UNIT-1: Formal Proof and Automata

Introduction to formal proof - Finite Automata (FA) - Deterministic finite Automata (DFA) - Non-deterministic finite Automata (NFA) - Equivalence between NFA and DFA - Finite Automata with Epsilon transitions - conversion of NFA into DFA - Equivalence and minimization of automata.

UNIT-2: Regular Expressions and properties of regular languages.

Regular expression - Equivalence of finite automata and regular expressions - Proving languages not to be regular (Pumping Lemma) - closure properties of regular languages.

UNIT-3: Context Free Grammars and Push Down Automata (PDA)

Context-Free Grammar (CFG) - Parse trees - Ambiguity in grammars and languages. Push Down Automata - Definition of the pushdown Automata (PDA) - Languages of PDA - Equivalence of PDA and CFG - Deterministic Pushdown Automata.

UNIT-4: Context Free Languages and Turing Machines

Normal Forms for CFG - chomsky Normal Form and Greibach Normal Form - Pumping lemma for CFL - closure properties of CFL - Turing machines: Basic model - definition and representation - Instantaneous Description - Language acceptance by TM - Transition diagram for TM - Language acceptance by TM - Programming TM as computer of Integer functions - Programming techniques for Turing machines (subroutines)

UNIT-5: Undecidability

Language that is not Recursively Enumerable (RE) - An undecidable problem that is RE - Undecidable problems about Turing machine - Post's correspondence problem - The classes P and NP - Kruskal's algorithm - Traveling salesman Problem.

26/07/24

UNIT - 1

INTRODUCTION TO FORMAL PROOF

The formal proof is required to check the correctness of the program or model developed from automata theory.

There are two types of proof namely.

1) Deductive proof
2) Inductive proof.

DEDUCTIVE PROOF:

* The sequence of statement whose truth ~~are~~ or conclusion are detrived from the given statement is called hypothesis (H)
* Each step of the proof follows the previous step or use some mathematical laws

Eg: If H then C.

INDUCTIVE PROOF:

This is used to prove recursively defined structure with three terms: Basic, Assumption, Induction.

Proof by Deduction:

2m

1) Prove the statement if $\underbrace{x \geq 4}_{}$ H then $2^x \geq x^2$

For example.

	2^x	x^2
$x=1$	2	1
$x=2$	4	4
$x=3$	8	9
$x=4$	16	16
$x=5$	32	25
$x=6$	64	36

$\frac{25}{16} = 1.56$

$\frac{36}{25} = 1.45$

$\frac{49}{36} = 1.36$

For every value of x after 4 L.R.H.S of the conclusion will be incremented twice but R.H.S will be incremented with the maximum value of 1.56. So, the conclusion $2^x > x^2$ is always true for the values of x after 4.

2) Prove if x is sum of the squares of four positive integer then $2^x \geq x^2$.

statement	Justification
1) $x = a^2 + b^2 + c^2 + d^2$ $a \geq 1, b \geq 1, c \geq 1, d \geq 1$	from the given statement
2) $x = 1^2 + 1^2 + 1^2 + 1^2$	given statement as (+)ve integer
3) $x = 4$	from ②
4) $x \geq 4$	from given statement and ②
5) $x \geq 4$ then $2^x \geq x^2$	from the statement $x \geq 4$. $2^x \geq x^2$

Reduction to definition (sets):

(i) 'S' is a finite set if there exist integer 'n' such that 'S' has 'n' elements. $S = \|n\|$

(ii) If 'S' and 'T' are both subsets of an universal set 'U' then 'T' is complement of 'S' if

(i) $S \cup T = U$

(ii) $S \cap T = \phi$

$U = \{1, 2, 3, 4, 5, 6, 7, 8, 9, 10\}$

$S = \{2, 3, 4\}$

$T = \{1, 5, 6, 7, 8, 9, 10\}$

3) Let 'S' be finite subset of infinite universal set 'U'. Let 'T' be the complement of 'S' with respect to 'U'. Prove that 'T' is infinite.

Statement	Justification
1. T is finite	negation of conclusion
2. $T = \|m\|$	from ①
3. S is finite $S = \|n\|$	from the hypothesis
4. T is complement of S $S \cup T = U$ $\|n+m\| = \|x\|$	from the hypothesis from definition
5. T is infinite	from ④ which rejects the hypothesis.

So, the conclusion T is infinite.

4) Show that the sum of the squares of two consecutive odd integers is even.

Statement	Justification
1) $S = x^2 + (x+2)^2$ $= x^2 + x^2 + 2x + 4$ $= 2x^2 + 2x + 4$	From given definition
2) $S = 2(x^2 + x + 2)$	From ①

Multiple of 2 will be even

Hence proved

OTHER FORMS OF PROOF:

(i) if then

(ii) if and only if (iff)

(i) If H then C i.e, if hypothesis H is true for some values then the conclusion C is true for same set of values.

Other ways of representing if then.

(i) $H \rightarrow C$

(ii) H only if C

(iii) C if H

(ii) if and only if:

This is represented as if A then B and if B then A.

Other ways of representing iff:

(i) $A \Leftrightarrow B$

(ii) A iff B

(iii) A exactly when B

Proof by sets:

Set: A group of characters or string forms a language which is represented as a set.

If 'E' and 'F' are two expressions representing a set then E = F means every element of 'E' is in 'F'

Properties:

(1) commutative law

(i) $A \cup B = B \cup A$

(ii) $A \cap B = B \cap A$

2) Associative law

(i) $A \cup (B \cup C) = (A \cup B) \cup C$

(ii) $A \cap (B \cap C) = (A \cap B) \cap C$

3) Distributive law

(i) $A \cup (B \cap C) = (A \cup B) \cap (A \cup C)$

(ii) $A \cap (B \cup C) = (A \cap B) \cup (A \cap C)$

Theorem:

statement:

Prove that the two sets generated in distributive law of union over intersection or equal.

proof:

Take two set E and F where $E = A \cup (B \cap C)$ and $F = (A \cup B) \cap (A \cup C)$

Take a element x in E & show that it is in F.

statement	Justification
1) x is in E	given hypothesis
2) x is in $A \cup (B \cap C)$	from ①
3) x is in A or x is in $(B \cap C)$	from ②

5) Let x be a real number then $\lfloor x \rfloor = \lceil x \rceil$ if and only if x is an integer

sol: This is of the form $\lfloor x \rfloor = \lceil x \rceil$ iff x is integer

This has to be proved by two if then statement

(i) if A then B (ii) if B then A.

31/7/24

(i) if $\lfloor x \rfloor = \lceil x \rceil$ then x is integer.

Statement	Justification
1) $\lfloor x \rfloor = \lceil x \rceil$	hypothesis
2) $\lfloor x \rfloor \leq x$	from definition of floor & given statement
3) $\lceil x \rceil \geq x$	from definition of ceil & given statement
4) $\lfloor x \rfloor \geq x$	from 1
5) $\lfloor x \rfloor \leq x \geq \lfloor x \rfloor$	from 2 and 4
6) x is an integer	from 5

(ii) if x is an integer then $\lfloor x \rfloor = \lceil x \rceil$

statement	Justification
1) x is an integer	given hypothesis
2) $\lfloor x \rfloor = x$	from definition & 1
3) $\lceil x \rceil = x$	from definition & 1
4) $\lfloor x \rfloor = \lceil x \rceil$	given conclusion.

ADDITIONAL FORMS OF PROOF:

(i) Proof by sets

(ii) Proof by contradiction (contra positive)

(iii) Proof by counter example

Proof by contradiction:

contradiction is negation of conclusion

i.e) contradiction of if H then C is if not C then not H.

Proof by counter Example:

To deside whether the given theorem statement is true or false. We have to prove that statement. If the theorem statement cannot be proved as true, then prove that it is false.

For Example: All Prime integers are odd.

since the above statement is not true. since 2 is even. It can refframed as All Prime integers are true except 2.

2/8/24

Theorem:

statement:

There is no pair of integers a and b such that $a \% b = b \% a$

Proof: For a pair of integers, three possible cases are

~~It can be written as $a \% b \neq b \% a$~~

(i) $a > b$

(ii) $a < b$

(iii) $a = b$

case - 1: $a < b$

$a \% b = a$ — ①

$b \% a = 0$ to $a-1$ — ②

From ① & ②

$a \% b \neq b \% a$

case - 2: $a > b$

$a \% b = 0$ to $b-1$ — ③

$b \% a = b$ — ④

From ③ & ④

$a \% b \neq b \% a$

case - 3: $a = b$

$a \% b = a \% a = 0$ — ⑤

$b \% a = a \% a = 0$ — ⑥

From ⑤ & ⑥

$a \% b = b \% a$

In case - 3, the theorem statement for no pair of integers a, b $a \% b = b \% a$ is disproved. Hence the theorem statement is reframed as

$a \% b = b \% a$ iff $a = b$

Prove by Proof of contradiction

(i) If $a \% b = b \% a$ then $a = b$

Statement	Justification
1) $a \neq b$	Negation of conclusion
2) $a < b$ $a \% b = a$ $b \% a = 0$ to $a-1$	From definition of module
3) $a > b$ $a \% b = 0$ to $b-1$ $b \% a = b$	From definition of module
4) $a \% b \neq b \% a$	From ② & ③

2) If $a = b$ then $a \% b = b \% a$

Statement	Justification
1) $a = b$	Given statement
2) $a \% b = 0$ $b \% a = 0$	From definition
3) $a \% b = b \% a$	From 2

PROOF BY INDUCTION:

* It is used to prove recursively defined objects.
* Mathematical induction proves the statement with integers but in automata theory (computation) it is used to prove the concept of trees and expression of varies types.

MATHEMATICAL INDUCTION:

Given a statement $s(n)$ about an integer n, the statement is proved by three cases.

(i) Basis : Prove $s(i)$ is true for an integer i where i takes the first minimal value.

(ii) Assumption: Assume $s(n)$ is true for all values of n.

(iii) Induction: Prove that $s(n+1)$ is true for an integer $n+1$

Problems:

i) Prove by induction $\underbrace{1+2+3+\dots+n}_{L.H.S} = \underbrace{\frac{n(n+1)}{2}}_{R.H.S}, \forall n>0$

(i) Basis:

$s(\overset{i}{n})$ $n=1$

$L.H.S = 1$

$R.H.S = \frac{1(1+1)}{2} = \frac{2}{2} = 1$

$L.H.S = R.H.S$

Hence Basis is proved.

(ii) Assumption:

Assume $s(n)$ is true for all values of n

i.e, $1+2+3+\dots+n = \frac{n(n+1)}{2}$ is true, $\forall n>0$

(iii) Induction:

Prove that the theorem statement is true for $s(n+1)$ is true.

L.H.S

$= 1+2+3+\dots n+(n+1)$

$= \frac{n(n+1)}{2} + (n+1)$ (by induction)

$= \frac{n(n+1)+2(n+1)}{2}$

$= \frac{(n+1)(n+2)}{2}$ — ①

R.H.S

$= \frac{n(n+1)}{2}$

$= \frac{(n+1)((n+1)+1)}{2}$

$= \frac{(n+1)(n+2)}{2}$ — ②

From ① & ②

L.H.S = R.H.S

Hence, induction is true, i.e, $1+2+3+\dots+n = \frac{n(n+1)}{2}$

---

2) Prove by induction $\sum_{i=1}^{n} i^2 = \frac{n(n+1)(2n+1)}{6}$

$1^2+2^2+\dots+n^2 = \frac{n(n+1)(2n+1)}{6}$

(i) Basis:

$s(n)$ $n=1$

L.H.S $= 1^2 = 1$ — ①

R.H.S $= \frac{1(1+1)(2(1)+1)}{6} = \frac{1(2)(3)}{6} = \frac{6}{6} = 1$ — ②

From ① & ②

L.H.S = R.H.S

Hence, Basis is proved.

(ii) Assumption:

Assume $s(n)$ is true for all values of n

i.e, $1^2+2^2+\dots+n^2 = \frac{n(n+1)(2n+1)}{6}$ is true

(iii) Induction:

Prove that the theorem statement is true for $s(n+1)$.

$$L.H.S = 1^2 + 2^2 + \dots + n^2 + (n+1)^2$$

$$= \frac{n(n+1)(2n+1)}{6} + (n+1)^2 \quad \text{(by Assumption)}$$

$$= \frac{n(n+1)(2n+1) + 6(n+1)^2}{6}$$

$$= \frac{(n^2+n)(2n+1) + 6(n^2+2n+1)}{6}$$

$$= \frac{2n^3 + n^2 + 2n^2 + n + 6n^2 + 12n + 6}{6}$$

$$= \frac{2n^3 + 9n^2 + 13n + 6}{6} \quad \text{— (1)}$$

$$R.H.S = \frac{(n+1)(n+1+1)(2(n+1)+1)}{6}$$

$$= \frac{(n+1)(n+2)(2n+3)}{6}$$

$$= \frac{(n^2 + 2n + n + 2)(2n+3)}{6}$$

$$= \frac{2n^3 + 4n^2 + 2n^2 + 4n + 3n^2 + 6n + 3n + 6}{6}$$

$$= \frac{2n^3 + 9n^2 + 13n + 6}{6} \quad \text{— (2)}$$

From (1) & (2)

$L.H.S = R.H.S$

Hence, the induction is true

(i.e), $1^2 + 2^2 + \dots + n^2 = \frac{n(n+1)(2n+1)}{6}$

5/8/24

3) Using mathematical induction prove that $5^n - 2^n$ is divisible by 3 $\forall n \geq 1$

Ans:

(i) Basis:

$S(n) = 1$

$L.H.S: 5^1 - 2^1 = 5 - 2 = 3$

$\therefore$ 3 is divisible by 3

$\therefore$ Hence Basis is proved.

(ii) Assumption :

Assume $5^n - 2^n$ is divisible by 3 is true for n.

(iii) Induction:

show that is true for $n+1$

$= 5^{(n+1)} - 2^{(n+1)}$

$= 5 \cdot 5^n - 2 \cdot 2^n$

$= 5 \cdot 5^n - 5 \cdot 2^n + 5 \cdot 2^n - 2 \cdot 2^n$

$= 5(5^n - 2^n) + 2^n(5-2)$

$= 5 \times$ multiple of $3 + 2^n \times 3$

Any two terms which are multiples of 3 will be definitely divisible by 3. Hence the proof $5^n - 2^n$ is divisible by 3 $\forall n \geq 1$

---

4) show that $4^{(2n+1)} + 3^{(n+2)}$ is a multiple of 13 $\forall n \geq 0$.

(i) Basis:

$S(n), n=0$

$L.H.S = 4^{(2(0)+1)} + 3^{(0+2)} = 4^1 + 3^2 = 4 + 9 = 13$

$\therefore$ 13 is a multiples of 13

$\therefore$ Hence basis is proved.

(ii) Assumption:

Assume $4^{(2n+1)} + 3^{(n+2)}$ is multiple of 13 is true for n.

(iii) Induction:

show that is true for $n+1$

$= 4^{(2(n+1)+1)} + 3^{(n+1+2)}$

$= 4^{2n+3} + 3^{n+3}$

$= 4^{2n} \cdot 4^3 + 3^n \cdot 3^3$

$= 4^{2n+1} \cdot 4^2 + 3^{n+2} \cdot 3$

$= 16 \cdot 4^{2n+1} + 16 \cdot 3^{n+2} - 16 \cdot 3^{n+2} + 3 \cdot 3^{n+2}$

$= 16(4^{2n+1} + 3^{n+2}) + 3^{n+2}(3-16)$

$= 16 \times$ multiple of $13 + 3^{n+2}(-13)$

$\therefore S(n+1)$ is True.

12/8/24

STRUCTURAL INDUCTION: (Mathematical Induction in Automata theory)

* In Automata theory there are several recursively defined structure or object which need to be proved.
* The recursively defined structure have basis case where one or more elimentary are defined and the inductive step where complex structure are defined.

Example.

Recursive definition of tree (Application of mathematical Induction):

Basis case:

A single node of a tree which is a root node is called as the elementary structure of the tree.

Induction:

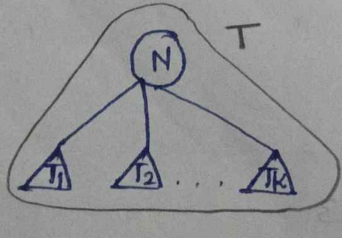
If $T_1, T_2, \ldots T_K$ are trees then a New tree (T) can be formed as follows.

step-1:

Make a new node (N) which is the root node of the newly formed tree (T).

step-2:

Append the existing trees T_1 to T_K with required edges to new node N.



show that the condition holds for the newly formed tree.

Theorem:

show that every tree has one node more than its edges i.e, $n = e+1$ where n is no. of nodes and e is no. of edges.

Basis:

The elementary structure is a single node.

○

Here, $n = 1$

$e = 0$

$\boxed{n = e+1}$ holds

∴ The condition is satisfied.

Assumption:

Assume that $n = e+1$ is true for all trees T_1 to T_K with $T_1 (n_1 = e_1 + 1)$, $T_2 (n_2 = e_2 + 1)$ . . . T_K as $n_K = e_K + 1$.

Induction:

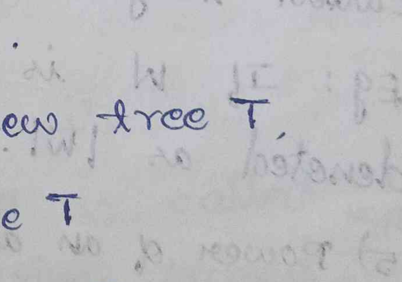
Introduce a new node N and append the existing trees with a new node N.

Let 'n' be the no. of nodes of new tree T.

'e' be the edges of new Tree T

Now, show that $n = e+1$ for tree T

$$\boxed{\begin{aligned} n &= n_1 + n_2 + \dots + n_K + 1 \\ e &= e_1 + e_2 + \dots + e_K + K \end{aligned}}$$



$$\begin{aligned} n &= n_1 + n_2 + \dots + n_K + 1 \\ &= (e_1+1) + (e_2+1) + \dots + (e_K+1) + 1 \\ &= e_1 + e_2 + \dots + e_K + K + 1 \end{aligned}$$

$\boxed{n = e+1}$

Central concepts of Automata Theory:

1) Alaphabets:

It is a finite set of symbols which is non-empty and is denoted as Σ.

Eg: $\Sigma = \{0, 1\}$, $\Sigma = \{a, b, c\}$

2) String:

It is a finite sequence of alphabets (symbols) choosen from Σ.

Eg: If $\Sigma = \{0, 1\}$ then the strings are $\{0, 1, 00, 01, 11, 000 \ldots\}$

3) Empty string:

A string with zero symbols is called empty string and is denoted as ϵ (epsilon)

4) Lenght of a string:

~~Number~~ Number of symbols in a string is called lenght of a string.

Eg: If W is a string then lenght of W is denoted as $|W|$.

13/8/24

5) Power of an alphabet:

If Σ is an alphabet, a set of all strings formed from the alphabet ~~which~~ is denoted as power of alphabet. It is represented as

If $\Sigma = \{0, 1\}$ then $\Sigma^0 = \{\} = \phi$, $\Sigma^1 = \{0, 1\}$,

$\Sigma^2 = \{00, 01, 10, 11\}$,

$\Sigma^3 = \{000, 001, 010, \ldots, 111\}$

6) Kleene closure:

Set of all strings over Σ including null strings forms Kleene closure and is denoted by $\Sigma^* = \Sigma^0 \cup \Sigma^1 \cup \Sigma^2 \cup \ldots$

If $\Sigma = \{a, b\}$, $\Sigma^* = \{\epsilon, a, b, aa, ab, ba, bb, aaa \ldots\}$

7) Positive closure:

Set of all string over sigma excluding null string forms positive closure and is denoted by $\Sigma^+ = \Sigma^1 \cup \Sigma^2 \cup \Sigma^3 \cup \ldots$

If $\Sigma = \{a, b\}$, $\Sigma^+ = \{a, b, aa, \ldots\}$

$\Sigma^* = \Sigma^+ \cup \epsilon$ (or) $\Sigma^+ + \epsilon$, $\Sigma^+ = \Sigma^* - \epsilon$

8) concatination of strings:

Let x, y be two strings, then the concatination of two strings is represented as xy.

If x = ababb, y = aabba

xy = ababbaabba

$|xy| = |x| + |y|$

9) Prefix and suffix:

Any number of strings formed with the leading symbols of the given string is called prefix of a string and those strings formed with the trailing symbols of the given string is called suffix of the string.

If a string W = 'kongu' then

prefix of W = {k, o, n, g, u, ko, on, ng, gu, kon, ong, ngu, kong, ongu, kongu ..}

suffix(W) = {u, g, n, o, k, ug, gn, no, ok, ugn, gno, nok, ugno, gnok, ugnok ..}

Note:

Prefix and suffix of the string except the original string can be a proper prefix or proper suffix.

For Example: hai

prefix = {h, ha, ~~hai~~}

suffix = {i, ai, ~~hai~~}

Types of Grammar:

Type	G	L	M
0	Phrase structured Grammar	Recursively Enumerable Language (REL)	Turing Machines (TM)
1	context sensitive Grammar (CSG)	context sensitive Language (CSL)	Lower Bound Automata (LBA)
2	context Free Grammar (CFG)	context Free Language (CFL)	Push Down Automata (PDA)
3	Regular Grammar (RG)	Regular Language (RL)	Finite Automata (FA)

Finite Automata:

A class of machines or models are designed to accept regular languages are called finite automata.

Finite automata contains (i) Finite no. of states

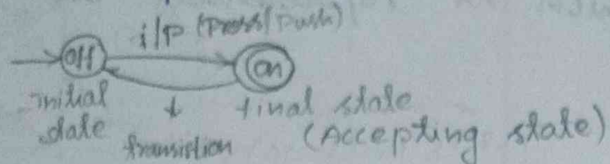
(ii) Finite no. of input symbols

(iii) Finite control block with a read write head.

(iv) Inputs tape which is divided into into no. of cells and each cell can hold atmost one symbol.

The finite automata model in general has a set of state, input symbols, Transistions, Initial and final state.

Eg: A simple switch



Types of Finite Automata:

1) Deterministic Finite Automata (DFA)
2) Non-Deterministic Finite Automata (NFA)

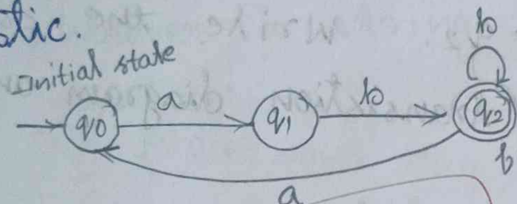
Applications of FA:

* ~~In~~ Designing an simple logic circuit
* Lexicil Analysis in complier (token separator)

Deterministic Finite Automata:

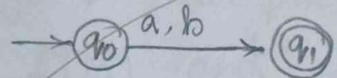
In an automata, for each input symbol there is only one transistion from a state, then it is deterministic.

Eg: (i)



on reading a, it goes to only one state i.e, q_1

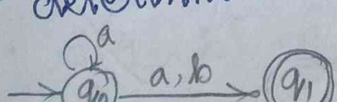
(ii)



Non-Deterministic Finite Automata:

In an automata, for each input symbol if there is more than one transistion from a state then it is Non-deterministic.

Eg: (i)



on reading a, it goes to two states i.e, q_0 + q_1

DFA:

Deterministic Finite automata model is represented as M with 5 tuples.

i.e, $M = \{Q, \Sigma, \delta, q_0, F\}$,

Where Q is set of all states of the model.

Σ is set of input symbols

δ is transistion which is represented as

$$Q \times \Sigma \to Q$$

q_0 is Initial state

F is Final or Accepting state of the model.

Transistion Diagram:

It is a directed graph whose vertices are states of DFA and edges are transistion from one state to another.

14/8/24

Problem:

1) If $m = (\{q_0, q_1, q_2\}, \{0,1\}, \delta, q_0, \{q_2\})$ and δ is $\delta(q_0, 0) = q_1$, $\delta(q_1, 0) = q_2$, $\delta(q_2, 1) = q_1$, $\delta(q_2, 0) = q_2$. Write the components of DFA and draw the transistion diagram and transistion table.

Sol:

components of DFA:

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

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

δ is $\delta(q_0, 0) = q_1$, $\delta(q_1, 0) = q_2$, $\delta(q_2, 1) = q_1$, $\delta(q_2, 0) = q_2$

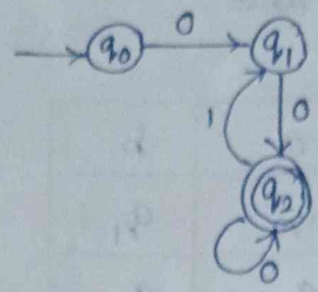
Initial state is q_0

Final state $(F) = \{q_2\}$

Transistion table:

states / I/P	0	1
→ q_0	q_1	–
q_1	q_2	–
* q_2	q_2	q_1

Transistion Diagram:



$L = \{00, 000, 0010, 0000, \ldots 001010 \ldots\}$

The set of all strings that starts with two zero will be accepted by this model.

Note:

A string W is accepted by automata 'M'

If $\hat{\delta}(q_0, W) = P$ Where $P \subseteq F$.

$\hat{\delta}$ is called as extended transistion function which is used whenever a string is to be processed instead of symbol.

$\delta(q_0, 0) = q_1$

$\hat{\delta}(q_0, 00) = q_1$

The language L accepted by automata M is denoted as L(M) and is defined as

$L(M) = \{W \mid \hat{\delta}(q_0, W) \in F\}$

2) If $M = (\{q_0, q_1, q_2\}, \{a, b\}, \delta, q_0, \{q_2\})$ and δ is $\delta(q_0, a) = q_0$, $\delta(q_0, b) = q_1$, $\delta(q_1, a) = q_2$, $\delta(q_1, b) = q_1$, $\delta(q_2, a) = q_2$, $\delta(q_2, b) = q_0$.

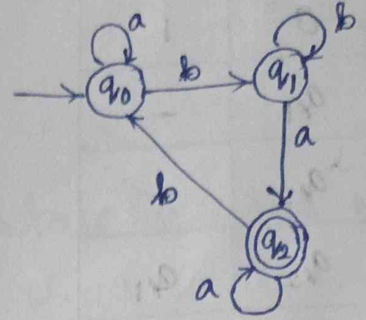
Draw the Transistion table, transistion diagram and find the language.

sol:

Transision Table

states / I/P	a	b
$\rightarrow q_0$	q_0	q_1
q_1	q_2	q_1
$*q_2$	q_2	q_0

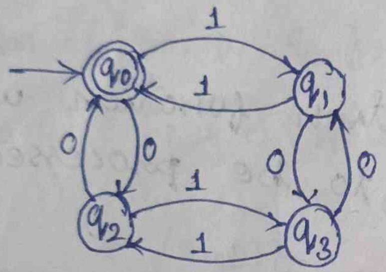
Transision Diagram:



$L = \{aba, ba, bba, abba, baa\}$

The set of all strings that ends with 'a' will be accepted by this model.

3) Find the language accepted by the given DFA.



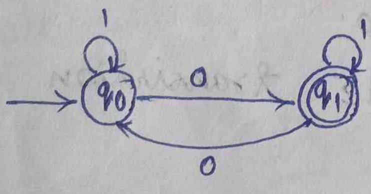
sol: $L = \{00, 11, 1001, 11001100, 1111, 0000, \ldots\}$

set of all strings with even no. of 0's or 1's will be accepted by this model.

4) Design a DFA for a language that accepts only odd no. of 0's over $\{0, 1\}$

sol:

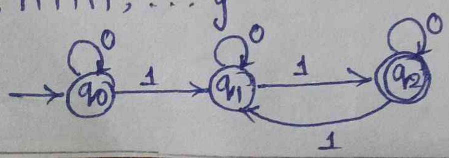
$L = \{0, 000, 00000, \ldots\}$



5) Design a DFA for a language that accepts only even no. of 1's over $\{0, 1\}$

sol:

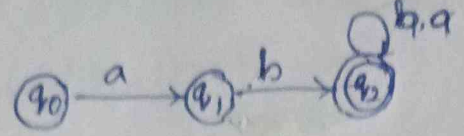
$L = \{11, 1111, 111111, \ldots\}$



6) Design a DFA that accepts strings that starts with ab over {a, b}.

sol:

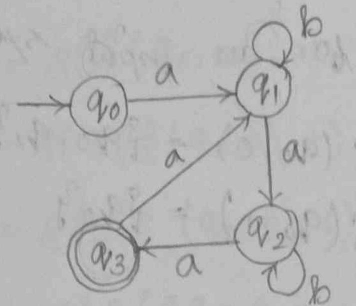
$L = \{ab, abb, abab, \dots\}$



7) Design a DFA for the language that accepts set of strings where no. of a's in the string is divisible by 3 over {a, b}.

sol:

$L = \{aaa, aaaaaa, abaa, aaba, \dots\}$



19/8/24

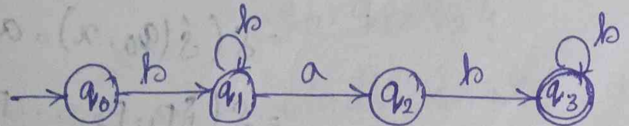
1) Design a DFA for a language $L = \{b^m a b^n \mid m, n \geq 1\}$

2) Design a DFA $L = \{w / w$ is of even length and begins with 01$\}$

3) Design a DFA for a language $L = \{a^m b^n \mid m$ and n are positive integers$\}$

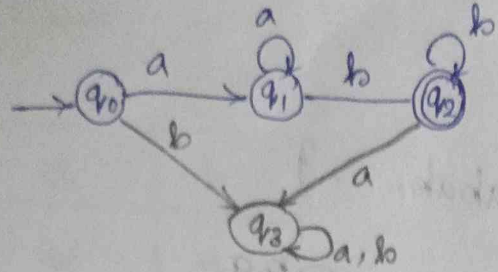
sol:

1) $L = \{bab, bbabb, bbbabbb \dots\}$



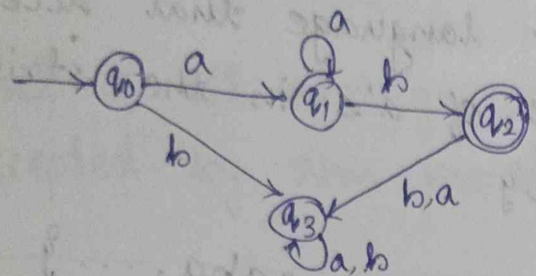
2) $L = \{01, 0101, 0111, \dots\}$

3) $L = \{ab, aab, aabb, abb, \ldots\}$



4) $L = \{a^n b \mid n \geq 1\}$

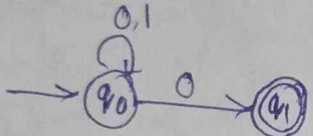
$L = \{ab, aab, aaab, \ldots\}$



NFA:

More than one transition for an input symbol from a state

Eg:



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

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

Language of NFA:

Language of NFA is defined as $L(M)$ and is denoted as $L(M) = \{w \mid \hat{\delta}(q_0, w) \in Q,\ Q \cap F \neq \phi\}$

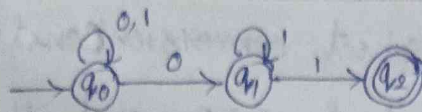
Extended transition Function ($\hat{\delta}$):

$\hat{\delta}$ is used whenever string is to be processed.

If a string $w = xa$ then

$$\hat{\delta}(q_0, w) = \hat{\delta}(q_0, xa)$$
$$= \delta(\hat{\delta}(q_0, x), a)$$
$$= \delta(\{P_1, P_2, \ldots P_n\}, a)$$
$$= \delta(P_1, a) \cup \delta(P_2, a) \cup \ldots \cup \delta(P_n, a)$$
$$= \{R_1, R_2, \ldots R_n\} \cap F \neq \phi$$

Example:



(i) check whether 11 is accepted.

$$\hat{\delta}(q_0, w) = \hat{\delta}(q_0, 11)$$
$$= \delta(\hat{\delta}(q_0, 1), 1)$$
$$= \delta(\delta(q_0, 1), 1)$$
$$= \delta(\{q_0\}, 1)$$
$$= \delta(q_0, 1)$$
$$= \{q_0\} \text{ is not a final state}$$

$\therefore$ 11 is rejected.

(ii) check whether 011 is accepted.

$$\hat{\delta}(q_0, w) = \hat{\delta}(q_0, 011)$$
$$= \delta(\hat{\delta}(q_0, 01), 1)$$
$$= \delta(\delta(\hat{\delta}(q_0, 0), 1), 1)$$
$$= \delta(\delta(\delta(q_0, 0), 1), 1)$$
$$= \delta(\delta(\{q_0, q_1\}, 1), 1)$$
$$= \delta(\{\delta(q_0, 1) \cup \delta(q_1, 1)\}, 1)$$
$$= \delta(\{q_0\} \cup \{q_1, q_2\}, 1)$$
$$= \delta(\{q_0, q_1, q_2\}, 1)$$
$$= \delta(q_0, 1) \cup \delta(q_1, 1) \cup \delta(q_2, 1)$$
$$= \{q_0\} \cup \{q_1, q_2\} \cup \{\}$$
$$= \{q_0, q_1, q_2\}$$

$\therefore$ q_2 is a final state

$\therefore$ 011 is accepted.

Conversion of NFA to DFA: (subset ~~conversion~~ construction)

Every NFA has an equivalent DFA and the class of languages accepted by NFA includes the same class of language accepted by DFA. This can be constructed by following rules.

Let NFA $N = \{Q_N, \Sigma, \delta_N, q_0, F_N\}$ and

DFA $D = \{Q_D, \Sigma, \delta_D, q_0, F_D\}$

Where Q_D is set of subsets of Q_N and

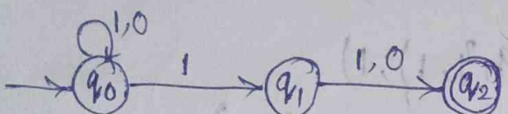
Q_D is the power set of Q_N

If Q_N has n states then Q_D has 2^n states

F_D is subset of Q_N such that $S \cap F_N \neq \phi$

$$\boxed{\delta_D(S, a) = \bigcup_{i \in S}^{n} \delta_N(p_i, a)}$$

i) construct the equivalent DFA for the given NFA.



Initial state of NFA $= q_0 =$ Initial state of DFA

$\downarrow$

A

$\delta_D(q_0, 0) = \delta_N(q_0, 0) = \{q_0\}$ - A

$\delta_D(q_0, 1) = \delta_N(q_0, 1) = \{q_0 q_1\}$ - B

$\delta_D(\{q_0 q_1\}, 0) = \delta_N(\{q_0 q_1\}, 0) = \delta_N(q_0, 0) \cup \delta_N(q_1, 0)$

$= \{q_0\} \cup \{q_2\} = \{q_0 q_2\}$ - C

$\delta_D(\{q_0 q_1\}, 1) = \delta_N(\{q_0 q_1\}, 1) = \delta_N(q_0, 1) \cup \delta_N(q_1, 1)$

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

$= \{q_0 q_1 q_2\}$ - D

$\delta_D(\{q_0 q_2\}, 0) = \delta_N(\{q_0 q_2\}, 0) = \delta_N(q_0, 0) \cup \delta_N(q_2, 0)$

$= \{q_0\} \cup \{\}$

$= \{q_0\}$ - A

$\delta_D(\{q_0 q_2\}, 1) = \delta_N(\{q_0 q_2\}, 1) = \delta_N(q_0, 1) \cup \delta_N(q_2, 1)$

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

$= \{q_0 q_1\} - B$

$\delta_D(\{q_0 q_1 q_2\}, 0) = \delta_N(\{q_0 q_1 q_2\}, 0) = \delta_N(q_0, 0) \cup \delta_N(q_1, 0) \cup \delta_N(q_2, 0)$

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

$= \{q_0 q_2\} - C$

$\delta_D(\{q_0 q_1 q_2\}, 1) = \delta_N(\{q_0 q_1 q_2\}, 1)$

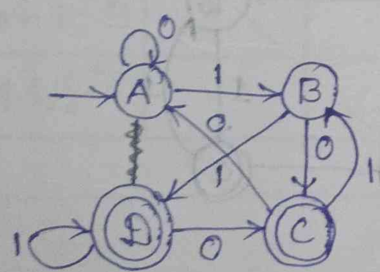
$= \delta_N(q_0, 1) \cup \delta_N(q_1, 1) \cup \delta_N(q_2, 1)$

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

$= \{q_0 q_1 q_2\} - D$

States	I/P symbols	
	0	1
$\rightarrow \{q_0\} - A$	A	B
$\{q_0 q_1\} - B$	C	D
$^* \{q_0 q_2\} - C$	A	B
$^* \{q_0 q_1 q_2\} - D$	C	D

Which ever states of DFA contains the states of NFA then there will be the final state.

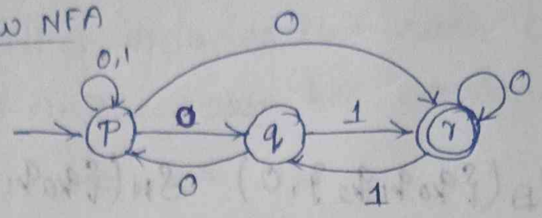


22/8/24

Q) construct the DFA for the given NFA

state	i/P	
	0	1
$\rightarrow p$	$\{p,q\}$	$\{p\}$
q	$\{p\}$	$\{r\}$
$*r$	$\{p,r\}$	$\{q\}$

Draw NFA



Initial state of NFA = DFA = $\{p\} \rightarrow A$

$\delta_D(p,0) = \delta_N(p,0) = \{pq\} \rightarrow B$

$\delta_D(p,1) = \delta_N(p,1) = \{p\} \rightarrow A$

$\delta_D(\{pq\},0) = \delta_N(\{pq\},0) = \delta_N(p,0) \cup \delta_N(q,0) = \{pq\} \cup \{p\} = \{pq\} \rightarrow B$

$\delta_D(\{pq\},1) = \delta_N(\{pq\},1) = \delta_N(p,1) \cup \delta_N(q,1) = \{p\} \cup \{r\} = \{pr\} \rightarrow C$

$\delta_D(\{pr\},0) = \delta_N(\{pr\},0) = \delta_N(p,0) \cup \delta_N(r,0) = \{pq\} \cup \{pr\} = \{pqr\} \rightarrow D$

$\delta_D(\{pr\},1) = \delta_N(\{pr\},1) = \delta_N(p,1) \cup \delta_N(r,1) = \{p\} \cup \{q\} = \{pq\} \rightarrow B$

$\delta_D(\{pqr\},0) = \delta_N(\{pqr\},0) = \delta_N(p,0) \cup \delta_N(q,0) \cup \delta_N(r,0)$

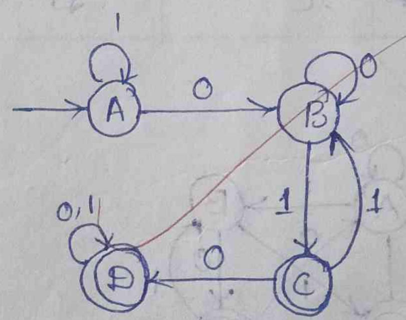
$= \{pq\} \cup \{p\} \cup \{pr\} = \{pqr\} \rightarrow D$

$\delta_D(\{pqr\},1) = \delta_N(\{pqr\},1) = \delta_N(p,1) \cup \delta_N(q,1) \cup \delta_N(r,1)$

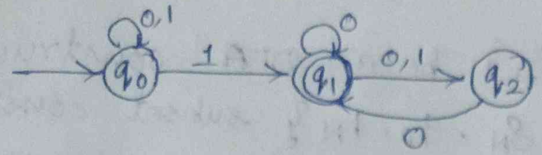
$= \{p\} \cup \{r\} \cup \{q\} = \{pqr\} \rightarrow D$

DFA

states	i/P	
	0	1
$\rightarrow \{p\}$ - A	B	A
$\{pq\}$ - B	B	C
$\{pr\}$ - $*C$	D	B
$\{pqr\}$ - $*D$	D	D



3) convert the NFA to DFA



Initial state of NFA = DFA = $\{q_0\}$

$\delta_D(q_0, 0) = \delta_N(q_0, 0) = \{q_0\} \rightarrow A$

$\delta_D(q_0, 1) = \delta_N(q_0, 1) = \{q_0 q_1\} \rightarrow B$

$\delta_D(\{q_0 q_1\}, 0) = \delta_N(\{q_0 q_1\}, 0) = \delta_N(q_0, 0) \cup \delta_N(q_1, 0)$

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

$\delta_D(\{q_0 q_1\}, 1) = \delta_N(\{q_0 q_1\}, 1) = \delta_N(q_0, 1) \cup \delta_N(q_1, 1)$

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

$\delta_D(\{q_0 q_1 q_2\}, 0) = \delta_N(\{q_0 q_1 q_2\}, 0) = \delta_N(q_0, 0) \cup \delta_N(q_1, 0) \cup \delta_N(q_2, 0)$

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

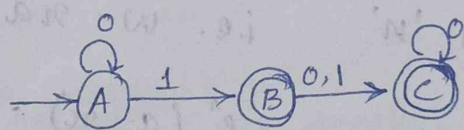
$= \{q_0 q_1 q_2\} \rightarrow C$

$\delta_D(\{q_0 q_1 q_2\}, 1) = \delta_N(\{q_0 q_1 q_2\}, 1) = \delta_N(q_0, 1) \cup \delta_N(q_1, 1) \cup \delta_N(q_2, 1)$

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

$= \{q_0 q_1 q_2\} \rightarrow C$

states	i/P	
	0	1
$\rightarrow \{q_0\}$ - A	A	B
$\{q_0 q_1\}$ - *B	C	C
$\{q_0 q_1 q_2\}$ - *C	C	C



23/8/24

Theorem:

If DFA $D = \{Q_D, \Sigma, \delta_D, q_0, F_D\}$ be a DFA constructed from an NFA $N = \{Q_N, \Sigma, \delta_N, q_0, F_N\}$ subset construction method then $L(D) = L(N)$. → set of all strings accepted in NFA

↑ set of all strings accepted in DFA

The theorem is proved by mathematical induction.

(i) Basis

$w = \epsilon$

$\delta_D(q_0, \epsilon) = q_0$

$\delta_N(q_0, \epsilon) = q_0$

$\delta_D(q_0, \epsilon) = \delta_N(q_0, \epsilon)$

so the basis case is true.

(ii) ~~Induction~~ Assumption

The theorem has ~~been~~ to be proved by induction on length 'w' such that $\boxed{\hat{\delta_D}(q_0, w) = \hat{\delta_N}(q_0, w)}$

Let $w = xa$

$|x| = w < |w| = n+1$

Assume, the statement is true for the string of length 'n' i.e. $w = xa$

$$\hat{\delta_D}(q_0, x) = \hat{\delta_N}(q_0, x)$$

(iii) Induction

Now, prove for $|w| = xa = n+1$

$$\hat{\delta_D}(q_0, w) = \hat{\delta_N}(q_0, w)$$

L.H.S → $\hat{\delta_D}(q_0, w) = \hat{\delta_D}(q_0, xa)$

$= \delta_D(\hat{\delta_D}(q_0, x), a)$

$= \delta_D(\hat{\delta_N}(q_0, x), a)$

$= \delta_D(\{P_1, P_2, \ldots P_n\}, a)$

$= \delta_D(P_1, a) \cup \delta_D(P_2, a) \cup \ldots \delta_D(P_n, a)$

$= \delta_D(S, a)$

$= \bigcup_{i=1}^{n} \delta_N(P_i, a)$ — ①

R.H.S $\rightarrow \hat{\delta_N}(q_0, w) = \hat{\delta_N}(q_0, xa)$

$= \delta_N(\hat{\delta_N}(q_0, x), a)$

$= \delta_N(\{P_1, P_2, \ldots P_n\}, a)$

$= \delta_N(P_1, a) \cup \delta_N(P_2, a) \ldots \cup \delta_N(P_n, a)$

$= \bigcup_{i=1}^{n} \delta_N(P_i, a)$ — ②

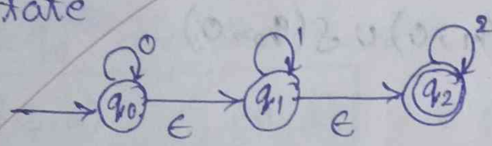
From ① and ②

$\hat{\delta_D}(q_0, w) = \hat{\delta_N}(q_0, w)$ which is $L(D) = L(N)$

∴ Hence proved.

NFA with ε-moves

NFA accepts the string 'w' if there is some path labeled with epsilon (ε) from initial state to final state



q_0 moves to q_1 without reading any symbol.

ε-closure(q)

It denotes the set of all states $\{P_1, P_2, \ldots P_n\}$ such that there is a path from state q to state P with label ε

ε-closure(q_0) = $\{q_0, q_1, q_2\}$ only with continuous ε we should write

ε-closure(q_1) = $\{q_1, q_2\}$

ε-closure(q_2) = $\{q_2\}$

Note:

If P is set of states then ε-closure(P) is $\bigcup_{q \in P}$ ε-closure(q)

$$\boxed{\hat{\delta}(q_0, \epsilon) = \epsilon\text{-closure}(q_0)}$$

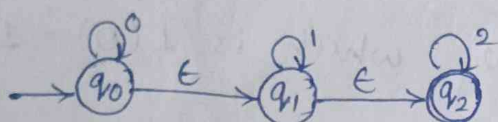
27/8/24

ε-NFA

For the given NFA, find the $\hat{\delta}$ for the following.

(i) $\hat{\delta}(q_0, \epsilon)$ (iii) $\hat{\delta}(q_1, 2)$

(ii) $\hat{\delta}(q_0, 0)$ (iv) $\hat{\delta}(q_0, 12)$



(i) $\hat{\delta}(q_0, \epsilon) = \epsilon\text{-closure}(q_0)$

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

(ii) $\hat{\delta}(q_0, 0) = \delta(\hat{\delta}(q_0, \epsilon), 0)$

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

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

$= \{q_0\} \cup \phi \cup \phi$

$= \epsilon\text{-closure}(q_0)$

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

(iii) $\hat{\delta}(q_1, 2) = \delta(\hat{\delta}(q_1, \epsilon), 2)$

$= \delta(\epsilon\text{-closure}(q_1), 2)$

$= \delta(\{q_1, q_2\}, 2)$

$= \delta(q_1, 2) \cup \delta(q_2, 2)$

$= \phi \cup \{q_2\}$

$= \epsilon\text{-closure}(q_2)$

$= \{q_2\}$

(iv) $\hat{\delta}(q_0, 12) = \delta(\hat{\delta}(q_0, 1), 2)$

$= \delta(\delta(\hat{\delta}(q_0, \epsilon), 1), 2)$

$= \delta(\delta(\{q_0, q_1, q_2\}, 1), 2)$

$= \delta(\delta(q_0, 1) \cup \delta(q_1, 1) \cup \delta(q_2, 1), 2)$

$= \cancel{\delta(\{\} \cup \{q_1\} \cup \{\}, 2)}$ $\delta(\epsilon\text{-closure}\{\phi \cup q_1 \cup \phi\}, 2\}$

$= \cancel{\delta(\{q_1\}, 2)}$ $\delta(\epsilon\text{-closure}(q_1), 2)$

$= \delta(\{q_1, q_2\}, 2)$

$= \delta(q_1, 2) \cup \delta(q_2, 2)$

$= \epsilon\text{-closure}(\phi \cup q_2)$

$= \{q_2\}$

---

conversion of ε-NFA to NFA without an ε-moves.

Let $E = \{Q_E, \Sigma, \delta_E, q_0, F_E\}$ be an NFA with ε-transitions and $N = \{Q_N, \Sigma, \delta_N, q_0, F_N\}$ be an NFA without ε transitions.

Where $Q_E = Q_N$

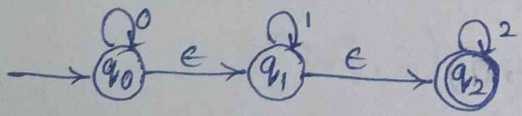
Σ & q_0 are same for E & N

$$\boxed{\delta_N(q, a) = \hat{\delta}_E(q, a)}$$

$$F_N = \begin{cases} F_E & \text{if } \epsilon\text{-closure}(q_0) \text{ does not contain the final state of } E \\ F_E \cup \{q_0\} & \text{if } \epsilon\text{-closure}(q_0) \text{ contains the final state of } E \end{cases}$$

10m

1) convert the given ε-NFA to NFA



sol:

$\epsilon\text{-closure}(q_0) = \{q_0, q_1, q_2\}$
↓
it contains the final state of E

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

$\Sigma = \{0, 1, 2\}$

$q_0 = \{q_0\}$

$F_N = F_E \cup \{q_0\} = \{q_0, q_2\}$

$\delta_N(q_0, 0) = \hat{\delta_E}(q_0, 0)$

$= \delta(\hat{\delta_E}(q_0, \epsilon), 0) = \delta(\epsilon\text{-closure}(q_0), 0)$

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

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

$= \epsilon\text{-closure}(q_0 \cup \phi \cup \phi)$

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

$\delta_N(q_0, 1) = \hat{\delta_E}(q_0, 1)$

$= \delta(\hat{\delta_E}(q_0, \epsilon), 1) = \delta(\epsilon\text{-closure}(q_0), 1)$

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

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

$= \epsilon\text{-closure}(\phi \cup q_1 \cup \phi)$

$= \{q_1, q_2\}$

$\delta_N(q_0, 2) = \hat{\delta_E}(q_0, 2)$

$= \delta(\hat{\delta_E}(q_0, \epsilon), 2)$

$= \delta(\{q_0, q_1, q_2\}, 2)$

$= \delta(q_0, 2) \cup \delta(q_1, 2) \cup \delta(q_2, 2)$

$= \epsilon\text{-closure}(\phi \cup \phi \cup q_2)$

$= \{q_2\}$

$\delta_N(q_1, 0) = \hat{\delta_E}(q_1, 0)$

$= \delta(\hat{\delta_E}(q_1, \epsilon), 0) = \delta(\epsilon\text{-closure}(q_1), 0)$

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

$= \delta(q_1, 0) \cup \delta(q_2, 0)$

$= \epsilon\text{-closure}(\phi \cup \phi)$

$= \{\}$

$\delta_N(q_1, 1) = \hat{\delta_E}(q_1, \cancel{0}1)$

$= \delta(\hat{\delta_E}(q_1, \epsilon), 1)$

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

$= \delta(q_1, 1) \cup \delta(q_2, 1)$

$= \epsilon\text{-closure}(q_1 \cup \phi)$

$= \{q_1, q_2\}$

$\delta_N(q_1, 2) = \hat{\delta_E}(q_1, 2)$

$= \delta(\hat{\delta_E}(q_1, \epsilon), 2)$

$= \delta(\{q_1, q_2\}, 2)$

$= \delta(q_1, 2) \cup \delta(q_2, 2)$

$= \epsilon\text{-closure}(\phi \cup q_2)$

$= \{q_2\}$

$\delta_N(q_2, 0) = \hat{\delta_E}(q_2, 0)$

$= \delta(\hat{\delta_E}(q_2, \epsilon), 0) = \delta(\epsilon\text{-closure}(q_2), 0)$

$= \delta(\{q_2\}, 0)$

$= \delta(q_2, 0)$

$= \epsilon\text{-closure}(\phi) = \{\}$

$\delta_N(q_2, 1) = \hat{\delta_E}(q_2, 1)$

$= \delta(\hat{\delta_E}(q_2, \epsilon), 1)$

$= \delta(\{q_2\}, 1) = \delta(q_2, 1)$

$= \epsilon\text{-closure}(\phi) = \{\}$

$\delta_N(q_2, 2) = \hat{\delta_E}(q_2, 2)$

$= \delta(\hat{\delta_E}(q_2, \epsilon), 2)$

$= \delta(\{q_2\}, 2) = \delta(q_2, 2)$

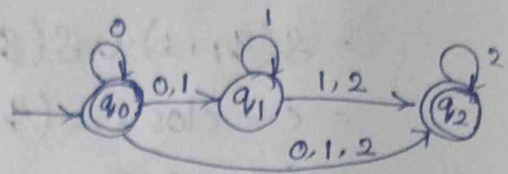
$= \epsilon\text{-closure}(q_2)$

$\{\epsilon\} = \{q_2\}$

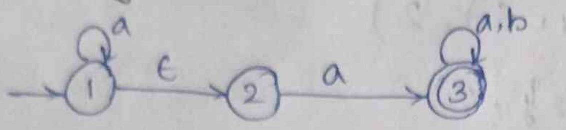
NFA

S/T

δ	0	1	2
$\rightarrow q_0$	$\{q_0, q_1, q_2\}$	$\{q_1, q_2\}$	$\{q_2\}$
q_1	ϕ	$\{q_1, q_2\}$	$\{q_2\}$
$^*q_2$	ϕ	ϕ	$\{q_2\}$



2) convert the given ε-NFA to NFA



sol:

$Q_N = Q_E = \{1, 2, 3\}$

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

$q_0 = \{1\}$

$F_N = \{3\}$

ε-closure(1) = {1, 2}

does not contain the final state q_E

$\delta_N(1, a) = \hat{\delta_E}(1, a)$

$= \delta(\hat{\delta_E}(1, \epsilon), a)$

$= \delta(\epsilon\text{-closure}(1), a)$

$= \delta(\{1, 2\}, a)$

$= \delta(1, a) \cup \delta(2, a) = \{1, 3\}$

$= \epsilon\text{-closure}(1, 3) = \epsilon\text{-closure}(1) \cup \epsilon\text{-closure}(3)$

$= \{1, 2\} \cup \{3\}$

$= \{1, 2, 3\}$

$\delta_N(1, b) = \hat{\delta_E}(1, b) = \delta(\hat{\delta_E}(1, \epsilon), b) = \delta(\{1, 2\}, b)$

$= \delta(1, b) \cup \delta(2, b)$

$= \epsilon\text{-closure}(\phi \cup \phi$

$= \phi$

$\delta_N(2, a) = \hat{\delta_E}(2, a) = \delta(\hat{\delta_E}(2, \epsilon), a) = \delta(\epsilon\text{-closure}(2), a)$

$= \delta(\{2, 3\}, a) = \epsilon\text{-closure}(3) = \{3\}$

$\delta_N(2, b) = \hat{\delta_E}(2, b) = \delta(\hat{\delta_E}(2, \epsilon), b) = \delta(2, b)$

$= \epsilon\text{-closure}(\phi) = \phi$

$\delta_N(3,a) = \hat{\delta_E}(3,a) = \delta(\hat{\delta_E}(3,\epsilon),a) = \delta(\epsilon\text{-closure}(3),a)$

$= \delta(\{3\},a) = \epsilon\text{-closure}(3) = \{3\}$

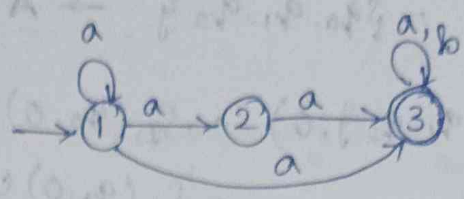
$\delta_N(3,b) = \hat{\delta_E}(3,b) = \delta(\hat{\delta_E}(3,\epsilon),b) = \delta(3,b)$

$= \epsilon\text{-closure}(3) = \{3\}$

NFA

i/p

δ	a	b
$\rightarrow 1$	$\{1,2,3\}$	ϕ
2	$\{3\}$	ϕ
$*3$	$\{3\}$	$\{3\}$



ϵ-NFA to DFA

Given any ϵ-NFA E, DFA D can be constructed from E that accepts the same language as E. If $E = \{Q_E, \Sigma, \delta_E, q_0, F_E\}$ then $D = \{Q_D, \Sigma, \delta_D, q_D, F_D\}$

Where Q_D is the subset of Q_E

Σ is same for E and D

q_D is ϵ-closure(q_0) → Initial state of ϵ-NFA

F_D is set of final states in DFA that contains atleast one of the states in F_E

δ_D is defined as $\delta_D(S,a)$ is

(i) Let $S = \{P_1, P_2, \ldots P_n\}$

(ii) compute $\bigcup_{i=1}^{n} \delta_D(P_i, a) = \{r_1, r_2, \ldots r_K\}$

(iii) $\delta_D(S,a) = \epsilon\text{-closure}(\{r_1, r_2, \ldots r_K\})$

$= \bigcup_{i=1}^{K} \epsilon\text{-closure}(r_i)$

1) convert the given ϵ-NFA to DFA

sol:

Initial state of DFA

$q_D = \epsilon\text{-closure}(q_0)$

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

$\delta_D(\{q_0, q_1, q_2\}, 0) = \delta_D(q_0, 0) \cup \delta_D(q_1, 0) \cup \delta_D(q_2, 0)$

$= \delta_E(q_0, 0) \cup \delta_E(q_1, 0) \cup \delta_E(q_2, 0)$

$= q_0 \cup \phi \cup \phi$

$= \epsilon\text{-closure}(q_0)$

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

$\delta_D(\{q_0, q_1, q_2\}, 1) = \delta_E(q_0, 1) \cup \delta_E(q_1, 1) \cup \delta_E(q_2, 1)$

$= \phi \cup q_1 \cup \phi$

$= \epsilon\text{-closure}(q_1)$

$= \{q_1, q_2\} \rightarrow B$

$\delta_D(\{q_0, q_1, q_2\}, 2) = \delta_E(q_0, 2) \cup \delta_E(q_1, 2) \cup \delta_E(q_2, 2)$

$= \phi \cup \phi \cup q_2$

$= \epsilon\text{-closure}(q_2)$

$= \{q_2\} \rightarrow C$

$\delta_D(\{q_1, q_2\}, 0) = \delta_E(q_1, 0) \cup \delta_E(q_2, 0)$

$= \phi \cup \phi = \epsilon\text{-closure}(\phi)$

$= \phi$

$\delta_D(\{q_1, q_2\}, 1) = \delta_E(q_1, 1) \cup \delta_E(q_2, 1)$

$= q_1 \cup \phi$

$= \epsilon\text{-closure}(q_1)$

$= \{q_1, q_2\} \rightarrow B$

$\delta_D(\{q_1, q_2\}, 2) = \delta_E(q_1, 2) \cup \delta_E(q_2, 2)$

$= \phi \cup q_2$

$= \epsilon\text{-closure}(q_2)$

$= \{q_2\} \rightarrow C$

$\delta_D(\{q_2\}, 0) = \delta_E(q_2, 0) = \epsilon\text{-closure}(\phi)$

$= \phi$

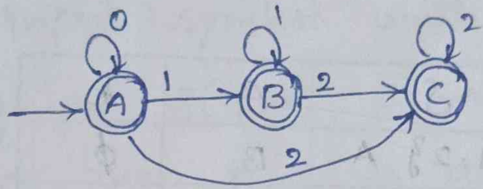
$\delta_D(\{q_2\}, 1) = \delta_E(q_2, 1) = \epsilon\text{-closure}(\phi)$

$= \phi$

$\delta_D(\{q_2\}, 2) = \delta_E(q_2, 2) = \epsilon\text{-closure}(q_2)$

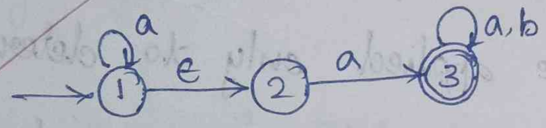
$= \{q_2\} \rightarrow C$

δ	0	1	2
$\{q_0, q_1, q_2\}$ A	A	B	C
$\{q_1, q_2\}$ B	ϕ	B	C
$\{q_2\}$ C	ϕ	ϕ	C



∴ since all states (A, B, C) contain q_2, (all are final state)

2) convert the given ϵ-NFA to DFA



Initial state of DFA

$q_D = \epsilon\text{-closure}(1)$

$= \{1, 2\} \rightarrow A$

$\delta_D(\{1, 2\}, a) = \delta_E(1, a) \cup \delta_E(2, a)$

$= 1 \cup 3 = \{1, 3\}$

$= \epsilon\text{-closure}(1) \cup \epsilon\text{-closure}(3)$

$= \{1, 2\} \cup \{3\}$

$= \{1, 2, 3\} \rightarrow B$

$\delta_D(\{1,2\}, b) = \delta_E(1, b) \cup \delta_E(2, b)$

$= \phi \cup \phi = \epsilon\text{-closure}(\phi)$

$= \phi$

$\delta_D(\{1,2,3\}, a) = \delta_E(1,a) \cup \delta_E(2,a) \cup \delta_E(3,a)$

$= 1 \cup 3 \cup 3 = \{1,3\}$

$= \epsilon\text{-closure}(1) \cup \epsilon\text{-closure}(3)$

$= \{1,2\} \cup \{3\} = \{1,2,3\} \rightarrow B$

$\delta_D(\{1,2,3\}, b) = \delta_E(1,b) \cup \delta_E(2,b) \cup \delta_E(3,b)$

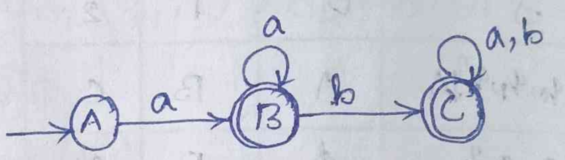
$= \phi \cup \phi \cup 3$

$= \epsilon\text{-closure}(3) = \{3\} \rightarrow C$

$\delta(\{3\}, a) = \delta_E(3,a) = \epsilon\text{-closure}(3) = \{3\} \rightarrow C$

$\delta_D(\{3\}, b) = \delta_E(3,b) = \epsilon\text{-closure}(3) = \{3\} \rightarrow C$

δ	a	b
$\{1,2\}$ A	B	ϕ
$\{1,2,3\}$ B	B	C
$\{3\}$ C	C	C



29/8/24

Equivalence & Minimization of DFA

Minimization can be applied only to deterministic finite Automata (DFA)

step-1: Find the states that are equivalent and replace the states with the single state which minimizes the automata.

step-2: Two states p and q are said to be equivalent if for all input strings w, $\hat{\delta}(p, w)$ is an accepting state iff $\hat{\delta}(q, w)$ is accepting.

i.e, for all input symbols of DFA model

$\delta(p, a) = \delta(q, a)$

If two states are not equivalent then they are called as Distinguishable pairs.

step-3:
To find the equivalent pairs in the automata, first find the distinguishable pairs in the automata.

step-4:
The algorithm that is used to find the equivalent pair of automata is called as table filling algorithm and its recursion.

Table Filling Algorithm:

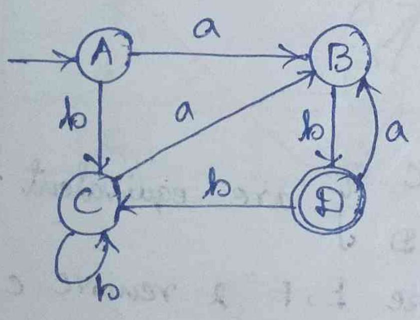
* If 'p' is an accepting state and 'q' is an non-accepting state then the pair (p,q) is distinguishable pair.

* The transition for an input symbol with 'p' is, $\delta(p,a) = r$ and 'q' is $\delta(q,a) = s$ where r and s are different states then p,q is a distinguishable pair.

* If any two states are not distinguishable by an table filling algorithm then they are equivalent pairs.

Problem:

1) Find the minimized DFA for the following model.



Transition table:

δ	a	b
→ A	B	C
B	ϕ	D
C	B	C
* D	B	C

sol:

B	X		
C		X	
D	X	X	X
	A	B	C

D ⟨ A, B, C

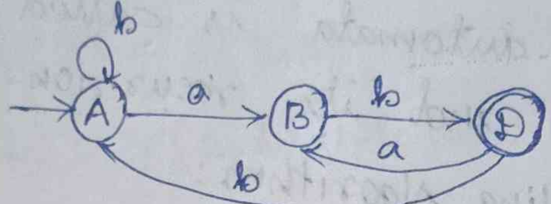
A & C are equivalent pairs

take A & remove C (replace C with A)
↓
initial state

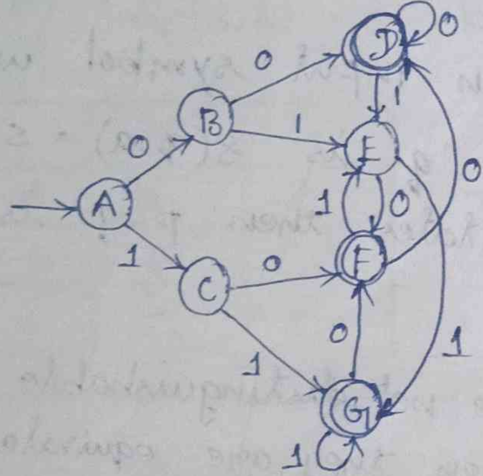
(accepting & non-accepting cannot be a equivalent state)

i/p

δ	a	b
A	B	A
B	φ	D
D	B	A



2) Find the minimized DFA for the given model.



δ	0	1
→A	B	C
B	D	E
C	F	G
*D	D	E
E	F	G
*F	D	E
*G	F	G

sol:

B	B					
C	x	x				
D	X	X	X			
E	x	x		X		
F	X	X	X		X	
G	X	X	X	x	X	x
	A	B	C	D	E	F

D ⟨ A, B, C, E

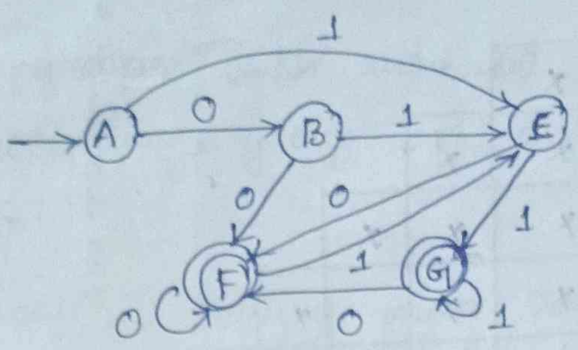
F ⟨ A, B, C, E

G ⟨ A, B, C, E

E & C } are equivalent pairs
F & D

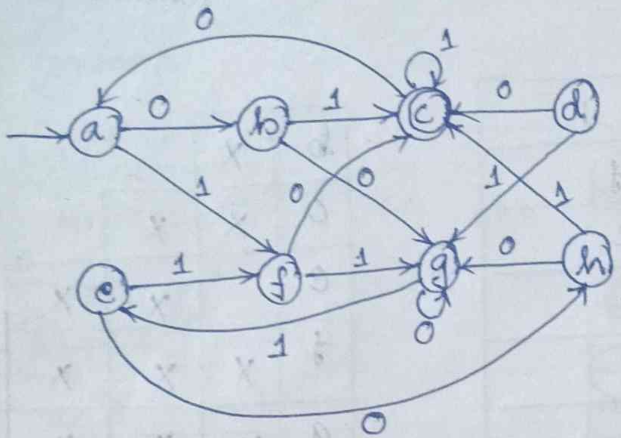
take E, F & remove C, D

δ	0	1
→A	B	E
B	F	E
E	F	G
*F	F	E
*G	F	G



30|8|24

3) Find the minimized DFA for the given automata.



sol:

i/p

δ	0	1
→ a	b	f
b	g	c
* c	a	c
d	c	g
e	h	f
f	c	g
g	g	e
h	g	c

b	x						
c	x	x					
d	x	x	x				
e	x	x	x	x			
f	x	x	x		x		
g	x	x	x	x	x	x	
h	x		x	x	x	x	x
	a	b	c	d	e	f	g

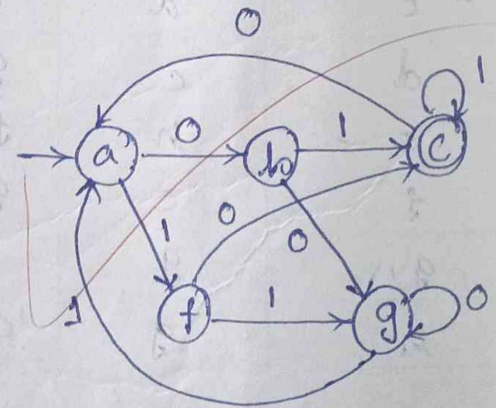
b, h & d-f are equivalent pairs take b & f and remove h & d

δ	0	1
→ a	b	f
b	g	c
* c	a	c
e	b	f
f	c	g
g	g	e

b	x				
c	x	x			
e		x	x		
f	x	x	x	x	
g	x	x	x	x	x
	a	b	c	e	f

a, e are equivalent pair take a & remove e.

δ	0	1
→ a	b	f
b	g	c
* c	a	c
f	c	g
g	g	a



Grouping method:

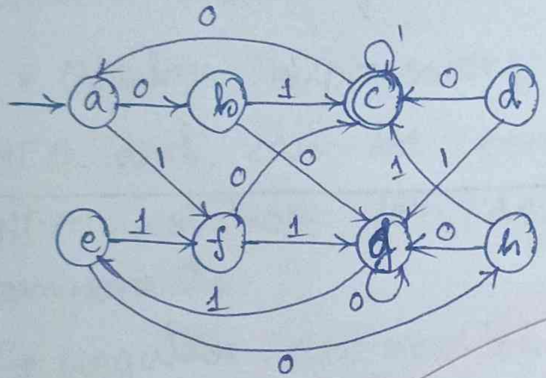
step-1: Form two groups with the set of states of DFA. One is accepting group and another is non-accepting group.

step-2: If the group is having more than one state, check the transition for all input symbols. Group the states that are having transition to same group. If not form a new group.

step-3: Repeat the process until no new group can be formed.

Problem

Find the minimized DFA for the given model.



S	0	1
→a	b	f
b	g	c
*c	a	c
d	c	g
e	h	f
f	c	g
g	g	e
h	g	c

group 1 = {a, b, d, e, f, g, h}

group 2 = {c}

	✓ a	// b	# d	✓ e	# f	✓ g	// h
0	1	1	2	1	2	1	1
1	1	2	1	1	1	1	2

group 3 = {a, e, g}

group 4 = {b, h}

group 5 = {d, f}

	✓ a	// b	# d	✓ e	# f	? g	// h
0	4	3	2	4	2	3	3
1	5	2	3	5	3	3	2

group 6 = {a, e}

group 7 = {b, h}

group 8 = {d, f}

group 9 = {g}

	✓ a	// b	# d	✓ e	# f	// h
0	7	9	2	7	2	9
1	8	2	9	8	9	2

(the same group is repeating, so stop here)

{a, e}, {b, h}, {d, f} are equivalent pairs.

take a, b, f & remove e, h, d

δ	0	1
→ a	b	f
b	g	c
* c	a	c
f	c	g
g	g	a