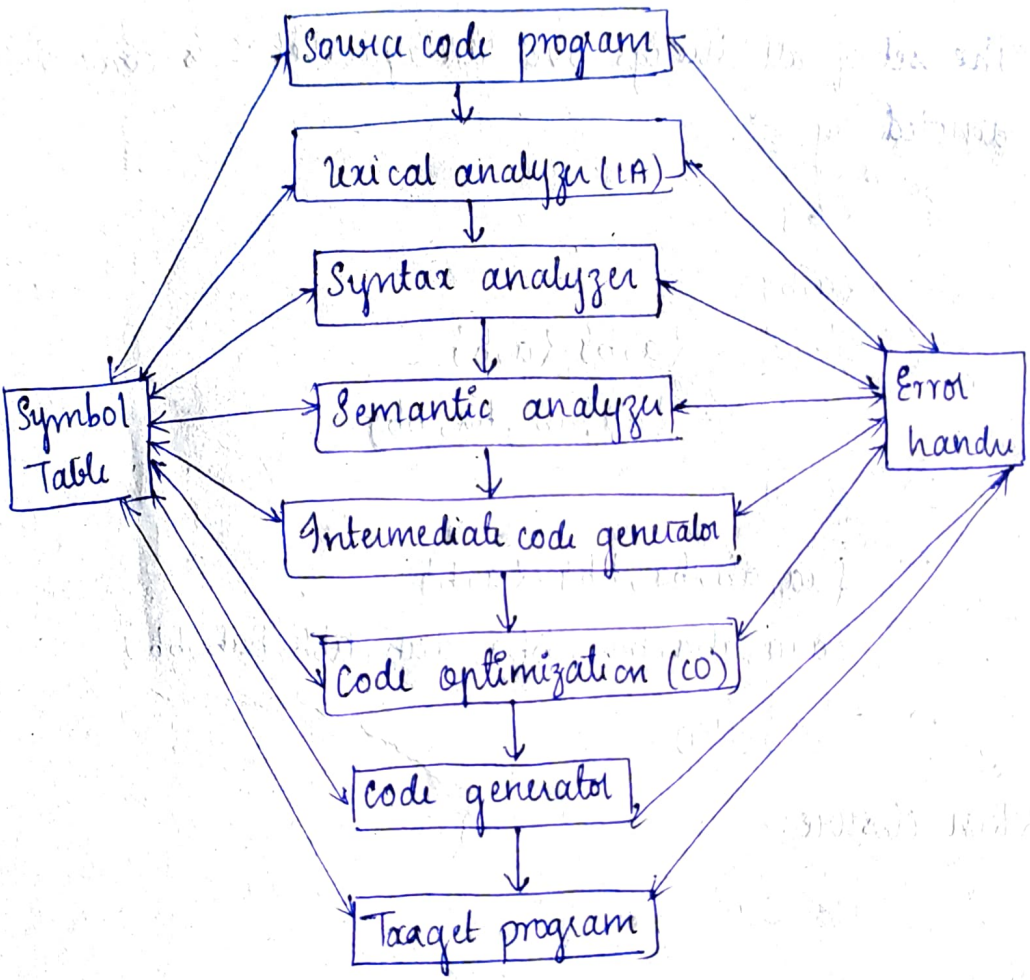
23/8/23

→ Compiler: a compiler is a program that helps in translating the compiler code from one programming language

* Phases of a Compiler



Alphabet : An alphabet is a finite set of symbols represented using Σ symbol

→ Alphabets can be binary alphabet which include 0 + 1

(or)

character alphabet which include the lower case alphabets a to z (or) A to Z

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

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

→ The set of all strings over an alphabet Σ is conventially denoted by Σ^*.

$\Sigma^0 = \{\epsilon\}$

$\Sigma^1 = \{a, b\}$

$\Sigma^2 = \Sigma^1 \cdot \Sigma^1 = \{a, b\} \cdot \{a, b\}$

$= \{aa, ab, ba, bb\}$

$\Sigma^3 = \Sigma^2 \cdot \Sigma^1$

$= \{aa, ab, ba, bb\} \cdot \{a, b\}$

$\{aaa, aba, baa, bba, aab, abb, bab, bb\}$

$\Sigma^* = \Sigma^0 \cup \Sigma^1 \cup \Sigma^2 \ldots$

★ Kleen Closure:

$\Sigma^* = \Sigma^0 \cup \Sigma^1 \cup \Sigma^2 \ldots$

★ Positive Closure:

$\Sigma^+ = \Sigma^1 \cup \Sigma^2 \cup \ldots$

The relationship between +ve closure & the kleen closure is $\boxed{\Sigma^* = \Sigma^+ \cup \{\epsilon\}}$

$0^* = \{\epsilon, 0, 00, 000, 0000 \ldots\}$

$1^* = \{\epsilon, 1, 11, 111, 1111, \ldots\}$

$0^+ = \{0, 00, 000, 0000 \ldots\}$

$1^+ = \{1, 11, 111, 1111, \ldots\}$

$0^* 1^* = \{\epsilon, 1, 0, 01, 001, 011 \ldots\}$

$0^* 1^+ = \{1, 01, 001, 011, \ldots\}$

$0^* 1 = \{1, 01, 001, 0001, \ldots\}$

$01^* = \{0, 01, 011, 0111, \ldots\}$

Languages:

→ A set of strings all of which are chosen from some Σ^* where Σ is a particular alphabet called as Language

→ if Σ is an alphabet & $L \subseteq \Sigma^*$ then L is a language over Σ

Ex: The language of all strings consisting if n 0's followed by n 1's for $n \geq 0$

→ The set of strings of 0's & 1's with an equal no. of each

$L = \{01, 10, 0011, 1100, 0101, 1010, \ldots\}$

→ The set of binary numbers whose value is a prime

$L = \{10, 11, 101, 111, 1011, \ldots\}$

→ $\emptyset$ the empty language is a language over an alphabet

→ $\{\epsilon\}$ the language consisting of empty string is also a lang

$\emptyset \neq \{\epsilon\}$

→ Construct a language for even no. of 0's & 1's

$L = \{00, 11, 00001111, 01010101, 1010, 001, \ldots\}$

Operations :

1) Union of sets (∪)

If A and B are the given sets then the resultant set on taking union of A & B may be represented as

$$C = \{x \mid x \in A \text{ or } x \in B\}$$

2) Intersection (∩)

$$C = \{x \mid x \in A \text{ and } x \in B\}$$

3) Disjoint sets

Two sets are said to be disjoint when ∩ of the 2 given sets results in a null set

Ex: $A = \{1, 2\}$ $B = \{3, 4\}$

$$A \cap B = \{\phi\}$$

4) Set difference:

$A - B$

It is a binary operation performed on 2 sets & result in a set of elements that are present in the 1st set & not present in the 2nd set

5) Compliment of a set (S')

$S' = U - S$ [U - universe]

This can be defined as set difference of universe set & given set.

6) Symmetric difference ($\oplus$)

$$A \oplus B = (A - B) \cup (B - A)$$

7) Cartesian product (x)

CP of 2 sets A & B is a set of all possible ordered pairs (a, b) where $a \in A$, $b \in B$

Strings:

A string is finite sequence of symbols choosen from some alphabet

Ex: 01101 is a string from the binary alphabet

$\Sigma = 0, 1$

Empty string: This is a string with 0 occurences of symbols this is represented by 'ϵ'

Length of a string: The no. of positions for symbols in the string

Ex: 01101 length = 5

Concatenation of string: Let x & y be strings then xy denotes the concatenation of x & y

Sub string: a string x is a substring of another string y if there are strings w & z either 1 or both of which may be null exist such that

$y = wxz$

if w is null then x is said to be prefix

if z is null then x is said to be sufix

25/8/23

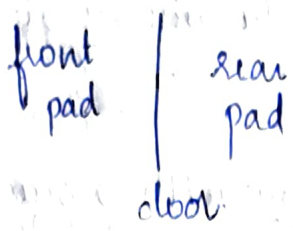
Automata: audomata deals with definitions & properties of mathematical models used in computation

This models play a major role in area of computer science

i) Finite automata (FA) used in compilers

ii) CFG (context free grammer) used in programming lang

Ex: automatic door in offices



front pad: the sensor detects the person who is about to enter

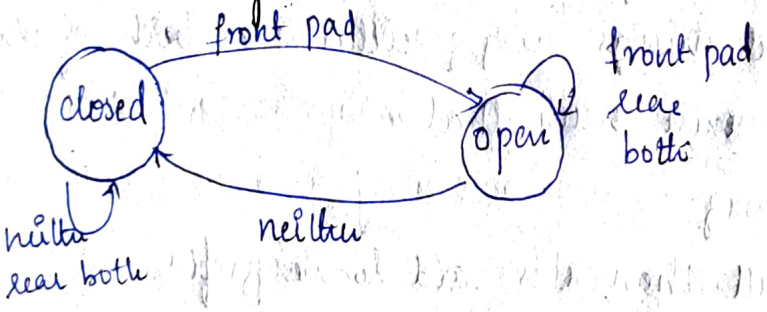
rear pad: the sensor detects the person behind the door

There are 2 states in automatic door - open & close.

four probabilities are:

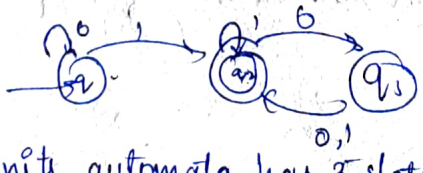
1) front pad 2) rear pad 3) Both 4) neither

Transition diagram:



Transition Table:

possibilities / States	front	rear	neither	Both
closed	O	C	C	C
opened	O	O	C	O



a finite automata has 3 states

3 states q_1, q_2, q_3 are given in above transition diagram

initial state q_1 is indicated by arrow

accept or final state q_2 is represented with double circles

The arrows goings from 1 state to another state are called transitions

I/p string 1101, check whether the o/p is accepted or rejected

i) start with q_1
ii) read '1', follow transition from $q_1 \rightarrow q_2$
iii) read '1', follow transition from $q_2 \rightarrow q_2$
iv) read '0', follow transition from $q_2 \rightarrow q_3$
v) read '1', follow transition from $q_3 \rightarrow q_2$
vi) accept

Ex: 1, 01, 11, 0101010101, 100, 0100, 110000, 0101000000 → are accepted strings

0, 10, 10100 → These are not accepted

Finite State Machine:

→ The FS sys represents a mathematical model of a sys with certain i/p, the model finally gives certain o/p

→ The i/p when is given to the machine it is processed by various states

→ These states are called "intermediate states

Types of automata :

deterministic — non-deterministic finite automata

A FA is a collection of 5 tuple $(Q, \Sigma, \delta, q_0, F)$

Q - finite set of state which is non empty

Σ - input alphabet

δ - transition

q_0 - initial state i.e $q_0 \in Q$

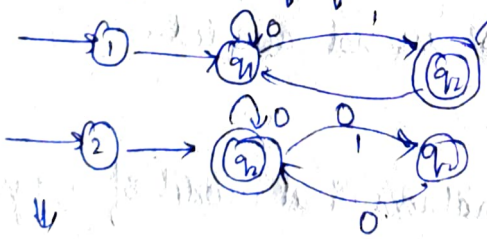
F - set of final states $F \subseteq Q$

DFA :

In DFA there is only one path for a specific i/p from current state to next state.

The DFA can be represented by the same 5 tuple described in the definition of finite automata.

Ex :



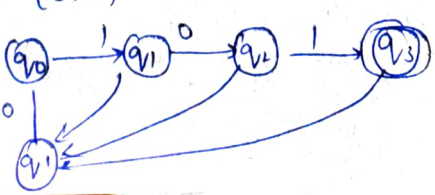
$\{q_1, q_2\}, \{0,1\}, \delta \{q_1, 1\}, \{q_2, 1\}$

$(\{q_1, q_2\}, \{1, 0\}, \delta, \{q_1\}\{q_1\})$

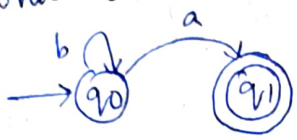
State	0	1
q_1	q_1	q_2
q_2	q_1	q_2

* define FA which accepts the i/p 101 over the i/p alphabet

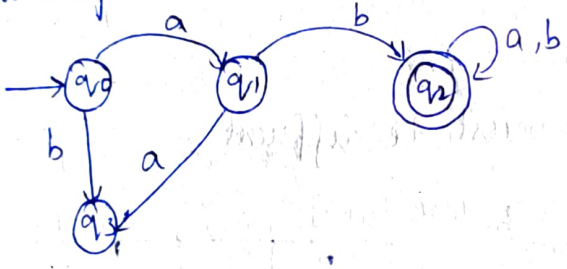
$\Sigma = (0,1)$



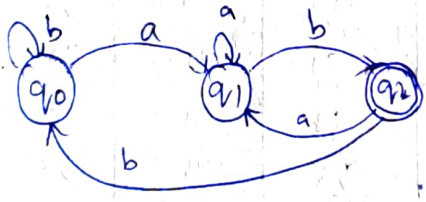
* design DFA in which set of all string can be accepted which ends with 'a' where $\Sigma = a, b$.



starting with ab



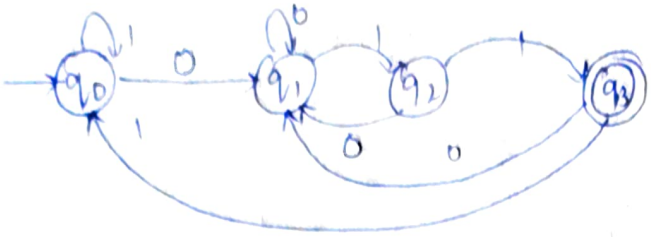
end with ab



self loop nearest state it has to go one state in DFA

s \ b	a	b
q_0	q_1	q_0
q_1	q_1	q_2
q_2	q_1	q_0

* Draw a DFA to accept string of 0's & 1's ending with the string 011

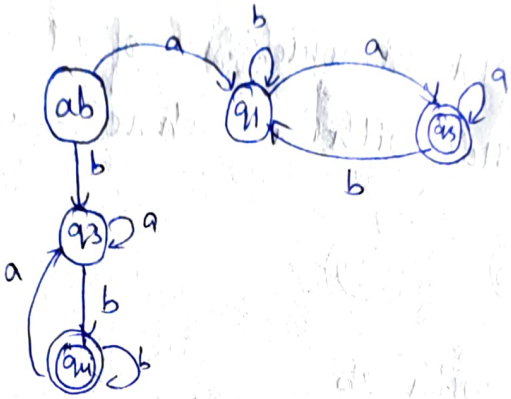


δ	0	1
q_0	q_1	q_0
q_1	q_1	q_2
q_2	q_1	q_3
q_3	q_1	q_0

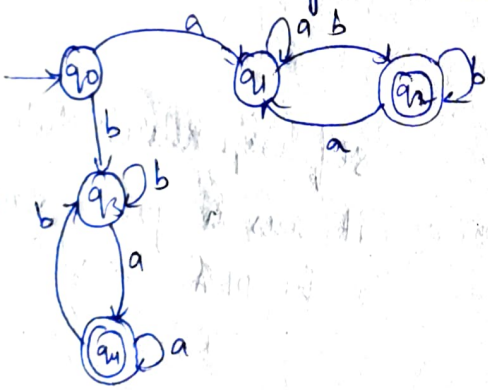
$\{(q_0, q_1, q_2, q_3), (0,1), \delta, q_0, q_3\}$

Q) start & end must be same $\Sigma a, b$

δ	a	b
q_0	q_1	q_3
q_1	q_2	q_1
q_2	q_2	q_1
q_3	q_3	q_4
q_4	q_3	q_1



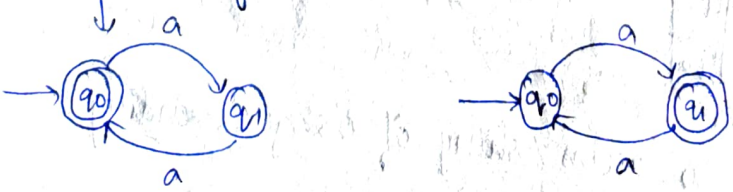
start and end symbol must be different



δ	a	b
q_0	q_1	q_3
q_1	q_1	q_2
q_2	q_1	q_2
q_3	q_4	q_3
q_4	q_4	q_3

Q) all binary numbers are divisible by 2.

Q) even no. of a's

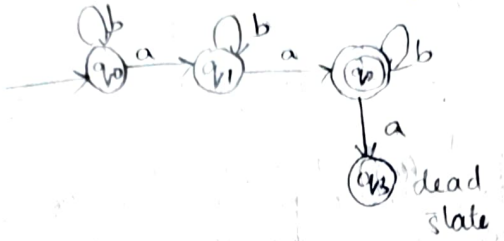


29/8/23

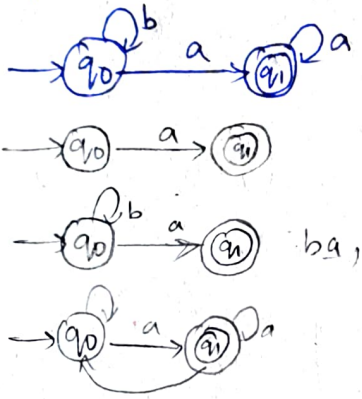
2) Design a DFA such that number of 'a' equal to 2 & there is no restriction over length of 'b'

DFA: 5-tuple = {a, b}

{Q, Σ, δ, q0, F}



2) Design a DFA in which set of all strings can be accepted which ends with a, where Σ = a, b



a ends with a

ba, baa, baaa

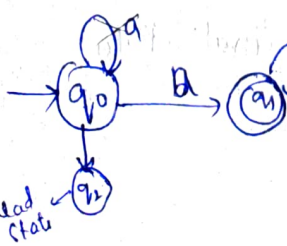
baaba

{{q0, q1}, {a, b}, δ, q0, q1}

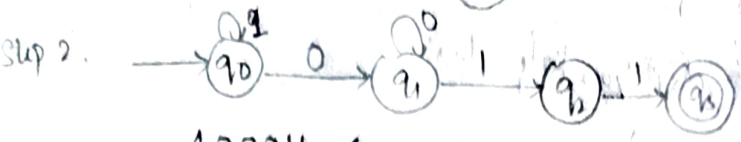
δ can represented as

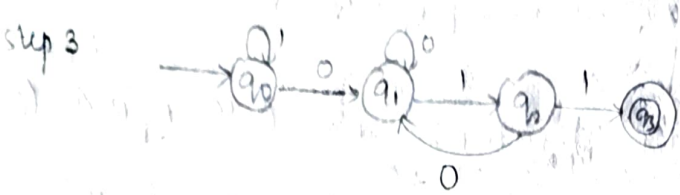
state \ i/p	a	b
q0	q1	q0
q1	q1	q0

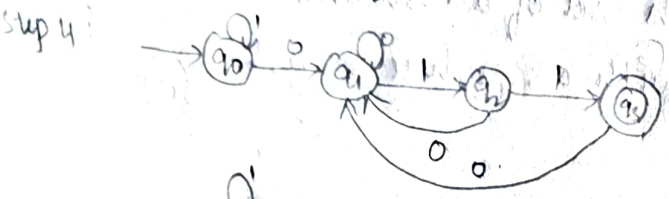
* starts with a.

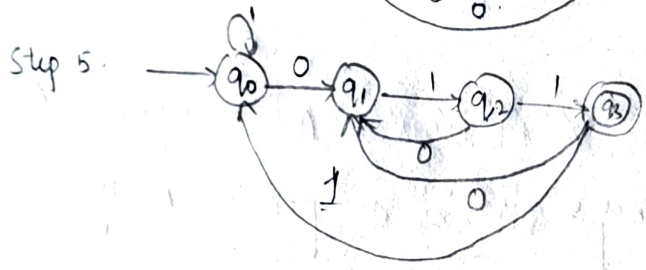


Q) Draw a DFA to accept set of strings {0,1} ending with string 011.

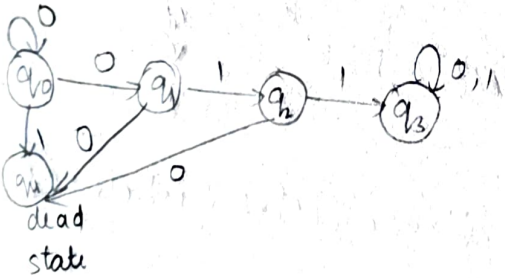




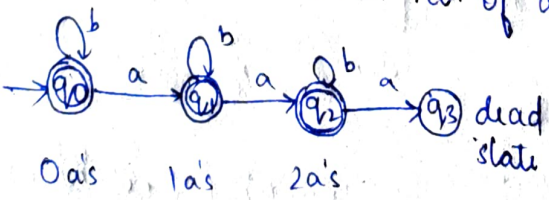




starts with 011

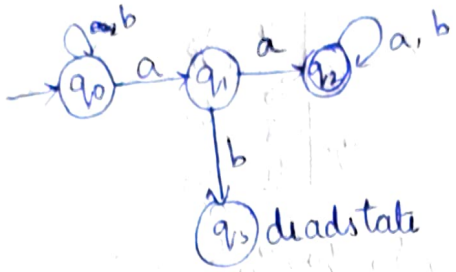


Q) design a DFA in which no. of a's in string are ≤ 2



since it should accept 0, 1, 2 a's all are final state

8) Design a DFA such that 2nd symbol from LHS is 'a' over $\Sigma = a, b$



31/8/23

8) Design DFA in which no. of a's are divided by 2 & i/p alphabet is a, b

$\{\{q_0, q_1\}, \{a, b\}, \delta, q_0, q_0\}$

δ can be represented as

I/P's / States	a	b
final q_0	q_1	q_0
q_1	q_0	q_1

8) Design a DFA such that it has/contain the substring as ab

	a	b
q_0	q_1	q_0
q_1	q_1	q_2
q_2	q_2	q_2

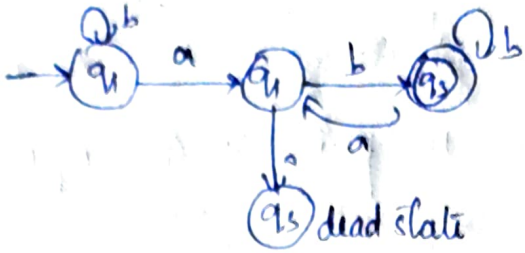
$\{\{q_0, q_1, q_2\}, \{a, b\}, \delta, q_0, q_2\}$

8) Design a DFA such that a should never be followed by

$\{\{q_0, q_1\}, \{a, b\}, \delta, q_0, \{q_0, q_1\}\}$

	a	b
q_0	q_1	q_0
q_1	q_1	

→ Every a should be followed by b

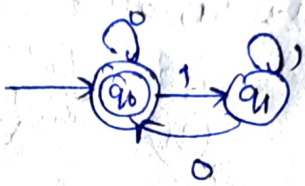


	a	b
q_0	q_1	q_0
q_1	q_0	q_2
q_3	q_1	q_3

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

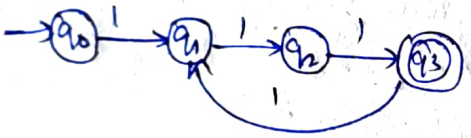
Q) Design DFA such that all binary no's are divided by 2.

(or)



(complement)

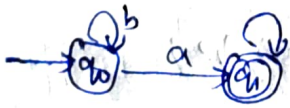
Q) Design a FA (consider DFA) which checks whether the given binary no is div by 3 ≤ 1



	1
q_0	q_1
q_1	q_2
q_2	q_3
q_3	q_1

$\{\{q_0, q_1, q_2, q_3\}, \{1\}, \delta, q_0, q_3\}$

Q) that accepts the string with atleast 1 'a'



	a	b
q_0	q_1	q_0
q_1	q_1	q_1

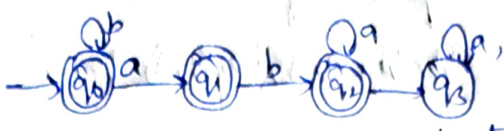
$\{\{q_0, q_1\}, \{a, b\}, \delta, q_0, q_1\}$

Q) Atleast 2 a's

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

	a	b
q_0	q_1	q_0
q_1	q_2	q_1
q_2	q_2	q_2

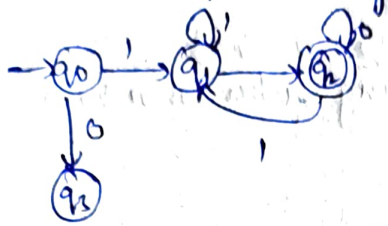
Q) such that 'a' should never be followed by 'b'



	a	b
q_0	q_1	q_0
q_1	q_1	q_2
q_2	q_2	-

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

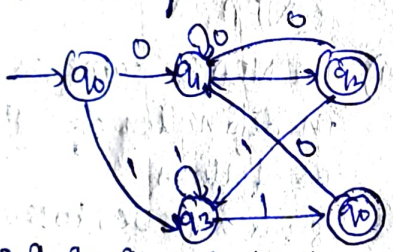
Q) which accepts strings & starts with 1 & ends with 0



	0	1
q_0	q_1	q_0
q_1	q_1	q_2
q_2	q_1	-

$\{\{q_0, q_1, q_2\}, \{0, 1\}, \delta, q_0, q_2\}$

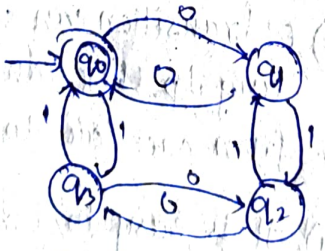
Q) accept strings that ends with 01 or 10



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

$\{\{q_0, q_1, q_2, q_3, q_4\}, \{0, 1\}, \delta, q_0, \{q_2, q_4\}\}$

Q) Which accepts even no of 0's & 1's



This FA will consider 4 diff stages for i/p 0 & 1

1) Even no of 0's & even no of 1's $\leftrightarrow q_0$ (final state)
2) Even no of 0's & odd no of 1's $\leftrightarrow q_3$
3) odd no of 0's & even no of 1's $\leftrightarrow q_1$
4) odd no of 0's & odd no of 1's $\leftrightarrow q_2$

NFA : (Non deterministic finite automata) - An NFA is represented essentially like a DFA i.e 5 tuple representation $(Q, \Sigma, \delta, q_0, F)$

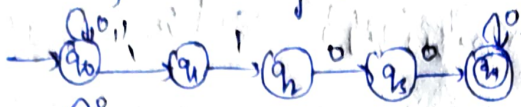
An NFA we can go to set of states over the current i/p symbol

Ex: Design NFA accepting all strings ending with 01

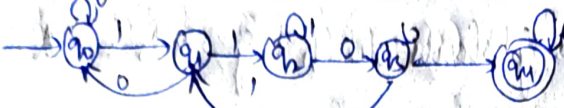
$\{q_0, q_1, q_2\}, \{0,1\}, \delta, q_0, q_2\}$

→ 11 is followed by 00

NFA

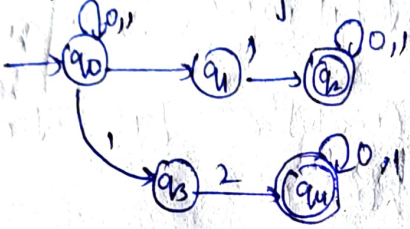


DFA



NFA that accepts the string containing either 01 or 10 over

$\Sigma = (0,1)$



DFA	NFA
1) It is a 5 tuple representation i.e $M = Q, \Sigma, \delta, q_0, F$ $\delta: Q \times \Sigma \to Q$	1) The NFA is same as DFA except in defination of δ. Here δ is as follows $\delta: Q \times (\Sigma \cup e)$ to subset of
2) There can be 0/1 transition on i/p symbol	2) 0 or more transitions
3) No ϵ transition exists, i.e there should not be any transition on i/p symbol	3) ϵ transition can select ie without any i/p these can be transit from one state to another
4) It is difficult to construct	4) It is easy to construct
5) It takes more space but have speed of computation	5) NFA's are more compact but takes more time
6) Can recognize a regular Language.	6) NFA is

Construct DFA for the following strings

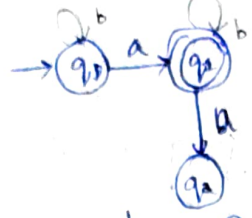
a) Exactly one 'a'

b) atleast one 'a'

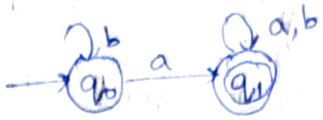
c) atmost one 'a'

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

a)



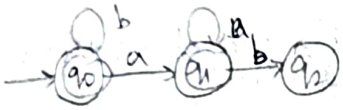
b)



c)

{ atmost 1 a means 0 / 1 so both q0 & q1 are final state }

→ Construct DFA that doesn't contain substring as ~~ab~~ ab is



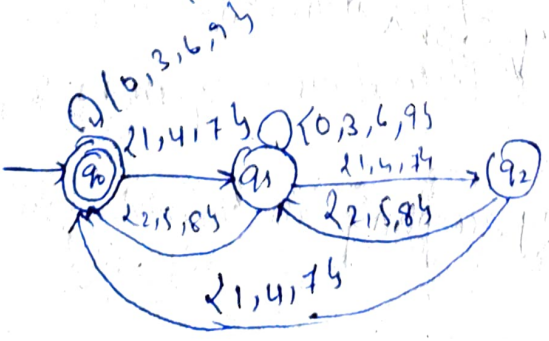
** → Construct DFA for decimal no.s divisible by 3

$\Sigma = \{0,1,2,3,4,5,6,7,8,9\}$

rem 0 → {0, 3, 6, 9}

rem 1 → {1, 4, 7}

rem 2 → {2, 5, 8}



DFA

Regular Expressions

- Let Σ be an alphabet which is used to denote the input the regular exp of Σ can be defined as follows
- ϕ is a regular exp which denotes the empty set
- ϵ is a regular exp and denotes set ϵ (epsilon) and it denotes a null string
- For each 'a' in Σ a is regular exp & denotes the set a
- If r and s are regular expressions denoting the languages L_1 and L_2 respectively then
  a) R+S is equivalent to $L_1 \cup L_2$ i.e union
  b) rs is equivalent to $L_1 . L_2$ i.e concatanation
  c) r^* is equivalent to L_1^* i.e closure

Note: r^* is known as clean closure or closure which indicates occurence of r for infinity no. of times.

Ex: $R.\epsilon = a^* = \{\epsilon, a, aa \ldots\}$

+ve closure of L can be shown as L^+. The L^+ denotes set of all the strings except the epsilon / null string.
The null string is denoted by ϵ or $\wedge$

$R.\epsilon = a^+ = \{a, aa \ldots\}$

ex: Write a RE for the lang accepting all combinations of a over the set $\Sigma = a$.

$RE = a^*$

Design a RE for the lang accepting all the combination of a's over $\Sigma = a$

$R.E = a^+$

Design R.E for lang containing all the strings containing any no of a's & b's

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

Construct R.E for lang containing all string having any no of a's & b's except the null string

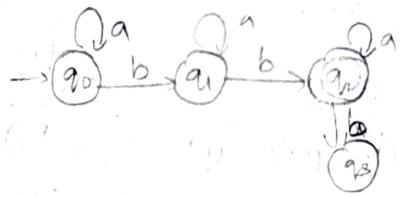
$RE = (a+b)^+$

1) All the strings which are ending 00 over the set $\Sigma = (0,1)$

$R.E = (0+1)^*00$

ex: consisting of exactly 2 b's over the set $\Sigma = a, b$

$R.E = a^*ba^*b$



ex: Write RE which denotes a lang 'L' over the set $\Sigma = 1$ having odd length of strings.

$RE = 1(11)^*$

$1(\epsilon) = 1$ $1(11)^1 = 111$ $1(11)^2 = 11111$

ex: construct RE which denotes lang over the $\Sigma = 0$ having even length of string

$RE = (00)^*$

ex: write RE to denote lang L over $\Sigma = a, b$ such that all the strings do not contain the sub string ab

$RE = b^*a a^*$

Identity Rules:

→ The two R.E P & Q are equivalent if & only if P represents the same set of strings as Q does

→ Let P, Q, R are R.E then the identity rules are as given below.

1. $\epsilon R = R\epsilon = R$
2. $\epsilon^* = \epsilon$
3. $\emptyset^* = \epsilon$
4. $\emptyset R = R\emptyset = \emptyset$
5. $\emptyset + R = R$
6. $R + R = R$
7. $RR^* = R^*R = R^+$
8. $(R^*)^* = R^*$
9. $\epsilon + RR^* = R^*$
10. $(P+Q)R = PR + QR$
11. $(P+Q)^* = (P^*Q^*)^* = (P^* + Q^*)^*$
12. $R^*(\epsilon + R) = (\epsilon + R)R^* = R^*$
13. $R + (\epsilon)^* = R^*$
14. $\epsilon + R^* = R^*$
15. $(PQ)^*P = P(QP)^*$
16. $R^*R + R = R^*R$

6/9/23

★ NFA to DFA convertion

Let $M = (Q, \Sigma, \delta, q_0, F)$ is an NFA which accepts the language $L(M)$ there should be equivalent DFA denoted by $M' = (Q', \Sigma', \delta', q_0', F')$ such that $L(M) = L(M')$

The convertion method will follow the following steps.

St 1: The start state of NFA will be start state for DFA Hence add q_0 of NFA to Q' then find the transitions from this start state

St 2: For each state $[q_0, q_1, q_2 \ldots]$ in Q' for each

a) the transitions for each input symbol Σ can be obtained as

$\delta'([q_1, q_2 \ldots], a) = \delta(q_1, a) \cup \delta(q_2, a) \ldots$

St 2: b) Add the state $[q_1, q_2 \ldots]$ to DFA if it is not already added in Q'

c) Then find the transitions for every i/p symbol from Σ for state $[q_1, q_2 \ldots]$

If we get some state $[q_1, q_2, q_3 \ldots]$ which is not in Q' of DFA then add the state to Q'

d) If there is no new state generating then stop the process after finding all the transitions

St 3: For the state $[q_1, q_2 \ldots] \in Q'$ of DFA if any state is a final state of NFA then $[q_1, q_2 \ldots]$ becomes a final state. Thus the set of all the final states belongs to F' of DFA.

10M

Q) Convert the given NFA to DFA.

State \ i/p	0	1
q_0	$\{q_0, q_1\}$	q_0
q_1	q_2	q_1
q_2	q_3	q_3
((q_3))	$\emptyset$	q_2

q_0 is the initial state in NFA so q_0 is initial state in DFA

$\delta'([q_0], 0) = \delta(q_0, 0) = \{q_0, q_1\}$

$\delta'([q_0], 1) = \{q_0\}$

$\delta'([q_0,q_1],0) = \delta(q_0,0) \cup \delta(q_1,0)$

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

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

$\delta'([q_0,q_1],1) = \delta(q_0,1) \cup \delta(q_1,1)$

$\{q_0\} \cup \{q_1\}$

$\{q_0,q_1\}$

p.s

$\delta'([q_0,q_1,q_2],0) = \delta'(q_0,0) \cup \delta'(q_1,0) \cup \delta'(q_2,0)$

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

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

$\delta'([q_0,q_1,q_2],1) = \delta'(q_0,1) \cup \delta'(q_1,1) \cup \delta'(q_2,1)$

$\{q_0\} \cup \{q_1\} \cup \{q_3\}$

$\{q_0,q_1,q_3\}$

$\delta'([q_0,q_1,q_2,q_3],0) = \delta'(q_0,0) \cup \delta'(q_1,0) \cup \delta'(q_2,0) \cup \delta'(q_3,0)$

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

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

$\delta'([q_0,q_1,q_2,q_3],1) = \delta'(q_0,1) \cup \delta'(q_1,1) \cup \delta'(q_2,1) \cup \delta'(q_3,1)$

$= \{q_0\} \cup \{q_1\} \cup \{q_3\} \cup \{q_2\}$

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

$\delta'([q_0,q_1,q_3],0) = \delta'(q_0,0) \cup \delta'(q_1,0) \cup \delta'(q_3,0)$

$= \{q_0,q_1\} \cup \{q_2\} \cup \{\emptyset\}$

$\{q_0,q_1,q_2\}$

$\delta'([q_0, q_1, q_3], 1) = \delta'(q_0, 1) \cup \delta'(q_1, 1) \cup \delta'(q_3, 1)$

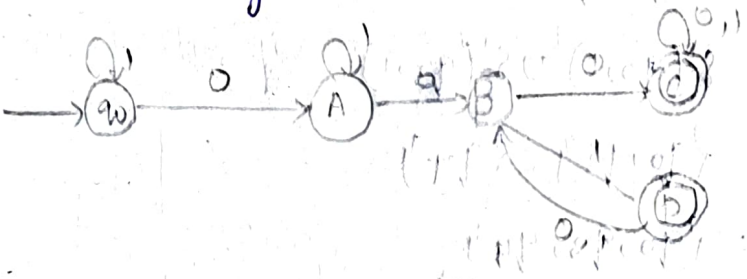
$= \{q_0\} \cup \{q_1\} \cup \{q_2\}$

$\{q_0, q_1, q_2\}$

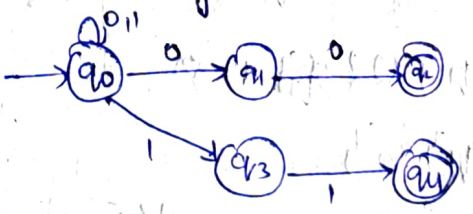
Transition table for DFA:

State \ i/p	0	1
q_0 A	$\{q_0, q_1\}$ B	q_0 A
$\{q_0, q_1\}$ B	$\{q_0, q_1, q_2\}$ C	$\{q_0, q_1\}$ D
$\{q_0, q_1, q_2\}$ C	$\{q_0, q_1, q_2, q_3\}$ C	$\{q_0, q_1, q_2\}$ C
$\{q_0, q_1, q_3\}$ D	$\{q_0, q_1, q_2\}$ B	$\{q_0, q_1, q_2\}$ B

Transition diagram.



Q) Convert the given NFA to equivalent DFA



State \ i/p	0	1
q_0	$\{q_0, q_1\}$	$\{q_0, q_3\}$
q_1	$\{q_2\}$	$\emptyset$
q_2	$\emptyset$	$\emptyset$
q_3	$\emptyset$	q_4
q_4	$\emptyset$	$\emptyset$

Initial state in DFA is q_0, so initial state in DFA

$\delta'(q_0, 0) = \{q_0, q_1\}$

$\delta'(q_0, 1) = \{q_0, q_3\}$

$\delta'([q_0, q_1], 0) = \delta'(q_0, 0) \cup \delta'(q_1, 0)$

$\{q_0, q_1\} \cup \{q_2\}$

$\{q_0, q_1, q_2\}$.

$\delta'([q_0, q_1], 1) = \delta'(q_0, 1) \cup \delta'(q_0, 1)$

$= \{q_0, q_3\} \cup \{\emptyset\}$

$= \{q_0, q_3\}$

$\delta'([q_0, q_3], 0) = \delta'(q_0, 0) \cup \delta'(q_3, 0)$

$= \{q_0, q_1\} \cup \emptyset$

$= \{q_0, q_1\}$

$\delta'([q_0, q_3], 1) = \delta'(q_0, 1) \cup \delta'(q_3, 1)$

$= \{q_0, q_3\}, \{q_4\}$

$= \{q_0, q_3, q_4\}$

$\delta'([q_0, q_1, q_2], 0) = \delta'(q_0, 0) \cup \delta'(q_0, 0) \cup \delta'(q_2, 0)$

$= \{q_0, q_1\} \cup \{q_1\} \cup \emptyset$

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

$\delta'([q_0, q_1, q_2], 1) = \delta'(q_0, 1) \cup \delta'(q_1, 1), \cup \delta'(q_2, 1)$

$= \{q_0, q_3\} \cup \{\emptyset\} \cup \{\emptyset\}$

$\{q_0, q_3\}$

$\delta'([q_0,q_3,q_4],0) = \delta'(q_0,0) \cup \delta'(q_3,0) \cup \delta'(q_4,0)$

$= \{q_0,q_1\} \cup \emptyset \cup \emptyset$

$\{q_0,q_1\}$

- $\delta'([[q_0,q_3,q_4],1] = \delta'(q_0,1) \cup \delta'(q_3,1) \cup \delta'(q_4,1)$

$= \{q_0,q_3\} \cup \{q_4\} \cup \emptyset$

$\{q_0,q_3,q_4\}$

Transition table for DFA

state \ i/p	0	1
q_0	$\{q_0,q_1\}^A$	$\{q_0,q_3\}^B$
$\{q_0,q_1\}^A$	$\{q_0,q_1,q_2\}^C$	$\{q_0,q_3\}^B$
$\{q_0,q_3\}^B$	$\{q_0,q_1\}^A$	$\{q_0,q_3,q_4\}^D$
$\{q_0,q_1,q_2\}^C$	$\{q_0,q_1,q_2\}^C$	$\{q_0,q_3\}^B$
$\{q_0,q_3,q_4\}^D$	$\{q_0,q_1\}^A$	$\{q_0,q_3,q_4\}^D$