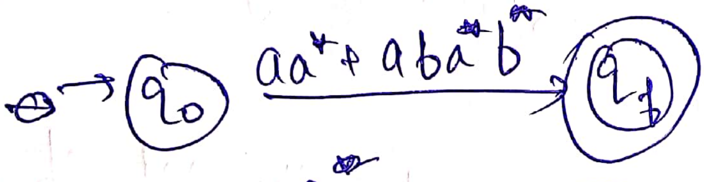
c) $\{a^n \mid n$ is divisible by 2 or 3 or n = 5$\}$

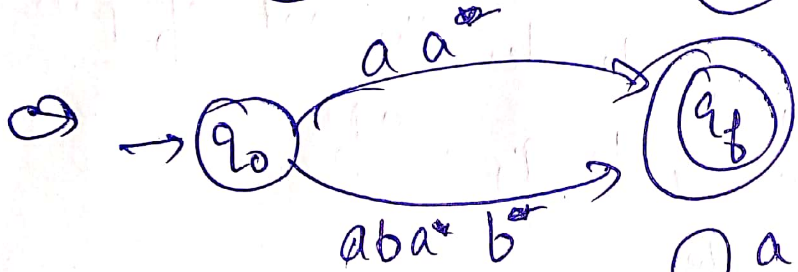
$= (aa)^* + (aaa)^*$

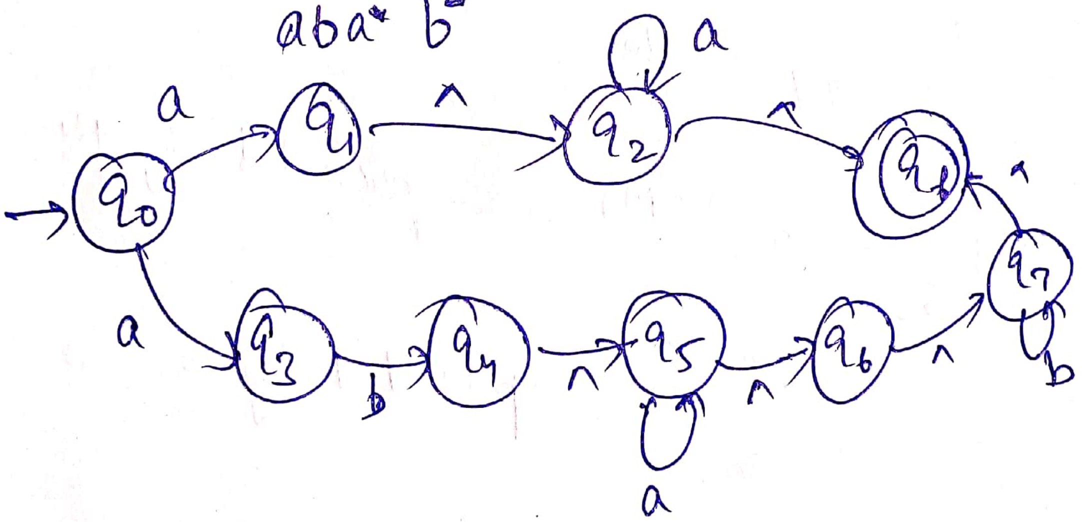
d). Set of all strings over $\{a, b\}$ beginning and ending with a = $a(a+b)^* a$

6) a). $aa^* + aba^* b^*$

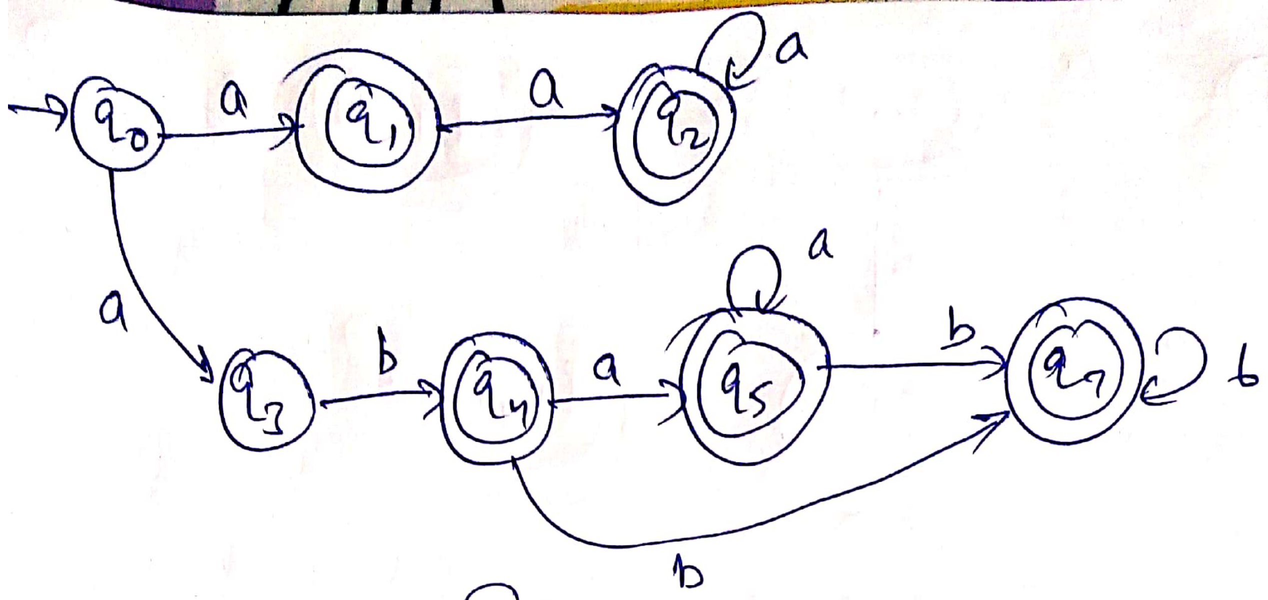
Step 1 (Constructions of NDFA)

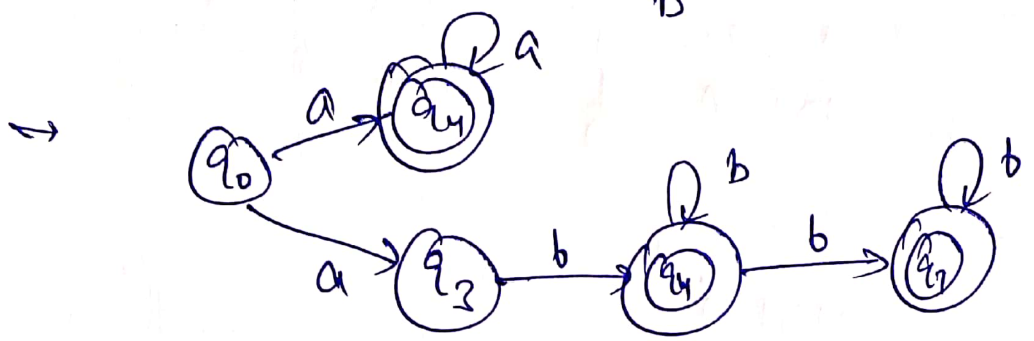






G





step 2

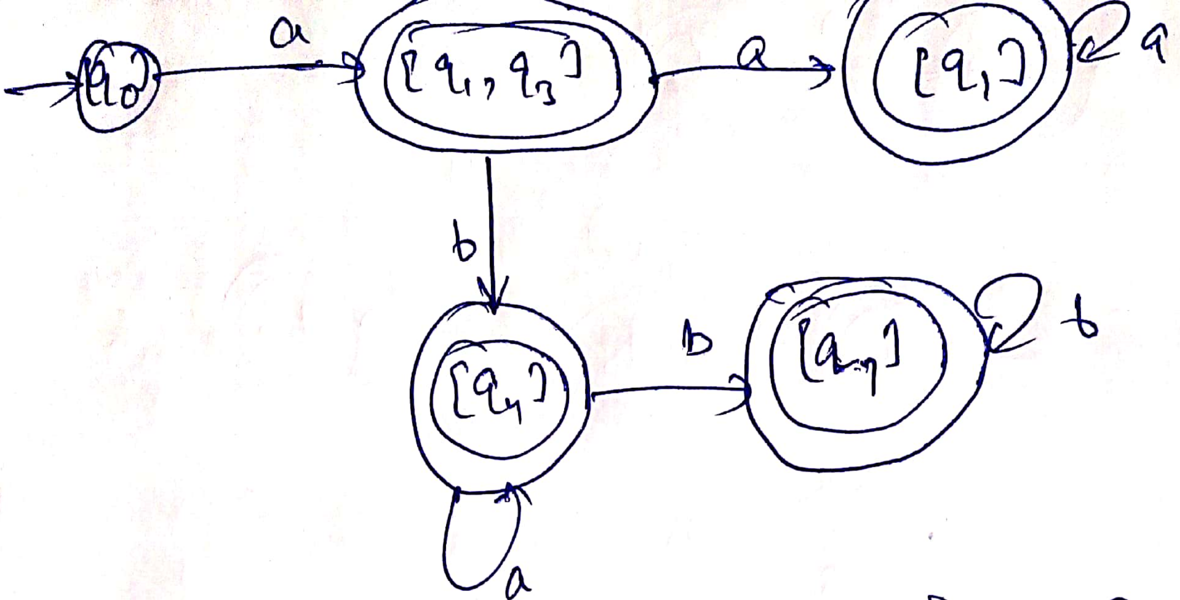
Construct the DFA

Corresponding to above NDFA, we can create transition table, as :-

State	a	b
→ q_0	q_1, q_3	-
q_1	q_1	-
q_3	-	q_4
q_4	q_4	q_7
q_7	-	q_7

The successor table can also be constructed as :-

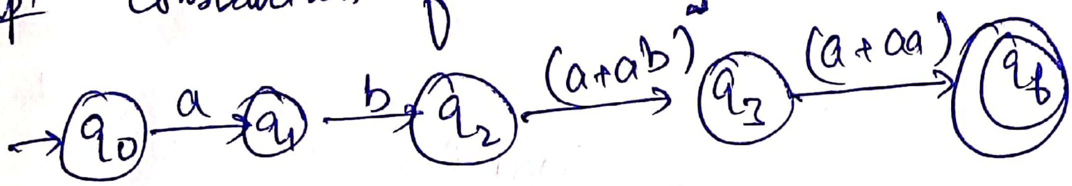
Q	Q_a	Q_b
$[q_0]$	$[q_1, q_3]$	ϕ
$[q_1, q_3]$	$[q_1]$	$[q_4]$
$[q_1]$	$[q_1]$	ϕ
$[q_4]$	$[q_4]$	$[q_7]$
$[q_7]$	ϕ	$[q_7]$

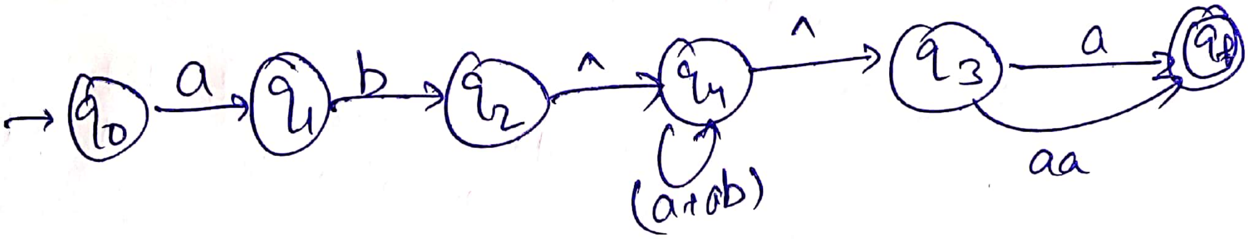


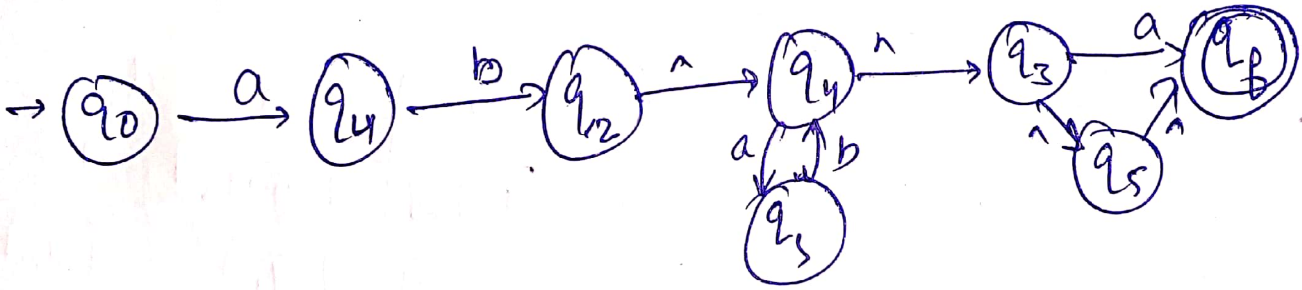
The above is DFA of given r.e.

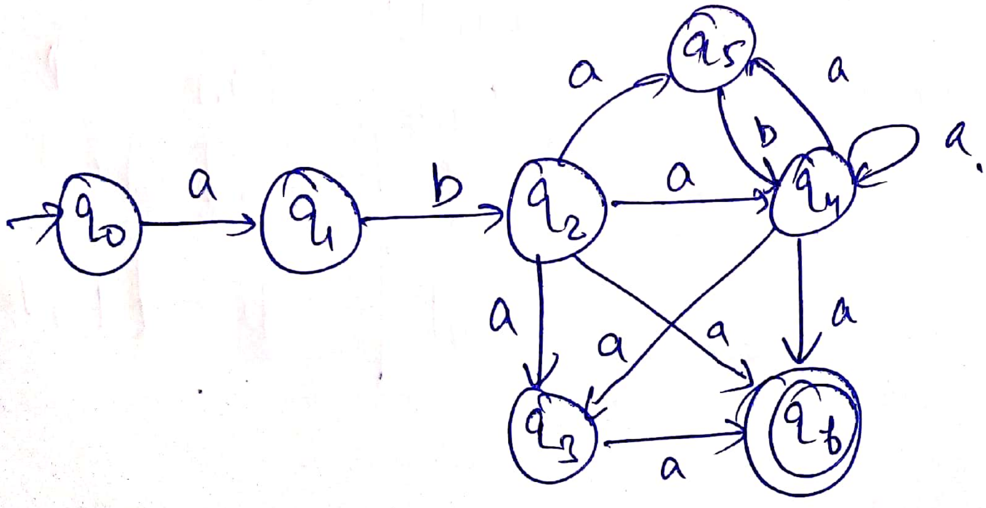
b). $ab(a+ab)^*(a+aa)$

step1 Construction of NDFA









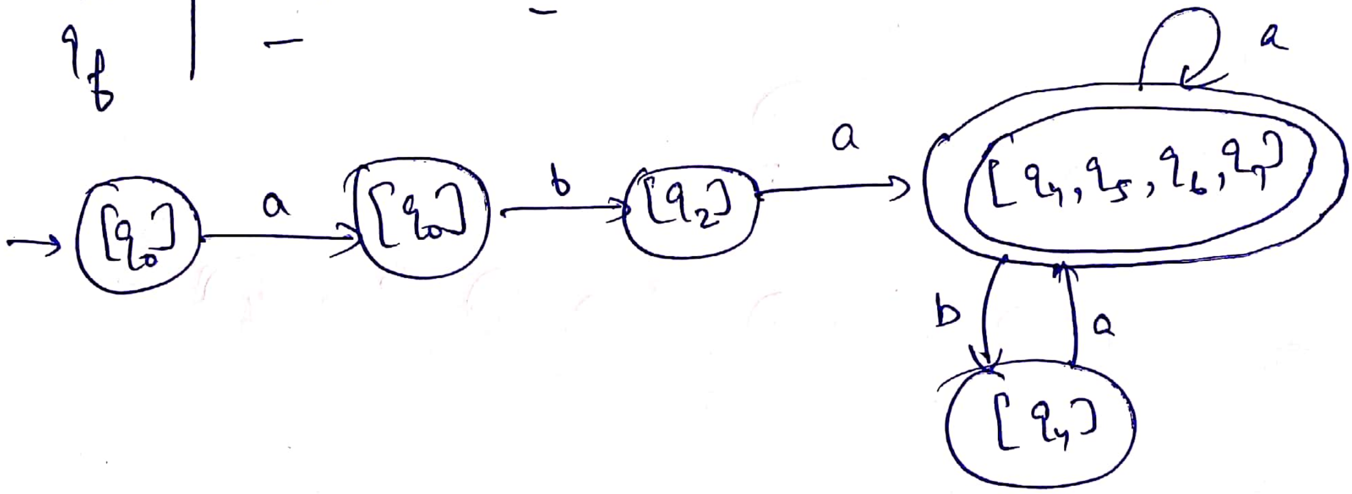
Step 2

Construct DFA

States	a	b
q_0	q_1	–
q_1	–	q_2
q_2	q_4, q_5, q_6, q_7	–
q_4	q_4, q_5, q_6, q_7	–
q_5	–	q_4
q_6	q_f	–
q_f	–	–

Successor table

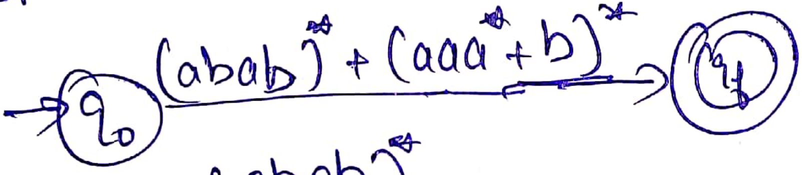
Q	Q_a	Q_b
→ $[q_0]$	$[q_1]$	ϕ
$[q_1]$	ϕ	$[q_2]$
$[q_2]$	$[q_4, q_5, q_6, q_7]$	ϕ
$[q_4, q_5, q_6, q_7]$	$[q_4, q_5, q_6, q_7]$	$[q_4]$
$[q_4]$	$[q_4, q_5, q_6, q_7]$	ϕ

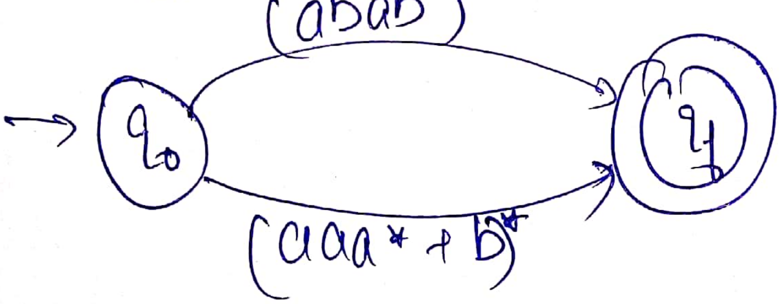


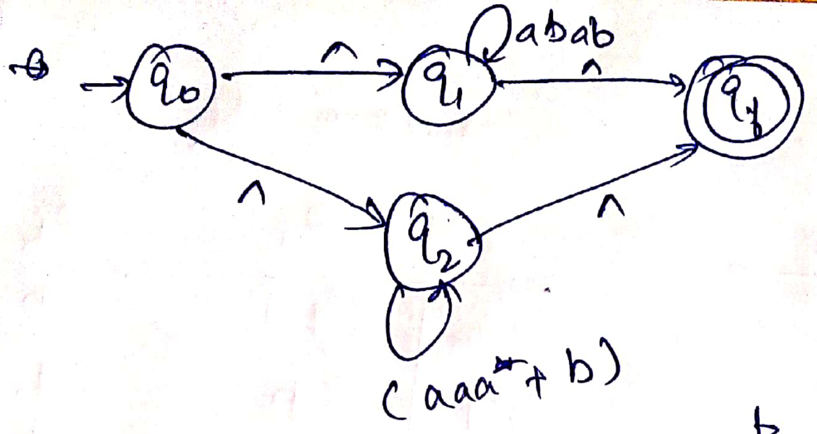
Q6

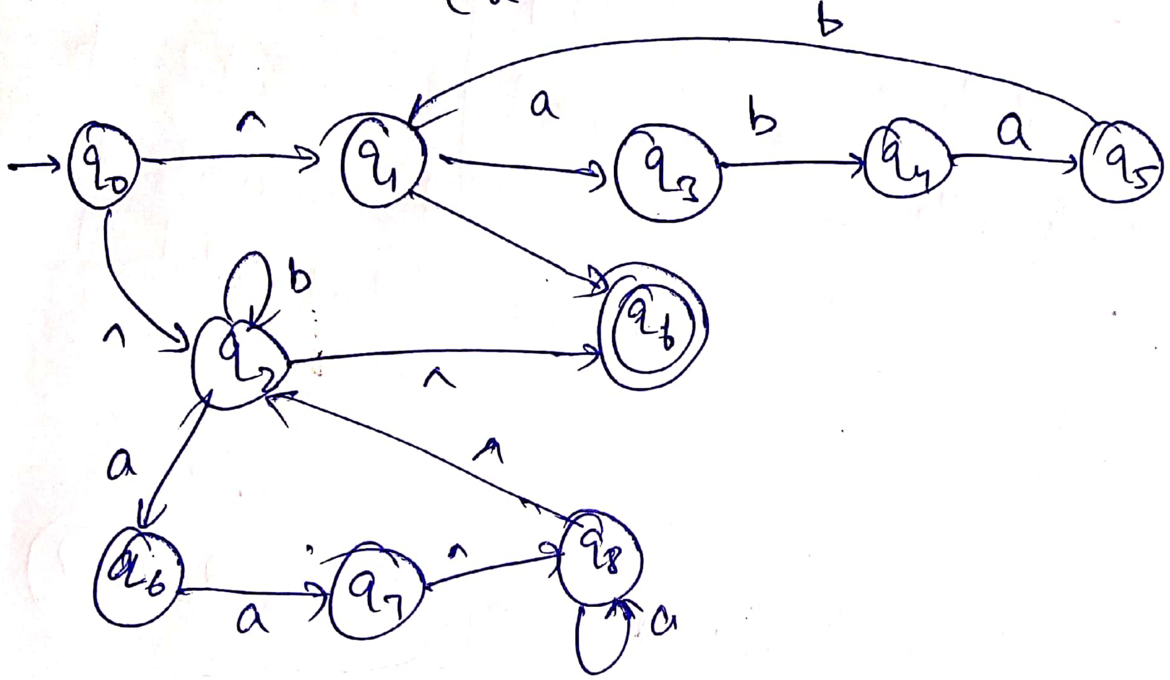
c). $(abab)^* + (aaa^* + b)^*$

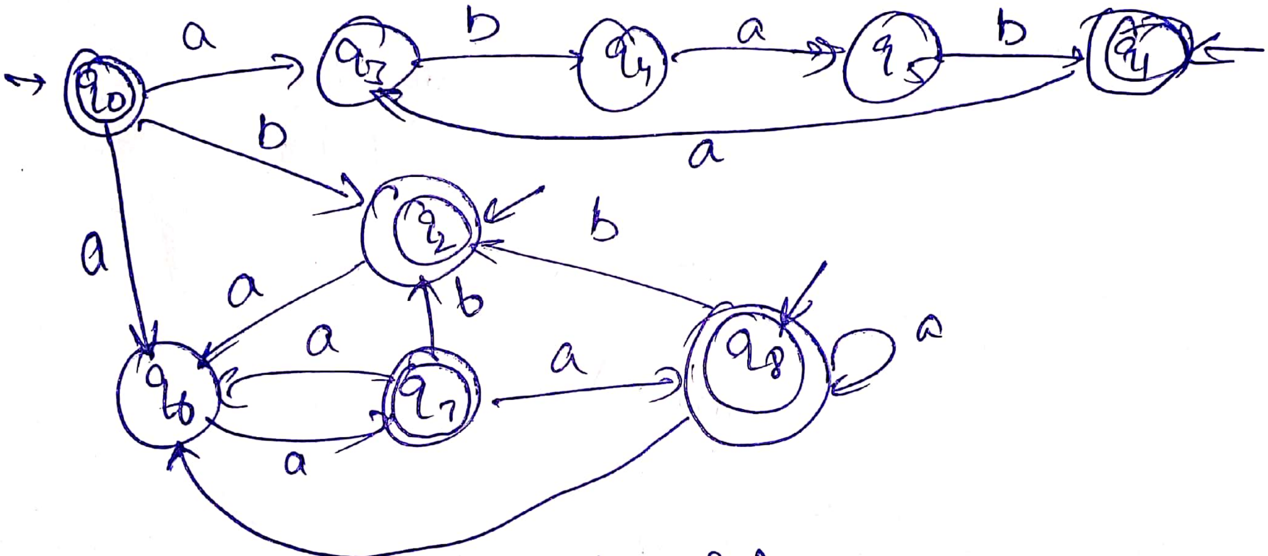
Step 1 (Construction of NDFA)









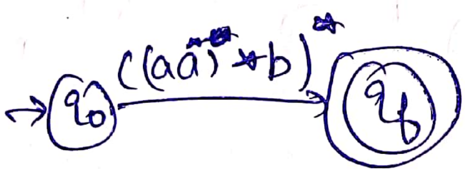


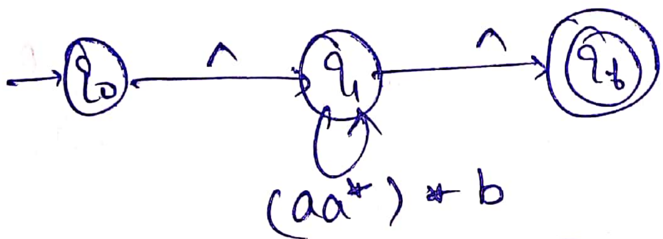
Step 2 Construction of DFA

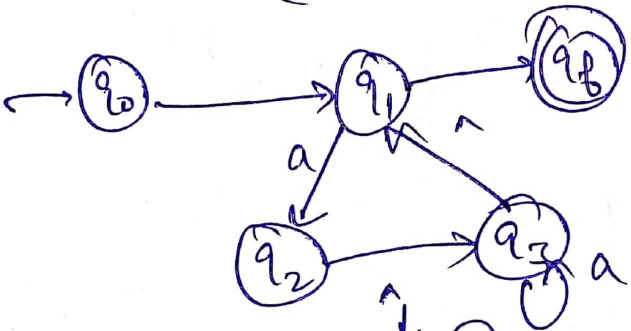
The transition table is given as

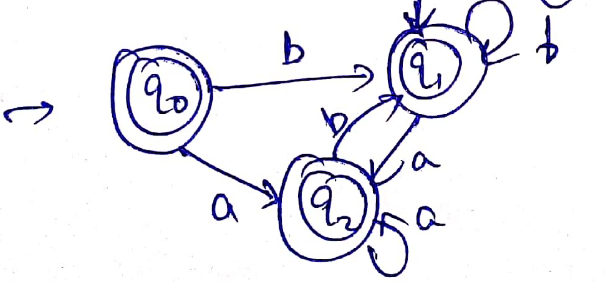
States	a	b
→ q_0	$q_3 q_6$	q_2
→ q_1	q_3	–
→ q_2	q_6	q_2
q_3	–	q_4
q_4	q_5	–
q_5	–	q_1
q_6	q_7	q_1
→ q_7	q_6, q_6	q_2
→ q_8	q_6, q_8	q_2

Q6) d). $((aa^*)^*b)^*$









DFA

Q7) a) $\{a^n : n \geq 0, n \neq 4\}$

For this set, we can write a regular expression as $\Lambda + a + aa + aaa + aaaaaa^*$

Hence it is a regular set.

b). $\{a^n : n$ is either a multiple of 3 or a multiple of 5$\}$

For this set, we can write a regular expression as $aaa(aaa)^* + aaaaa(aaaaa)^*$

c). $\{a^n : n$ is multiple of 3$\}$

For this set we can construct a DFA such as