THEORY OF COMPUTATION

★ Sets → Collection of Objects.

- Properties
  - Non-ordered.
  - Elements are distinct
- Operations
  - Union - ($\cup$)
    $A \cup B = [x : x \in A \text{ or } x \in B]$
  - Intersection - ($\cap$)
    $A \cap B = [x : x \in A \text{ and } x \in B]$
  - Difference - (-)
    $A - B = [x : x \in A \text{ and } x \notin B]$
  - Unary - ($\bar{A}$)
    $\bar{A} = [x : x \notin A]$
- Properties w.r.t Operations.
  - Idempotency
    - $A \cup A = A$
    - $A \cap A = A$
  - Commutativity
    - $A \cup B = B \cup A$
    - $A \cap B = B \cap A$
  - Associativity
    - $(A \cup B) \cup C = A \cup (B \cup C)$
  - Distributivity.
    - $A \cup (B \cap C) = (A \cup B) \cap (A \cup C)$.
    - $A \cap (B \cup C) = (A \cap B) \cup (A \cap C)$.
  - Absorbtion.
    - $(A \cup B) \cap A = A$
    - $(A \cap B) \cup A = A$
  - De-Morgan's Law
    - $A - (B \cup C) = (A - B) \cap (A - C)$
    - $A - (B \cap C) = (A - B) \cup (A - C)$
    - $(A \cup B)' = A' \cap B'$
    - $(A \cap B)' = A' \cup B'$

Types

- Singleton (1 element)
- Empty (ϕ)
- Disjoint sets
  - 2 sets are disjoint in ($\cap$) is ϕ
- Subsets - $\subseteq$.
  - [$A \cup B = A$ if $A \subset B$]
- Proper subset
  - When $A \neq \phi$ & $A = B$ and $A \subset B$ then $A \subsetneq B$

- **Partition - Π**
  - Π is a partion of A if Π is a set of subset of A such that.

- Cartesian Product
  - $A \times B$
  - $[(a, b); a \in A \text{ \& } b \in B]$

- Power Set
  - set of all possible subsets of A, including ϕ & A.
  - cardinality $= 2^n$

★ Sequences.

- list of objects in some order.
- can be ∞ or finite.
- Order matters.
- Can have duplicates

★ Tuple

↳ A finite sequence.

- A sequence with k elements is a k-tuple.
- 2-tuples are called ordered pair.

★ Function

- Sets up an input & output relation. Also called a mapping.

func F maps D (domain) ⟶ R (range).

D → Set of possible inputs. R → Set of possible outputs.

- Predicate / Property.
  ↳ Range is [True, False]

★ Relation.

- It is a property / predicate whose domain is a set of k-tuples

★ Alphabet
↳ non empty finite set of symbols.

★ String
↳ finite list of symbols from alphabet

★ Language
↳ set of strings.

★ Types of Proofs

- By construction
  ↳ constructing an example to build a logic & prove true
- By contradiction
  ↳ By taking statement as false & proving this as false
- Mathematical Induction.

[Proof By Contradiction.
↳ $\sqrt{2}$ is rational.

Let $\sqrt{2}$ be rational.

$\therefore \sqrt{2} = \frac{m}{n}$ (m & n are in most reduced forms)

$2 = \frac{m^2}{n^2} \longrightarrow m^2 = 2n^2$

$\therefore m^2$ is an even number.

$\therefore m$ is even.

Let $m = 2k$

$n^2 = 2k^2$

$\therefore n^2 = 2k^2$

$\therefore n$ is also divisible by 2

m & n are both divisible by 2 but m & n are ~~it is not~~ in reduced form

~~but~~ $\therefore \sqrt{2}$ is not rational.

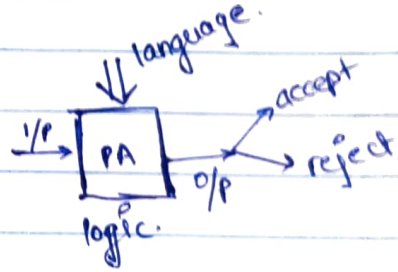
$\therefore \sqrt{2}$ is irrational.

★ History

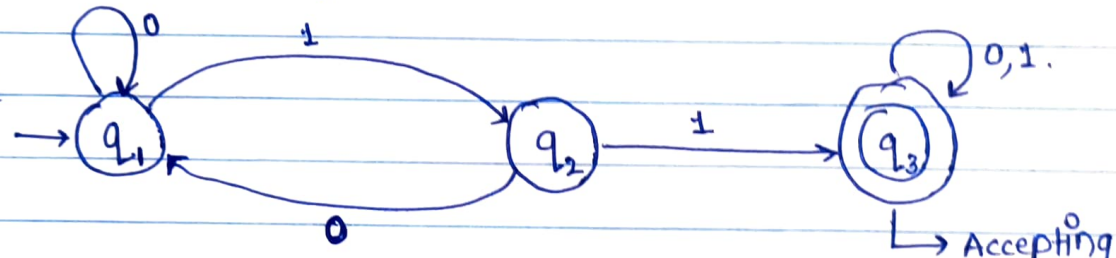
- 1930s → pre-computed era (Turing studies an abstract machine).
- 1940s & 1950s → machine called 'finite automata'.
- 1950s → Chomsky started studying 'grammers'.

★ Finite Automata (Intro)

- Finite Automata is a 5-tuple $(Q, \Sigma, \delta, q_0, F)$ where:
  - → Q → is a finite set of states.
  - → Σ → finite set called alphabets.
  - → $\delta \rightarrow Q \times \Sigma \rightarrow Q$ is transition func^n.
  - → $q_0 \rightarrow q_0 \in Q$ is the start state.
  - → $F \rightarrow$ set of accept states.



- M_1



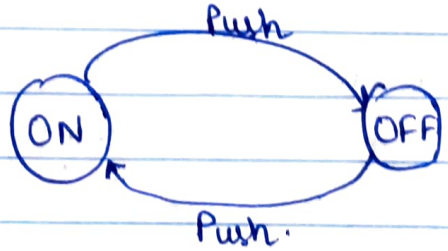
δ	0	1
q_1	q_1	q_2
q_2	q_3	q_3
q_3	q_3	q_3

$A = \{w \mid w$ contains string 11$\}$

$A = L\{M_1\}$.

↑ A is the lang of M_1.

- On / OF Switch.



★ Strings & Languages.

- A string is a finite set of symbols in Σ
- A language is a set of strings.
- An empty string ε is of length 0.
- An empty language is a set with no strings.

- Regular languages
  - A lang. is regular if a finite automat recognizes it.
  - M accepts string $w = w_1, w_2$

eg →

eg → Find odd one out.

- A = {w | w contains '11' string}
- B = {w | w has an even number of '1's}.
- C = {w | w has equal no. of 0s & 1s}.
- D = {w | w has odd no. of 0s & equal no. of 1s}.

★ Regular Operations.

- Regular operations on regular languages are called as regular operations.
- Union: $A \cup B = \{x \mid x \in A \text{ or } x \in B\}$.
- Concatenation: $A \circ B = \{xy \mid x \in A \text{ and } x \in B\}$.
- Star: $A^* = \{x_1, x_2 \cdots x_k \mid$

eg → A = {red, orange}.
B = {pineppple, apple, pen}.

A ∪ B ⇒ {red, orange, pineapple, apple, pen).

A ∘ B ⇒ {red pineapple .... }

★ Regular Expressions

- Expressions built from members of Σ, ϕ, ε (atomic) & using regular operations ($\cup$, $\circ$, *) on a language.

eg → $(0 \cup 1)^* = \Sigma^*$

$\Sigma^* 1$ = all strings that end in 1.

$\Sigma^* 11 \Sigma^* = L(M_1)$.

- Closure under '$\cup$' for Regular languages

Theorum :- If A_1 and A_2 are regular languages : so $A_1 \cup A_2$ is also a regular language.

Proof :- Let $M_1 = (Q_1, \Sigma_1, \delta_1, q_1, F_1)$ recognises A_1.

Let $M_2 = (Q_2, \Sigma_2, \delta_2, q_2, F_2)$ recognize A_2

Construct $M = (Q, \Sigma, \delta, q_0, F)$ recognize $A_1 \cup A_2$.

$Q = Q_1 \times Q_2$

$= [(q_1 q_2) \mid q_1 \in Q_1 \text{ and } q \in Q_2]$

$q_0 = (q_1, q_2)$ [starting state of M_1 & M_2]

$\delta((q,r), a) = [\delta_1(q,a), \delta(r,a)]$ [q & r are any general state in M_1 & M_2 resp].

$F = F_1 \times F_2$ is wrong as this means the machine must satisfy both final states

$= (F_1 \times Q_2) \cup (Q_1 \times F_2)$ is write, it should satisfy only 1 final state).

Σ is the same

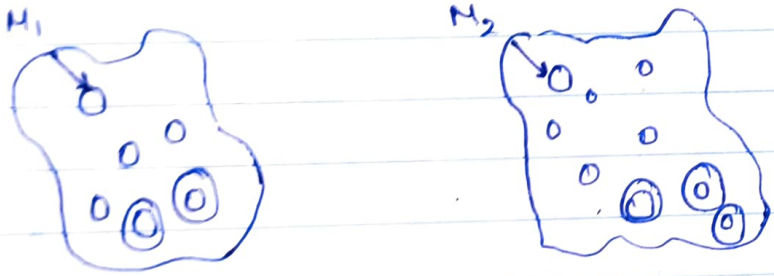
- Closure under ∘ for regular languages.

Theorem: If A_1 & A_2 are regular languages, so is $A_1 A_2$.

Proof: $M_1 = (Q_1, \Sigma, \delta_1, q_1, F_1)$ recognizes A_1

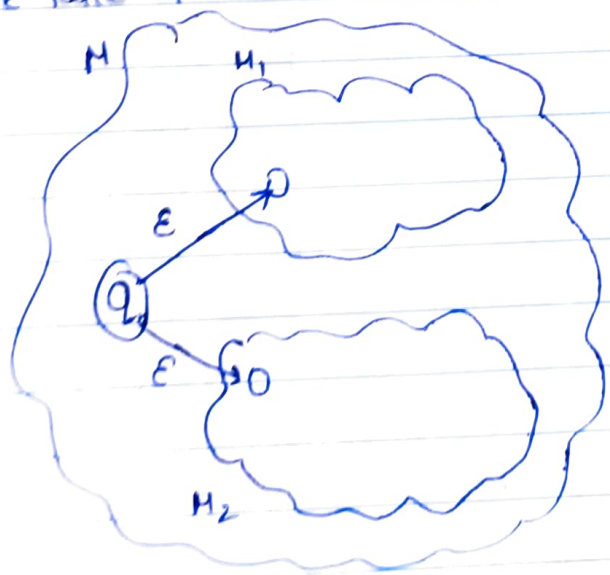
$M_2 = (Q_2, \Sigma, \delta_2, q_2, F_2)$ recognizes A_2

$M = (Q, \Sigma, \delta, q_0, F)$ recognizes $A \circ A_2$.

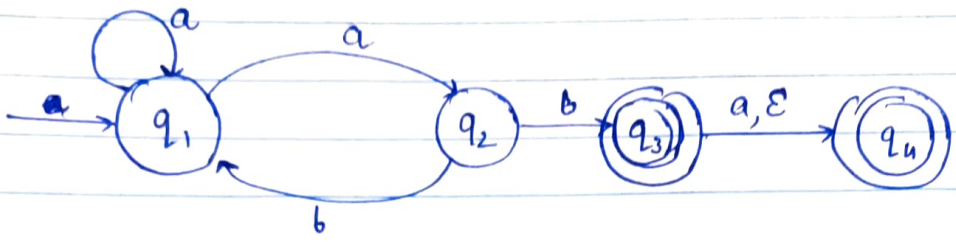


If we take M as a NFA

∴



★ Non-Deterministic Finite Automata (NFA).



- For q_1, 'a' can give back to q_1 or go to q_2
- ε transition means a if q_3 has a free move ~~from~~ to q_4.
- Accept if any one path leads to accepting state.

eg → ab → Accept aab → Accept.
aba → Accept abb → Reject.
aa → Reject.

- Formal Defination.
  - ↳ is also a 5 tuple $(Q, \Sigma, \delta, q_0, F)$.
    - → Q is same as FA
    - → Σ is same as FA
    - ★ → $\delta : (Q \times \Sigma_\varepsilon) \rightarrow P(Q)$ is a transition func^n.
    - → F ~~is set of~~ is same as FA

★ Convert NDFA to DFA.

- Theorum → If an NFA recognizes A, the A is regular.

Proof → Let NFA

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

Construct DFA

$M' = (Q', \Sigma, \delta', q_0', F')$ recognize A (ignore ε transitions)

IDEA DFA M' keeps track of the subset of possible states in NFA M.

1] $Q' = P(Q)$

2] $\delta'(R, a) = \{q \mid q \in \delta(r, a) \text{ for some } r \in R\}$

3] $q_0 = \{q_0\}$

4] $F = \{R \in Q \mid R \text{ intersects } F\}$.

Q] Design a FSM to check whether given binary no. is divisible by 4.

4 → 0100 ~~000 000~~,

$S \rightarrow \{q_s, q_0^*, q_1, q_2, q_3\}$

$I = \{0, 1\}$ → Input is binary no.

$O = \{Y, N\}$.

$F = \{q_0^*\}$.

* when we divide no. by 4, 4 remainders

0 q_0

1 q_1

2 q_2

3 q_3

	I / S	0	1
→	q_s	q_0	q_1
0	q_0^*	q_0	q_1
1	q_1	q_2	q_3
2	q_2	q_0	q_1
3	q_3	q_2	q_3

fig → State table.

$S \times I \rightarrow S$.

S / I	0	1
q_s	Y	N
q_0^*	Y	N
q_1	N	N
q_2	Y	N
q_3	N	N

fig → Machine Table

$S \times I \rightarrow O$.

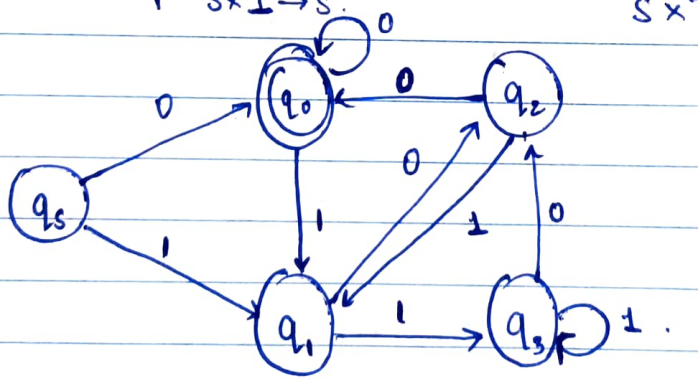


fig → State Transition Diagram.

$\delta(q_s, 1000)$

$\delta(q_1, 000$

$\delta(q_2, 00)$

$\delta(q_0, 0)$

q_0 → final state.

∴ Accept.

Q] Design a machine in which string is valid if it contains even no. of 0s & odd no. of 1s.

4 ways → even 0, even 1 | even 0, odd 1 | even 1, odd 0 | odd 0, odd 1 |

∴ 4 states → q_0 → e 1, e 0

q_1 → e 1, odd 0.

q_2 → odd 1, e 0

q_3 → odd 1, odd 0

∴ $S \rightarrow \{q_s, q_0, q_1, q_2, q_3\}$

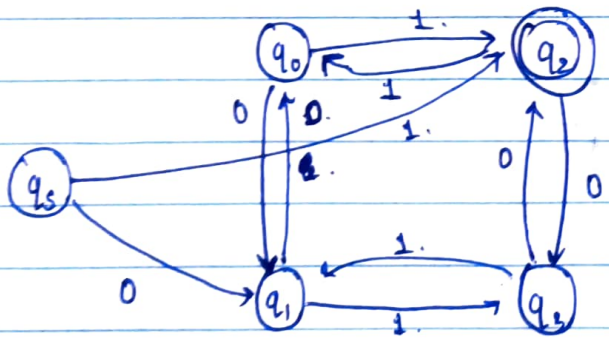
$I = \{1, 0\}$. $F \rightarrow \{q_2\}$.

$O \rightarrow \{Y, N\}$.

State transition

S \ I	0	1.
q_s	q_1	q_2
q_0	q_1	q_2
q_1	q_0	q_3
q_2	q_3	q_0
q_3	q_2	q_1

S \ I	0	1.
q_s	N	Y
q_0	N	Y
q_1	N	N
q_2	N	N
q_3	Y	N.



$\delta(q_s, 10101011010)$

$\delta(q_2, 0101011010)$

$\delta(q_3, 101011010)$

$\delta(q_1, 01011010)$

$\delta(q_0, 1011010)$

$\delta(q_2, 011010)$

$\delta(q_3, 11010)$

$\delta(q_1, 1010)$

$\delta(q_3, 010)$

$\delta(q_2, 10)$

$\delta(q_0, 0)$

$\delta \rightarrow q_1$

Q1] Design a machine in which string is valid if it ends with 110 over the input {0,1}.

Q2] Design a machine in which string is valid if it contains 1011 over the input {0,1}.

Ans 1] States → ends with 1 → q_0
ends with 11 → q_1
ends with 110. → q_2
ends with something else. → q_3.

$q_3 == q_s$

$S \rightarrow \{q_s, q_0, q_1, q_2, q_3\}$
$I \rightarrow \{1, 0\}$.

S \ I	1	0
q_s	q_0	q_3
q_0	q_1	q_3
q_1	q_1	q_2
q_2	q_1	q_3
q_3	q_0	q_3

S \ I	1	0
q_s	N	N
q_0	N	N
q_1	N	Y
q_2	N	N
q_3	N	N

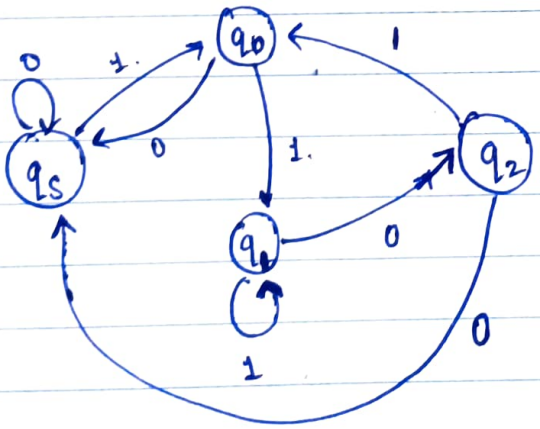


fig → finite state diagram.

$\delta(q_s, 10110)$
$\delta(q_0, 0110$
$\delta(q_s, 100)$
$\delta(q_0, 10)$
$\delta(q_1, 0)$
$\delta(q_2,)$
↳ Final State
Accepted

Q3] Design a machine in which string is valid if it starts with 'AA' & does not end with 'AA'.

★ Regular Expressions.

Def → It is algebraic expressions to represent a language which is accepted by finite automata.

Def^n → Regular expression is used to ~~define~~ denote regular language & is defined as.

1] ϕ is a regular expression to represent the language empty set
2] ϵ is a " " used " " " " " $\{\epsilon\}$.
3] a " " " " " " " " " $\{a\}$.
4]

Let us assume 'R' & 'S' be the two regular expressions used to represent the languages L_R & L_S respectively then
R|S is the regular expression used to represent regular language $L_R \cup L_S$.

5] R.S is the regular expression used to represent regular language. $L_R \cdot L_S$.

6] R^* be the regular exp. used to represent the language L_R^*

$R^* = R^0 \cup R^1 \cup R^2 \cdots$

eg $a^* =$

$a^0 = \epsilon$

$a^1 = a$

$a^2 = aa$

$a^3 = aaa$.

★ R^* (clean closure)

$R^* = \bigcup_{i=0}^{\infty} R^i$

R^+ (positive closure)

$R^+ = \bigcup_{i=1}^{\infty} R^i$

eg] $a^n b^n \mid n \geq 1$

Make DFA for above to check if in lang.

Ans] Not possible by FA. as we cannot count no. of a's & b's.

Thus, we can use the Push Down Automata.

~~baaabb~~

	RE	Language
1)	ε	$\{\varepsilon\}$
2)	a	$\{a\}$.
3)	~~(ab)~~ $^{(ab)}$	$\{ab\}$.
4)	a|b a+b	$\{a, b\}$.
5)	a^*	$\{\varepsilon, a, aa, aaa \ldots\}$.
6)	a^+	$\{a, aa, aaa, \ldots\}$.
7)	$(a+b)^*$	$\{\varepsilon, a, b, aa, ab, ba, bb, aaa, aab, aba, baa, \ldots\}$.
8)	$(aa)^*$	$\{\varepsilon, aa, aaaa, aaaaaa, \ldots\}$ → even length.
9)	$a(bb)^*$	$\{a, abb, abbbb, abbbbbb \ldots\}$.

Q] Construct regular expression for the following problem statements over the input (a, b).

1] There will be exactly 1 a → $L = \{a, ab, abb, ba, abb, bab, \ldots\}$.

$R.E = b^*ab^*$

2] Length of the string is exactly 2 → $L = \{aa, ab, ba, bb\}$.

$R.E = aa + ab + ba + bb.$

$= a(a+b) + b(a+b)$

$= (a+b)(a+b)$

3] Length of the string is atleast 2 → $L = \{aa \ldots\}$

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

4] String length atmost 2 → $L = \{0, 1, 2\}$.

$R.E = (\varepsilon + a + b)(a+b+\varepsilon)(a+b+\varepsilon)$

5) String ends with abb →

$(a+b)^*abb.$

6] String length is even over the input → $[(a+b)(a+b)]^*$

7] String length is odd over the input → $(a+b)[(a+b)(a+b)]^*$

8] String is divisible by 3 over input a, b. → $[(a+b)(a+b)(a+b)]^*$

9) String starts & ends with different letters. → $a(a+b)^*b + b(a+b)^*a$

10] string starts and ends with same symbol over input a, b.

$a(a+b)^*a + b(a+b)^*b + a + b + \varepsilon$

11] String starts with ab and ends with ba → $ab(a+b)^*ba + aba$.

~~NFA~~

FA

with final state
No output

1] DFA
2] NDFA

without final state.
with output

1] Moore machine
2] Mealy machine.

★ DFA

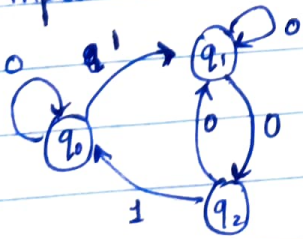
→ It consist of finite set of states, ~~1 state is~~
→ One state → start / initial state, &
→ Final state / Acceptable state.
→ From each input there will be exactly 1 transition.
→ DFA can be represented mathematically as follows

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

★ NFA

→ It consist of finite set of states.
→ Start state, Final state. / Acceptable state).
→ For each input there ~~will be only~~ 1 can be 0, 1, or more transitions.

eg →



q \ Σ	0	1
q_0	q_0	q_1
q_1	$\{q_2, q_1\}$	$\{\}$
q_2	q_1	q_0

★ Regular Expression to NFA

→ R.E = ϕ L = { }.

NFA

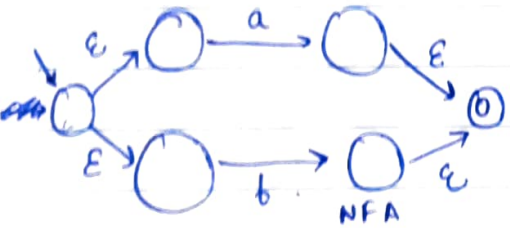
→ R.E = ε L = {ε}.

NFA

→ R.E = 'a'

NFA

→ R.E = 'a'|'b'

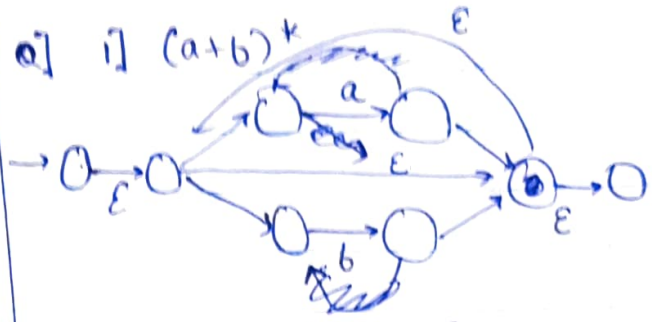


→ R.E = a.b.

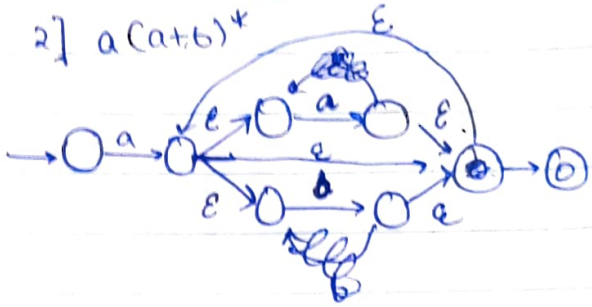
NFA.

→ R.E = a^* = {ε, a, aa, aaa ... }.

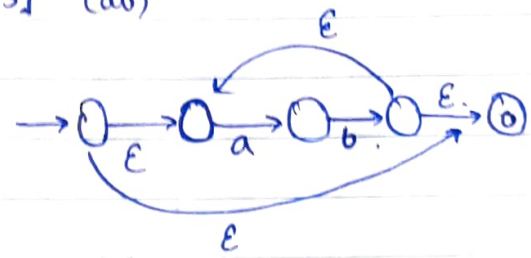
Q] 1] $(a+b)^*$



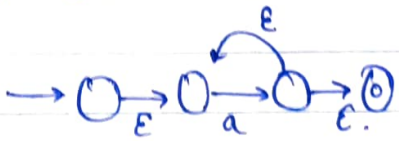
2] $a(a+b)^*$



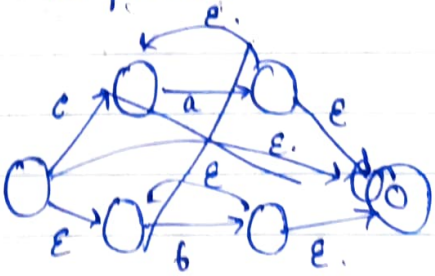
3] $(ab)^*$

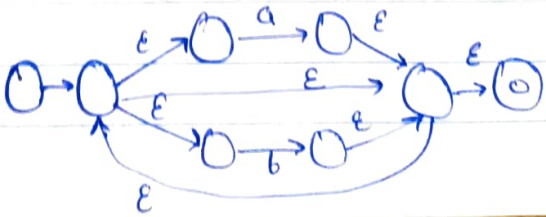


4) a^+



5] $a^* | (a+b)^*$

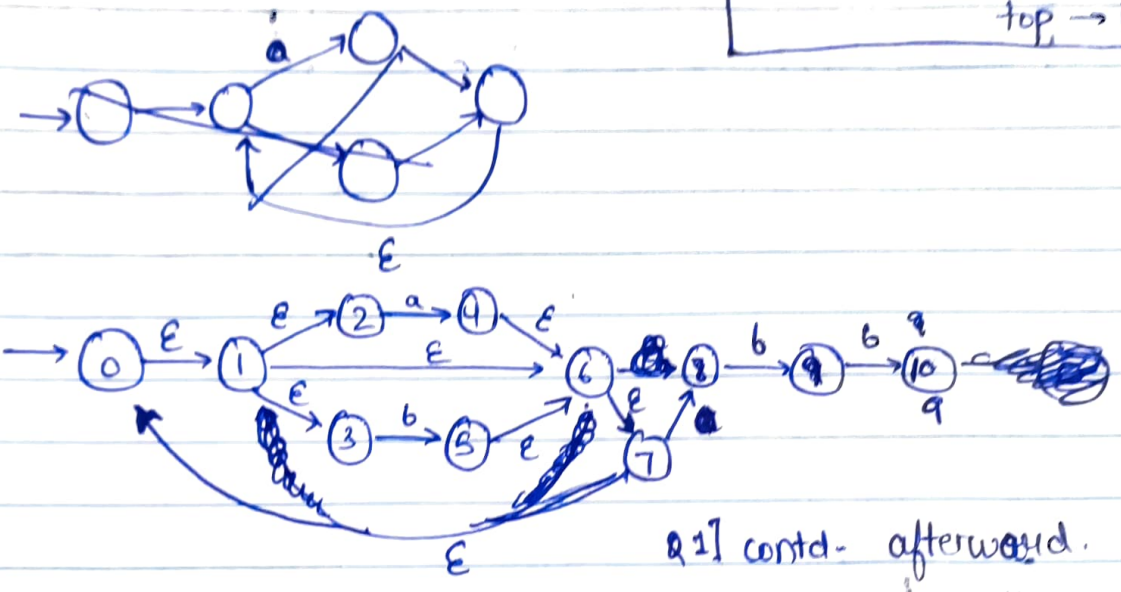




Q1] Construct NFA for accepting the string which ends with abb. over the i/p (a, b). Convert it into DFA and minimize DFA if possible.

Ans] R.E for $(a+b)^*abb$.

* For numbering states
convention → left → right
top → bottom



Q1] contd. afterward.

★ E-Closure of a state.

→ It is defined as a finite set of states which consists of ε transitions only for a state.

∴ eg → In above problem → e-closure of {0} = {0, 1, 2, 3, 6}.
e-closure of {4} = {6, 7, 8, 0, 1, 2, 3}.
e-closure of {8} = {8}.

★ Subset Formulation Method.

Next Page →

	x	y = ε-Closure()	δ(y, a)	δ(y, b)
A	{0}	{0, 1, 2, 3, 6, 7}	{4, 8}	{5}
B	{1, 8}	{0, 1, 2, 3, 6, 7, 8}	{4, 8}	{5, 9}
C	{5}	{0, 1, 2, 3, 6, 7}	{4, 8}	{5}
D	{5, 9}	{0, 1, 2, 3, 5, 6, 7, 9}	{4, 8}	{5, 10}
E	{5, 10}	{0, 1, 2, 3, 5, 6, 7, 10}	{4, 8}	{5}

* If the sets come in δ column it is replaced as x. If no new sets in δ then, all sets completed.

* Sets containing final state is the final state

A = {0, 1, 2, 3, 6, 7}.

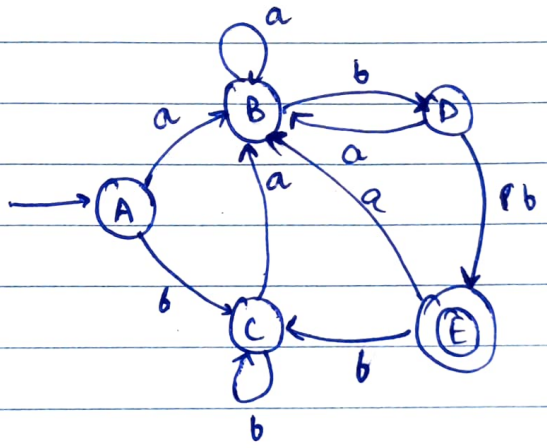
B = {0, 1, 2, 3, 6, 7, 8}.

C = {0, 1, 2, 3, 6, 7}.

D = {0, 1, 2, 3, 5, 6, 7, 9}.

E = {0, 1, 2, 3, 5, 6, 7, 10} → Final state as E contains 10.

States	a	b
A	B	C
B	B	D
C	B	C
D	B	E
E	B	C



→ A state is imp if there exist a transition on the given input ~~signe~~ symbol.

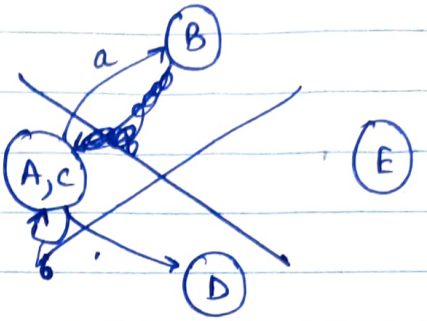
∴ Imp States → {2, 3, 7, 8, 9}.

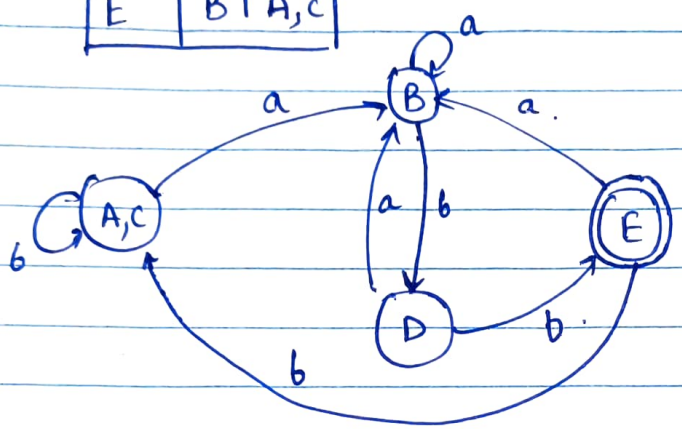
★ State Minimization using important State Method.

States can be ~~merge~~ merged if ⇒ {all the states contain same imp. states, all states include final state or all states exclude final states}

∴ Important states of A & C are same & don't contain final state.
So merge A & C.

	a	b
A,C	B	C
B	B	D
D	B	E
E	B	A,C

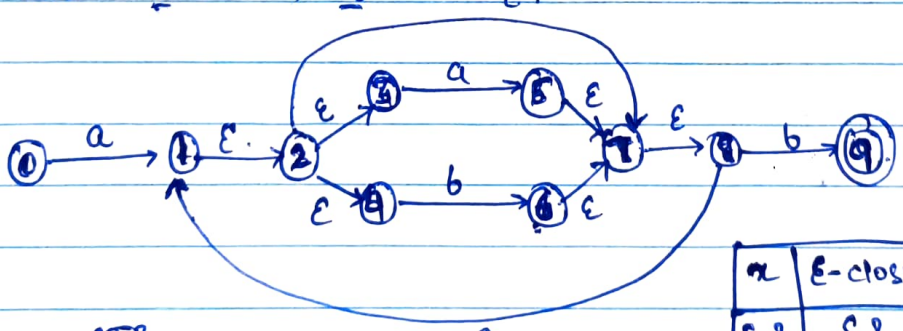




∴ fig → Minimized DFA.

Q2] Construct regular expression for string starts with a & ends with b. Convert to NFA, DFA, min. DFA.

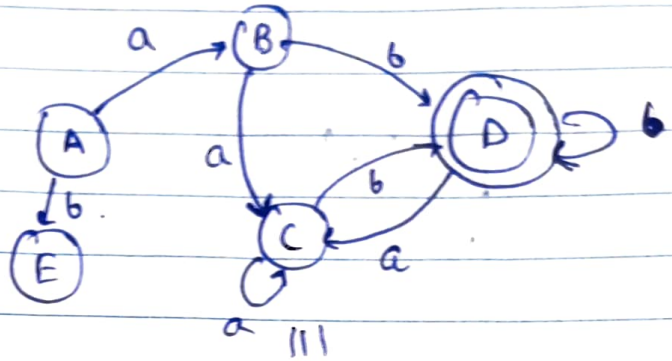
R.E = a (a+b)* b



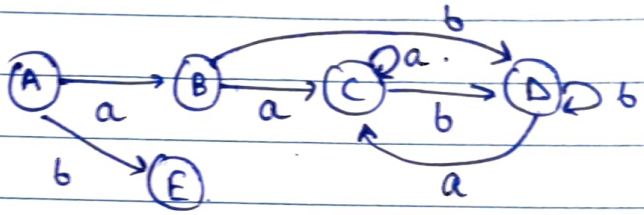
A → {0}.
B → {1, 2, 3, 4, 7, 8}
C → {1, 2, 3, 4, 7, 8}.
D → {1, 2, 3, 4, 7, 8}.
E = {}

	q	ε-closure	δ(q,a)	δ(q,b)
A →	{0}	{0}	{1,3}.	{}.
B →	{1}	{1,2,3,4,7}	{5}	{6,9}.
C →	{5}	{7,8,1,2,3,4}	{5}	{6,9}
D →	{6,9}	{7,8,1,2,3,4}	{5}	{6,9}
E →	{}	{}	{}	{}

	a	b.
A	B	E
B	C	D
C	C	D
D	C	D.
E	E	E



III

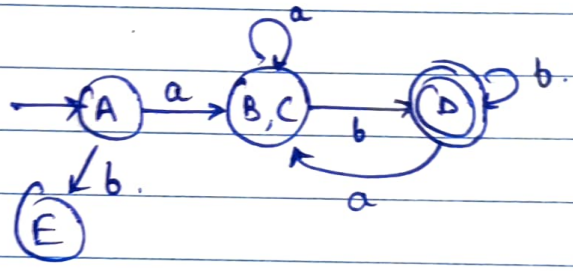


Imp → {0, 3, 4, 8}.

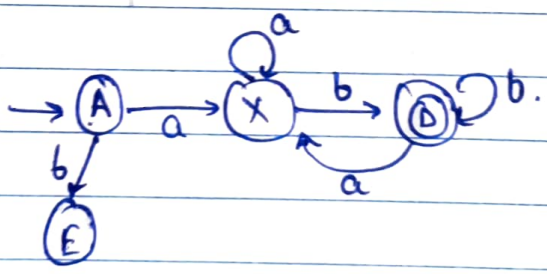
∴ B & C can be merged.

	a	b.
A	B,C	E
B,C	B,C	D
D	B,C	D.
E	E	E

B, C → X



	a	b
A	X	E
X	X	D
D	X	D
E	E	f



★ Finite Automata to Regular Expression (Arden's Theorum)

→ Let us assume 'P' & 'Q' be the regular expression define input Σ^*. If 'P' does not contain 'ε' then eqⁿ in R i.e.

$$\boxed{R = Q + RP}$$ has unique solⁿ.

$$\boxed{R = QP^*}$$

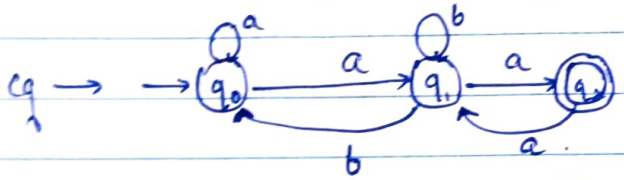
→ Procedure.

Step I : Construct the eqⁿ for each state.

Step II : Construct " " " " " consider incoming transitions only

Step III : Add ε in start state only.

Step IV : R.E represented in terms of final state which includes input symbols only.



$q_0 = q_0a + q_1b + \varepsilon$ —— ①

$q_1 = q_0a + q_1b + q_2a$ —— ②

$q_2 = q_1a$ —— ③.

Substitute ③ in ①.

$q_1 = q_0a + q_1b + q_1aa$

$q_1 = q_0a + q_1(b + aa)$.

$R = RQ + RP$

$\therefore R = QP^*$

$\therefore q_1 = q_0a(b + aa)^*$ —— ④

Substitute ④ in ①.

$q_0 = q_0a + q_0a(b+aa)^*b + \varepsilon$.

$q_0 = \varepsilon + q_0(a + a(b+aa)^*b)$

$R = Q + RP$ $\quad \therefore \boxed{q_0 = \varepsilon \cdot q(a + a(b+aa)^*b)^*}$ —— ⑤.

Sub eq ⑤ in eq ④.

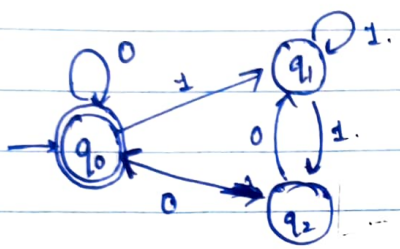
$q_1 = (a + a(b+aa)^* b)^* a(b+aa)^*$ — ⑥

Substitute ⑥ in ③

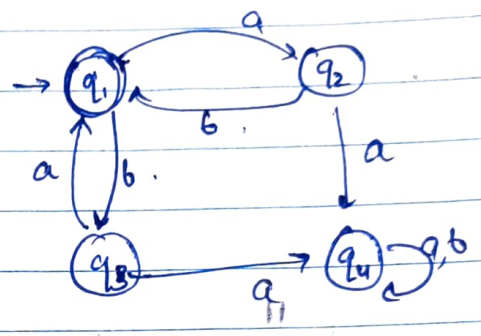
$q_2 = (a + a(b+aa)^* b)^* a(b+aa)^* a$

∴ R.E for given problem statement $(a + a(b+aa)^* b)^* a(b+aa)^* a$

Q1]



Q2]



Ans 1]

$q_0 = 0 \cdot q_2 + 0 \cdot q_0 + \varepsilon$ — ①

$q_1 = 1 \cdot q_1 + 1 \cdot q_0 + 0 \cdot q_2$ — ②

$q_2 = 1 \cdot q_1$ — ③.

$q_1 = 1 \cdot q_1 + 0 \cdot 1 \cdot q_1 + 1 \cdot q_0$

$q_1 = q_1(1 + 0.1) + 1 \cdot q_0$

$q_1 = 1 \cdot q_0 (1+0.1)^*$ — ④.

$q_0 = 1 q_1 + 0 \cdot q_0 + \varepsilon$

$q_0 = 1(1 \cdot q_0)(1+0.1)^* + 0 \cdot q_0 + \varepsilon$

$q_0 = (1 \cdot 1 \cdot (1+0.1)^* + 0) q_0 + \varepsilon$

$q_0 = (0 + 11(1+01)^* 0)^*$.

R.E = $(0 + 11(1+01)^* 0)^*$.

$q_1 = aq_3 + bq_2 + \varepsilon$

$q_2 = aq_1$

$q_3 = bq_1$

$q_4 = aq_4 + bq_4 + aq_2 + aq_3$

$q_1 = abq_1 + baq_1 + \varepsilon$

$q_1 = (ab+ba)^*$.

R.E = $(ab+ba)^*$.

★ ~~Mealy~~ Moore Machine.

→ It consist of finite set of states. It produces the output ~~seq~~ symbols from the given input ~~sequence~~ symbols

→ A symbol is associated with each state, such symbols are called as output symbols.

→ A moore machine can be represented mathematically as follows.

$M = \{Q, \Sigma, \Delta, \delta, \lambda, q_1\}$.

$Q \equiv$ Finite set of states.

$\Sigma =$ Input alphabet.

$\Delta =$ O/P ~~signal~~ alphabet.

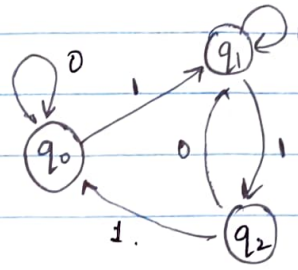
$\delta =$ transition func^n.

$Q \times E \to Q$.

$\lambda =$ mapping func^n.

$Q \to \Delta$

$q_0 \equiv$ initial state.



Moore M/c.

Q \ Σ	0	1
q_0	q_0	q_1
q_1	q_1	q_2
q_2	q_1	q_0

$\lambda \to Q \to \Delta$

$q_0 B_0 \to A$

$\& q_1 \to B$

$q_2 \to C$.

∴ For eq → input ⇒ 101.

$\delta(q_0, 101)$

$\delta(q_1, 01)$

$\delta(q_1, 1)$

$\delta(q_2)$ ∴ O/P → ABBC.