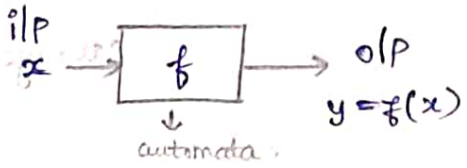
UNIT-1

FORMAL PROOF AND AUTOMATA

Applications of TOC:

1) Compiler design.
2) Robotics.
3) Artificial intelligence.
4) Knowledge engineering.

⇒TOC deals with the underlaying concept of computation.



1. Finite Automata (FA).
   → can be used for simple calculations/problems.
2. Turing Machine (TM)
   → can be used for solving complex problems.

Need for Automata Theory:

1. Essential for the study of the limits of computation.
2. Designing and checking the behaviour of digital circuits.
3. Pattern searching in websites.
4. Verify system of all types that have a finite no. of distinct states, such as communication protocols.

FINITE AUTOMATA:

→ Finite automata are useful model for important types of hardware and software.

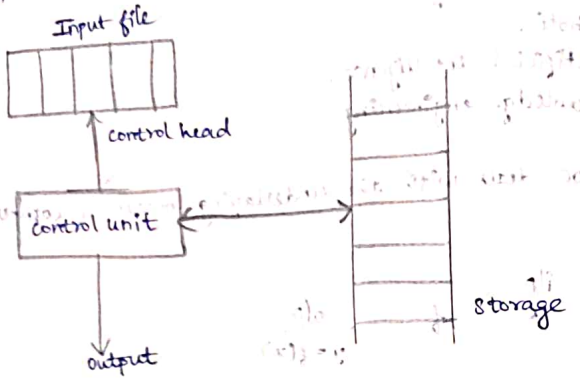
→ It is a mathematical model of a system with discrete inputs and outputs.

→ The system can be in any one of the finite no. of states.

→ The state provides information about the inputs and behaviour of the system.

→ A finite automata has a mechanism to read input which is a string over a given alphabet, this input is actually written on an input file which can be read by the automata but cannot be changed.

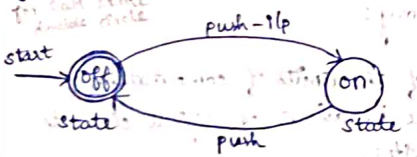
Automaton.



Advantages:

1. Implement a system with fixed set of resources.
2. Implement a system within a hardware circuit
3. complementing a system using software with finite set of inputs
4. Designing simple logical circuits
5. Lexical analysis of compiler.

Eg: on/off switch



Finite automata model for ON/OFF switch

29/1/24 Monday

INTRODUCTION TO FORMAL PROOF:

A formal proof is a finite sequence of sentences each of which follows from the preceding sentence in the sequence by a rule of inference.

1. Deductive proof.
2. Reduction to definitions.
3. other theorem forms.
4. Theorems that appear not to be if then statements.

1. DEDUCTIVE PROOF:

A deductive proof consist of a sequence of statements whose truth links us from some initial statements as "Hypothesis" or a given statement to a "conclusion" statement.

The Hypothesis may be true or false depending on values of its parameters.

Format:

Hypothesis — H
conclusion — C

If H then C ⇒ C is deduced from H

Eg: THEOREM:1

$x \geq 4$, then $2^x \geq x^2$

This is similar to If H then C format.

Its truth depends on the parameter x.

H is true for $x = 6$
$6 \geq 4$ ⇒ True.

H is false for $x = 3$.
$3 \geq 4$ ⇒ False

C is true for $x = 6$
$2^6 \geq 6^2$
$64 \geq 36$ ⇒ True.

C is false for $x = 3$
$2^3 > 3^2$
$8 \geq 9$ ⇒ False.

∴ The theorem "$x \geq 4$, then $2^x \geq x^2$" is true.

THEOREM: 2

Square of even no. is even.
Sum of any even and odd no. is odd.

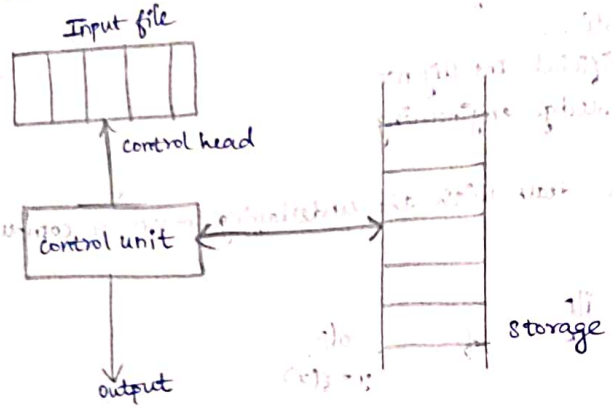
If x is the sum of the squares of four ~~possible~~ positive integers, then $2^x \geq x^2$.

H = If x is the sum of the square of 4 +ve integers,
C = $2^x \geq x^2$.

	statement	Justification.
1.	$x = a^2+b^2+c^2+d^2$	Given.
2.	$a \geq 1, b \geq 1, c \geq 1, d \geq 1$	Given.
3.	$a^2 \geq 1, b^2 \geq 1, c^2 \geq 1, d^2 \geq 1$	(2) & property of arithmetic. [if $y \geq 1$ then $y^2 \geq 1$]
4.	$x = 1+1+1+1$ $x \geq 4$	(1), (3) and Modus ponens rule.
5.	$2^x \geq x^2$	(4) and Theorem (1)

which is a string over a given alphabet, ... actually written on an input file which can be read by the automata but cannot be changed.

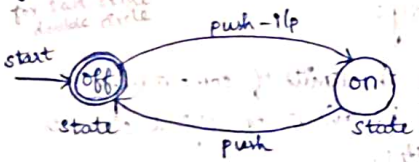
Automaton.



Advantages:

1. Implement a system with fixed set of resources.
2. Implement a system within a hardware circuit
3. complementing a system using software with finite set of inputs
4. Designing simple logical circuits.
5. Lexical analysis of compiler.

Eg: on/off switch



Finite automata model for ON/OFF switch

29/1/24 Monday

INTRODUCTION TO FORMAL PROOF:

A formal proof is a finite sequence of sentences each of which follows from the preceding sentence in the sequence by a rule of inference.

1. Deductive proof.
2. Reduction to definitions.
3. other theorem forms.
4. Theorems that appear not to be if then statements.

DEDUCTIVE PROOF:

A deductive proof consist of a sequence of statements whose truth leads us from some initial statements as "Hypothesis" or a given a statement to a "conclusion" statement.

The Hypothesis may be true or false depending on values of its parameters.

Format:

Hypothesis —— H
conclusion —— C

If H then C ⇒ C is deduced from H

Eg: THEOREM:1

$x \geq 4$, then $2^x \geq x^2$.

This is similar to If H then C format.

Its truth depends on the parameter x.

H is true for $x=6$.
$6 \geq 4 \Rightarrow$ True.
H is false for $x=3$.
$3 \geq 4 \Rightarrow$ False

C is true for $x=6$
$2^6 \geq 6^2$
$64 \geq 36 \Rightarrow$ True.
C is false for $x=3$
$2^3 > 3^2$
$8 \geq 9 \Rightarrow$ False.

∴ The theorem $x \geq 4$, then $2^x \geq x^2$ is true.

THEOREM: 2

Square of even no. is even.
Sum of any even and odd no. is odd.

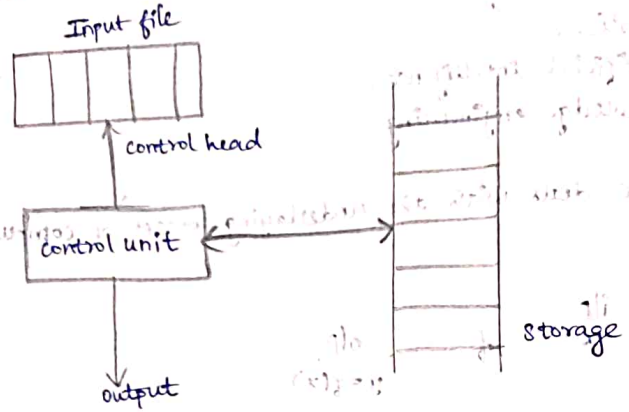
If x is the sum of the squares of four ~~possible~~ positive integers, then $2^x \geq x^2$.

H = If x is the sum of the square of 4 +ve integers
C = $2^x \geq x^2$.

	statement	Justification.
1.	$x = a^2+b^2+c^2+d^2$	Given.
2.	$a \geq 1, b \geq 1, c \geq 1, d \geq 1$	Given.
3.	$a^2 \geq 1, b^2 \geq 1, c^2 \geq 1, d^2 \geq 1$	(2) & property of arithmetic. [if $y \geq 1$ then $y^2 \geq 1$]
4.	$x = 1+1+1+1$ $x \geq 4$	(1), (3) and Modus ponens rule.
5.	$2^x \geq x^2$	(4) and Theorem (1)

which is a string over a given alphabet actually written on an input file which can be read by the automata but cannot be changed.

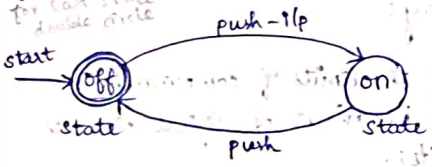
Automaton.



Advantages:

1. Implement a system with fixed set of resources.
2. Implement a system within a hardware circuit
3. complementing a system using software with finite set of codes
4. Designing simple logical circuits.
5. Lexical analysis of compiler.

Eg: on/off switch



Finite automata model for ON/OFF switch

29/1/24 Monday

INTRODUCTION TO FORMAL PROOF:

A formal proof is a finite sequence of sentences each of which follows from the preceding sentence in the sequence by a rule of inference.

1. Deductive proof.
2. Reduction to definitions
3. other theorem forms.
4. Theorems that appear not to be if then statements.

A deductive proof consist of a sequence of statements whose truth leads us from some initial statements as "Hypothesis" or a given a statement to a "conclusion" statement.

The Hypothesis may be true or false depending o values of its parameters.

should not take NULL value (ze

Format:

Hypothesis — H
conclusion — C

If H then C ⇒ C is deduced from H

Eg: THEOREM: 1

$x \geq 4$, then $2^x \geq x^2$

This is similar to If H then c format.

Its truth depends on the parameter x.

H is true for $x=6$.

$6 \geq 4 \Rightarrow$ True.

H is false for $x=3$.

$3 \geq 4 \Rightarrow$ False

C is true for $x=6$

$2^6 \geq 6^2$

$64 \geq 36 \Rightarrow$ True.

C is false for $x=3$

$2^3 > 3^2$

$8 \geq 9 \Rightarrow$ False.

∴ The theorem $x \geq 4$, then $2^x \geq x^2$ is true.

THEOREM: 2

Square of even no. is even.
Sum of any even and odd no. is odd.

If x is the sum of the squares of four ~~possible~~ po integers, then $2^x \geq x^2$.

H = If x is the sum of the square of 4 +ve intege

C = $2^x \geq x^2$.

	Statement	Justification.
1.	$x = a^2+b^2+c^2+d^2$	Given.
2.	$a \geq 1, b \geq 1, c \geq 1, d \geq 1$	Given.
3.	$a^2 \geq 1, b^2 \geq 1, c^2 \geq 1, d^2 \geq 1$	(2) & property of arithmetic. [if $y \geq 1$ then $y^2 \geq 1$]
4.	$x = 1+1+1+1$ $x \geq 4$	(1), (3) and Modus ponens rule.
5.	$2^x \geq x^2$	(4) and Theorem (1)

2. **REDUCTION TO DEFINITIONS:** If we start we go for this method

A set S is finite if there exists an integer n such that S has exactly n elements,

$\boxed{\|S\| = n}$ (no. of elements in set S)

If S and T are both subsets of some set U then T is a complement of S if and only if $S \cup T = U$ and $S \cap T = \phi$

eg: $U = \{5,7\}$, $S = \{1,2,3\}$, $T = \{4,5,6\}$

1. $S \cup T = U$; $U = \{1,2,3,4,5,6\}$
2. $S \cap T = \phi$; $S \cap T = \{\}$

Example: 1

Let S be a finite subset of some infinite set U. Let T be the complement of S with respect to U then T is infinite prove by using reduction to definition.

It is very difficult to prove by reduction so, it is proved by proof of contradiction where we take the negation of the conclusion and prove that the hypothesis is false.

	Statement	Justification
1.	T is finite.	from the negation of conclusion.
2.	$\|\|T\|\| = n$	from (1)
3.	S is finite.	Given.
4.	$\|\|S\|\| = m$	from (3)
5.	$\|\|S\|\|\|\|T\| = \|\|U\|\|$	from (2) x (4).
6.	$\|\|U\|\|$ is finite	False of hypothesis.
7.	T is infinite	Hence the hypothesis ~~is false~~.

3. **OTHER THEOREM FORMS:**

1. If then theorem forms.
2. If and only if statements.

i) **If then theorem.**

"If H then C", if hypothesis H is true for a given value of the parameter, then the conclusion C is true for some value if H then C might appear i) H implies C ii) H only if C iii) C if H iv) whenever H holds, then C follows

Example:

$x \geq 4$ then $2^x > x^2$.

i) H implies C $\rightarrow$ $x \geq 4$ implies $2^x > x^2$

ii) H only if C $\rightarrow$ $x \geq 4$ only if $2^x > x^2$

iii) C if H $\rightarrow$ $2^x > x^2$ if $x \geq 4$

iv) whenever H holds then C follows $\rightarrow$ whenever $x \geq 4$ holds then $2^x > x^2$ follows.

ii) **If and only if statement.**

"A if and only if B", this statement have different forms

A iff B is proved by two if then statements,

i) if A then B

ii) if B then A.

We prove A iff B by proving the following statements.

i) If part : if B then A

ii) only if part : if A then B

*can be denoted as $A \equiv B$ (or) $A \leftrightarrow B$

31/7/24 Wednesday

Notations:

$\Rightarrow \lfloor x \rfloor$ The floor of a real number x is the greatest integer equal to (or) less than x.

$10.6 = 10$
$10.2 = 10$

$\hookrightarrow \lfloor x \rfloor$

$\Rightarrow \lceil x \rceil$ – The ceiling of a real number x is the least integer equal to (or) greater than x.

$10.6 = 11$

THEOREM:

Let x be a real number then $\lfloor x \rfloor = \lceil x \rceil$ iff x is an integer.

PROOF:

only if part:

Assume $\lfloor x \rfloor = \lceil x \rceil$ and we have to prove x is an integer.

W.K.T, $\lfloor x \rfloor \leq x \rightarrow (1)$

$\lceil x \rceil \geq x \rightarrow (2)$

sub the ceiling for the floor in (1).

$\lceil x \rceil \leq x$

since both $\lceil x \rceil \leq x$ and $\lceil x \rceil \geq x$, by the property of arithmetic inequality we conclude $\lceil x \rceil = x$.

since $\lceil x \rceil$ is always an integer, x must also be an integer.

If part:

Assume x is an integer and we have to prove $\lfloor x \rfloor = \lceil x \rceil$.

By the definition of floor and ceiling when x is an integer both $\lceil x \rceil$ and $\lfloor x \rfloor$ are equal to x.

$\therefore \lfloor x \rfloor = \lceil x \rceil$.

4. ADDITIONAL FORMS OF PROOF:

1. Proof about sets.
2. Proof by contradiction.
3. Proof by counter example.

PROVING EQUIVALENCE ABOUT SETS:

A group of characters or strings which forms a language is called set.

If E and F are 2 expressions representing the sets, the statement E=F means that two sets represented are the same (i.e), every element in E is in F.

PROPERTIES:

1. Commutative law:

$$a \cup b = b \cup a$$
$$a \cap b = b \cap a$$

2. Distributive law:

$$a \cup (b \cap c) = (a \cup b) \cap (a \cup c)$$
$$a \cap (b \cup c) = (a \cap b) \cup (a \cap c)$$

3. Associative law:

$$a \cup (b \cup c) = (a \cup b) \cup c$$
$$a \cap (b \cap c) = (a \cap b) \cap c$$

THEOREM:

Proof that the two sets generated in the distributive law of union over intersection are equal.

PROOF:

Take two sets E and F,

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

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

Take an element 'x' in E and proof that x is in F.

	Statement	Justification.
1.	x is in $A \cup (B \cap C)$	Given statement.
2.	x is in A or x is in $B \cap C$	From the definition of union $\propto$(1)
3.	x is in A or (x is in B and x is in C)	From the definition of intersection $\propto$(2)
4.	x is in A or B. and x is in A or C	From the definition of union and intersection $\propto$(3)
5.	x is in $(A \cup B)$ and x is in $(A \cup C)$	From (4) $\propto$ definition of union and intersection.
6.	x is in $(A \cup B) \cap (A \cup C)$	From (5) $\propto$ definition of intersection.
7.	x is in F	Proved.

$\therefore$ If x is in E then x is in F has been proved.

Take an element 'x' in F and prove that x is in E

	Statement	Justification.
1.	x is in $(A \cup B) \cap (A \cup C)$	Given statement.
2.	x is in $(A \cup B)$ and x is in $(A \cup C)$	From (1) $\propto$ definition of intersection.
~~3.~~	x is in A or B and x is in A or C.	
9.	x is in A or x is in B and x is in A	From (2) $\propto$ union

4.	x is in $A \cup (B \cap C)$	x is in and proved.
5.	x is in E	

∴ If x is in F then x is in E has been proved.

By these two parts, we proved that E=F i.e,

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

Hence two sets generated in the distributive law of union over intersection are equal.

11/8/24 Thursday

PROOF BY CONTRADICTION (OR) CONTRAPOSITIVE:

Contrapositive of "If H then C" is "if not C then not H".

Four ways of representing this proof is,

1. H and C are true.
2. H and C are false.
3. H true and C false.
4. H false and C true.

Eg:

If $x \geq 4$ then $2^x \geq x^2$.

The contrapositive statement is,

"If not $2^x \geq x^2$ then not $x \geq 4$". It can be rewritten as "If $2^x < x^2$ then $x < 4$". sol → refer theorem 1

PROOF BY COUNTER EXAMPLE:

To decide whether or not a theorem is true, we have to prove the theorem. It is easier to prove that a statement is not a theorem than to prove it is a theorem. If we cannot prove it, then try to prove the theorem statement is false.

Eg:

All prime integers are odd.

⇒ The statement cannot be proved by numbers of odd values, it can be disproved with the value 2 which is even.

DISPROOF:

The integer 2 is prime but it is an even number so the statement is disproved.

THEOREM:

There is no pair of integers a and b such that a mod b = b mod a.

Possible cases for testing the given statement are,

i) $a < b$

ii) $a > b$

iii) $a = b$

case(i): $a < b$ $a = 3, b = 5$

if $a < b$ then

a mod b is a → ①
↳ 3 mod 5 = 3

b mod a is 0 to a-1 → ②
↳ 5 mod 3 = 2

From ① and ②, a mod b ≠ b mod a

⇒ True for this case.

case(ii): $a > b$ $a = 5, b = 3$

if $a \geq b$ then

a mod b is 0 to b-1 → ③
↳ 5 mod 3 = 2

b mod a is b → ④
↳ 3 mod 5 = 3.

From ③ and ④, a mod b ≠ b mod a

⇒ True for this case.

case(iii): $a = b$ $a = 5, b = 5$

if $a = b$ then

a mod b is 0 → ⑤
↳ 5 mod 5 = 0

b mod a is 0 → ⑥
↳ 5 mod 5 = 0

From ⑤ and ⑥, a mod b = b mod a

⇒ False for this case.

The theorem statement becomes false in case(iii) and therefore the theorem is disproved so, the theorem can be restated for some pair of integers such that a mod b = b mod a iff $a = b$.

2/8/24 Friday

only if part: if a mod b = b mod a.

if part: if a=b then a mod b = b mod a

Proving only if part (contradiction)

	Statement	Justification.
1.	$a \neq b$	negation of the conclusion.
2.	when $a<b$ a mod b = a b mod a = 0 to a-1	from the defn of mod.
3.	when $a>b$ a mod b = 0 to b-1 b mod a = b	From the defn of mod.
4.	a mod b ≠ b mod a	from ② & ③

one(s) or b is not taken since it is false.

∴ The negation of hypothesis is arrived. Then the statement is true.

Proving if part (deduction method)

	Statement	Justification.
1.	a=b	Given statement
2.	a mod b = 0 b mod a = 0	From the defn of mod
3.	a mod b = b mod a	From (2)

Hence if and only if part is true.

∴ Hence for some pair of integers a and b such that a mod b = b mod a if and only if a=b is true.

MATHEMATICAL INDUCTION:

→ Induction on Integers
→ Structural Induction.
→ Mutual Induction.

PROOF BY INDUCTION METHOD:

Special forms of proof called inductive proof is essential when dealing with recursively defined objects. Most familiar inductive proof deals with integers but in automata theory it is used to deal with concepts of trees, expressions of various types.

i) INDUCTION ON INTEGERS:

Given a statement S(n) for an integer "n", the statement can be proved by induction method as follows,

1. Basis: → verify if the statement is true for the lowest possible value. In this step mostly we take n value as 0 (or) 1.
2. Inductive step → In this step we assume S(n) is true for $n \geq i$, where i is the basis integer and check whether S(n+1) is true.

NOTE:

If the truth of S(n) leads to truth of S(n+1) then the statement S(n) is true for all values of n

Example:

Prove by induction. $1+2+3+\cdots\cdots+n = \frac{n(n+1)}{2}$

i) Basis:

let n=1 (lowest value)

LHS = 1

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

RHS = 1

LHS = RHS.

ii) Inductive step:

assume S(n) is true, i.e

$S(n) = 1+2+3+\cdots\cdots+n = \frac{n(n+1)}{2}$ is true.

then prove $s(n+1) = \underbrace{1+2+3+\cdots+n}_{\Downarrow \frac{n(n+1)}{2}} + n+1 = \frac{(n+1)(n+2)}{2}$ (RHS)

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

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

$\frac{n^2+n+2n+2}{2} = \frac{n^2+2n+n+2}{2}$

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

$\therefore$ LHS = RHS

Hence it is proved, In this example ~~they~~ we have taken n=1 as first step and proved s(1) is true. In the next step we have proved s(n) is true means s(n+1) is also true. since s(1) is true in first step (basis), s(2) also true. similarly s(2) true means s(3) also true.

5/8/24 Monday

1) Prove Proof by induction $1^2+2^2+3^2+\cdots+n^2 = \frac{n(n+1)(2n+1)}{6}$

sol:

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

i) Basis:

let n=1

LHS $= n^2$ ($\because$ lowest value)

$= 1^2$

$= 1$

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

$= 6/6$

RHS = 1

LHS = RHS

ii) Inductive step:

Let us assume s(n) is true. i.e,

$s(n) = 1^2+2^2+3^2+\cdots+n^2 = \frac{n(n+1)(2n+1)}{6}$ is true

then prove s(n+1) is true

$s(n+1) = \underbrace{1^2+2^2+3^2+\cdots+n^2}_{\frac{n(n+1)(2n+1)}{6}} + (n+1)^2 = \frac{(n+1)(n+1+1)(2(n+1)+1)}{6}$

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

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

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

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

$= \frac{2n^3+9n^2+13n+6}{6} = \frac{2n^3+3n^2+6n^2+9n+4n+6}{6}$

$\therefore$ LHS = RHS

$\therefore s(n+1)$ is true when s(n) is true.

So, the given statement is true for all values of n.

2) For $n \geq 1$ prove by induction, $2^1+2^2+2^3+\cdots+2^n = 2^{n+1}-2$

sol: s(n)

i) Basis:

let n=1

LHS $= 2^n$

$= 2^1$

$= 2$

RHS $= 2^{n+1}-2$

$= 2^{1+1}-2$

$= 2^2-2 = 4-2$

$= 2$

LHS = RHS.

ii) Inductive step:

Let us assume s(n) is true. i.e,

$s(n) = 2^1+2^2+2^3+\cdots+2^n = 2^{n+1}-2$ is true

then prove s(n+1) is true.

$s(n+1) = \underbrace{2^1+2^2+2^3+\cdots+2^n}+2^{n+1} = 2^{(n+1+1)}-2$

LHS $2^{n+1}-2+2^{n+1}$

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

$2^{n+1} \times 2^1 - 2$

$2^{n+2}-2$

RHS $= 2^{n+2}-2$

$\therefore$ LHS = RHS

$\therefore s(n+1)$ is true when s(n) is true.

So, the given statement is true for all values of n

3) using mathematical induction prove 5^n-2^n is divisible by for every $n \geq 0$

sol:

i) Basis:

let n = 0

$\Rightarrow 5^n-2^n \Rightarrow 5^0-2^0$

$\Rightarrow 1-1 \Rightarrow \boxed{0}$

let n = 1

$\Rightarrow 5^n-2^n \Rightarrow 5^1-2^1$

$\Rightarrow 5-2 \Rightarrow \boxed{3}$

$\therefore$ s(n) is true.

Assume $S(n)$

prove $S(n+1)$ is true. i.e,

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

$= 5^n 5^1 - 2^n \cdot 2^1$

Add and sub the same value $5 \cdot 2^n$

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

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

$5-2=3$ which is divisible

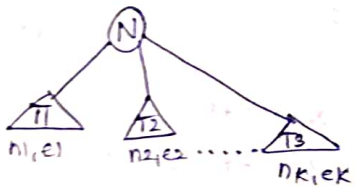
$S(n)$ already proved in Basis

$\therefore 5^{n+1} - 2^{n+1}$ is divisible by 3 proved.

So the given statement $5^n - 2^n$ is divisible by 3 for $n \geq 0$ is true.

II) STRUCTURAL INDUCTION:

Every tree has one more node than its edges (i.e) if T is a tree, and T has n nodes and e edges then $n = e+1$



Basis:

Let us take no. of nodes $n=1$ the $e=0$

So, the statement $n = e+1$ is true

Inductive step:

Assume $n = e+1$ is true,

No. of nodes in tree T is written as, $n = n_1 + n_2 + n_3 \cdots + \rightarrow (1)$

No. of edges in the tree is written as, $e = k + e_1 + e_2 + e_3 + \cdots + \rightarrow (2)$

$k \rightarrow$ edges for connecting k subtrees to the root node

$n = 1 + n_1 + n_2 + n_3 \cdots + n_k$

Sub $n = e+1$,

$n = 1 + (e_1+1) + (e_2+1) + \cdots (e_k+1)$

$n = 1 + e$

$\therefore$ Every tree has one more node than its edges is proved.

H.W

1) For $n \geq 1$ prove by induction $1\times2 + 2\times3 + 3\times4 + \cdots + n(n+1) = \frac{n(n+1)(n+2)}{3}$

2) Using principle of mathematical induction prove, $4^{n+4} + 3^{n+2}$ is multiple of 13.

Answers:

1) $S(n) = 1\times2 + 2\times3 + 3\times4 + \cdots + n(n+1) = \frac{n(n+1)(n+2)}{3}$

I) Basis:

let $n=1$

LHS: $n(n+1)$

$= 1(1+1)$

LHS $= 2$

RHS: $\frac{n(n+1)(n+2)}{3}$

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

RHS $= 2$

LHS = RHS.

II) Inductive step:

Assume $S(n) = 1\times2 + 2\times3 + 3\times4 + \cdots + n(n+1) = \frac{n(n+1)(n+2)}{3}$ is true. Then prove $S(n+1)$ is true. i.e,

$S(n+1) = 1\times2 + 2\times3 + 3\times4 + \cdots + n(n+1) + (n+1)(n+1+1)$

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

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

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

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

$\frac{n^3+3n^2+5n+3}{3} = \frac{n^3+6n^2+11n+6}{3}$

$\frac{n^3+2n^2+n^2+2n+3n^2+6n+3n+6}{3} = \frac{n^3+3n^2+3n^2+9n+6}{3}$

$\frac{n^3+6n^2+11n+6}{3} = \frac{n^3+6n^2+11n+6}{3}$

LHS = RHS

$S(n) = 4^{2n+1} + 3^{n+2}$ is multiple of 13

2) Basis:

Let $n=0$ $\quad 4^{2(0)+1} + 3^{0+2} = 4^{0+1} + 3^2 = 4^1 + 3^2 = 4+9 = 13$

Let $n=1$ $\quad 4^{2(1)+1} + 3^{1+2} = 4^{2+1} + 3^3 = 4^3 + 3^3 = 64+27 = 91$

$S(0)$ and $S(1)$ is true.

Inductive step:

Assume $S(n) = 4^{2n+1} + 3^{n+2}$ is divisible by 13 the

prove $S(n+1)$ is true.

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

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

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

Add & sub by $4^2 \cdot 3^{n+2}$:

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

$$= 4^2 \left[4^{2n+1} + 3^{n+2}\right] + 3^{n+2}\left[3 - 16\right]$$

(first bracket: ⇓ Divisible by 13; second bracket: ⇓ Divisible by 13)

∴ Hence proved.

12/8/19 Monday

MUTUAL INDUCTION:

Instead of proving a single statement by induction, it is necessary to prove a group of statements; $S_1(n), S_2(n) \ldots S_k(n)$ together a induction on n. This type of proof is called proof by mutual induction.

Eg: on-off switch.

CONCEPTS OF AUTOMATA THEORY:

1) Alphabets – It is a finite set of symbols which is not empty and denoted by Σ

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

2) String – String is finite sequence of symbols from some alphabet symbol

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

Possible strings = 0, 01, 10, 11 etc.

3) Empty string – A string with 0 symbols is an empty string which is denoted by 'ε'

4) Length of the string – No. of symbols (or) no of positions for the symbols in the string is called the length of the string. It is denoted by "| |".

Eg: If w is the string, its length is denoted by $|w|$.

$w = 0100$ over $\{0,1\}$

$|w| = 4$

NOTE:

The length of empty string or NULL string is zero.

5. Power of an alphabet – If Σ is an alphabet, the set of all strings of certain lengths formed from that alphabet is denoted as power of an alphabet. Expressed using an exponen notation.

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

$\Sigma^0 = \{\varepsilon\}$ $\quad \Sigma^2 = \{00, 01, 10, 11\}$

$\Sigma^1 = \{0,1\}$ $\quad \Sigma^3 = \{001, 010, 011, 100, 101, 110, 111\}$

$\Sigma^4, \Sigma^5, \Sigma^6 \ldots\ldots$

6) Kleene closure – The set of all strings over an alphabet Σ which includes NULL string is denoted as:

Eg: If $\Sigma = \{0,1\}$, then Σ^*?

$\Sigma^* = \{\Sigma^0 \cup \Sigma^1 \cup \Sigma^2 \cup \Sigma^3\}$

$= \{\varepsilon \cup 0,1 \cup 00,01,10,11 \cup 001,010,011,100,101,110,1$

$= \{\varepsilon, 0, 1, 00, 01, 10, 11, 001, 010, 011, 100, 110, 111\}$

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

7) Positive closure:

A set of all non-empty strings over an alphab Σ is called positive closure. Denoted by Σ^+.

Eg: $\Sigma = \{0,1\}$, then Σ^+?

$\Sigma^+ = \{0, 1, 00, 01, 10, 11\}$

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

NOTE: Relation

given by $\Sigma^* = \varepsilon + \Sigma^+$

8. concatenation of strings - Let x, y be strings then xy denotes concatenation of two strings x and y which is made by making a copy of string x followed by string y.

Eg: $x = 110 \quad y = 101.$

$xy = 110101.$

9. Language - set of all strings over summation which are choosen from Σ^* is called language.

Eg: A set of all strings with equal no. of 0's and 1's over binary alphabet $\Sigma = \{0,1\}$, then Σ^*.

indicates $\Sigma^* = \{\varepsilon, 0, 1, 00, 01, 10, 11\}$

language $L = \{1, 0, 01, 00, 11, \text{etc}\}$

10. Prefix and suffix - Any no. of leading symbol of a string are called prefix

- Any no. of trailing symbol of a string are called suffix.

$\omega = aabba$

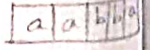
prefix: a, aa, aab, aabb, aabba

suffix: a, ba, bba, abba, aabba

continued

Finite automata contains,

- → finite no. of states
- → finite no. of input symbols.
- → finite control block.
- → input tape which is divided into no. of cells and each cell it can hold atmost one symbol.



TYPES OF FINITE AUTOMATA:

1. Deterministic finite automata (DFA)
2. Non-deterministic finite automata (NFA)

DETERMINISTIC FINITE AUTOMATA:

The term deterministic refers to the fact that on each input there is one and only one state to which the automata can have transition from its current state.

(or) components of DFA (or) Tuples.

MODEL OF FINITE AUTOMATA (OR) FORMAL DEFINITION:

The DFA can be represented by 5 tuples.

$M = \{Q, \Sigma, \delta, q_0, F\}$ where,

$Q \rightarrow$ finite set of states (i.e) $Q = \{q_0, q_1, q_2, \ldots q_n\}$

$\Sigma \rightarrow$ set of input symbols (i.e) $\Sigma = \{0,1\}$, $\Sigma = \{a, b, c\}$

$\delta \rightarrow$ Transition function which takes as argument state, input symbol and returns the state (i.e)

$\delta = Q \times \Sigma \rightarrow Q$

$q_0 \rightarrow$ Initial (or) starting state.

$F \rightarrow$ Final (or) accepting state.

Extended transition function:

It is denoted by $\hat{\delta}$. It is used whenever a string is to be processed instead of a symbol.

Eg: Let us a input string $\omega = xa$, then the transition function is represented as,

$$\hat{\delta}(q_0, \omega) = \hat{\delta}(q_0, xa)$$
$$= \delta(\hat{\delta}(q_0, x), a)$$

Transition diagram:

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

1) If $M = (\{q_0, q_1, q_2\}, \{0,1\}, \delta, q_0, \{q_2\})$ and δ is

$\delta(q_0, 0) = q_1$

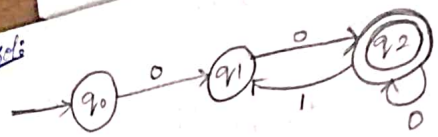
$\delta(q_1, 0) = q_2$

$\delta(q_2, 1) = q_1$

$\delta(q_2, 0) = q_2$

Draw its transition diagram?

Sol:



Transition table:

→ Transition table is a conventional tabular representation of a function δ that has two arguments and return a value.

→ The row represents state.

→ The column represents input.

State	I/P symbol	
	0	1
→ q0	q1	–
q1	q2	–
* q2	q2	q1

→ initial state

* final state

2) Construct transition table and draw the transition diagram for the states $M = (\{q_0, q_1\}, \Sigma = \{a, b\}, \delta, q_1, q_1)$

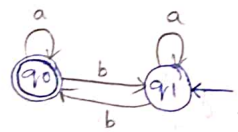
$\delta(q_0, a) = q_0$

$\delta(q_0, b) = q_1$

$\delta(q_1, a) = q_1$

$\delta(q_1, b) = q_0$

Sol:

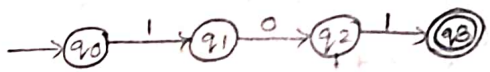


State	I/P symbol.	
	a	b
* q0	q0	q1
→ q1	q1	q0

3) Design a DFA which accepts only input 0,1 over the input set $\Sigma = \{0,1\}$

Sol:

4) Design a DFA which accepts only string 101 over the input set $\Sigma = \{0,1\}$



5) Design a DFA which accepts string starting with 0.

write the language

$L = \{0, 00, 001, 000, 10, 010 \ldots\}$

(0 is the smallest symbol (one literal))

NOTE:

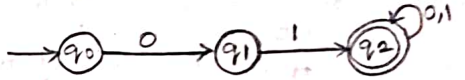
Take the minimum string in L, the no. of symbols in this string + 1 = state. 1+1 = 2 (state)

6) starting with 01

Sol:

assume, $L = \{01, 010, 011, 0100 \ldots\}$

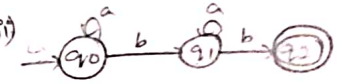
No. of states ⇒ 2+1 ⇒ 3 states



7) i) Design a DFA that starts with aba over $\Sigma = \{a, b\}$

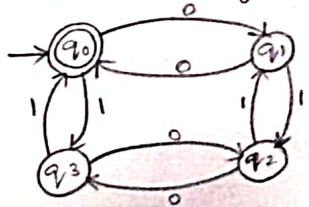
The string starts with a and end with b.

ii)

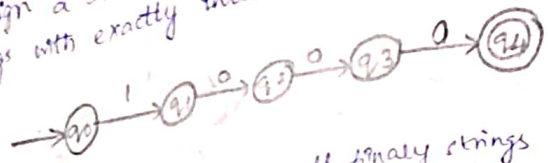


19/8/24 Monday:

8) Design a DFA ~~that~~ over $\Sigma = \{0,1\}$ that accepts even no. of 0's and even no of 1's.



9) Design a DFA with strings with exactly three consecutive

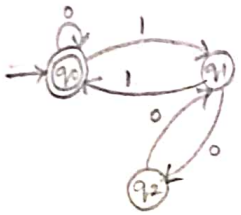


10) Construct a DFA to accept all binary strings which are divisible by 3.

$L = \{0, 11, 110, 1001, 1100, 1111, \ldots\}$

~~No of states = 2 + 1~~
~~= 3~~

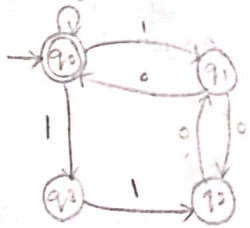
NOTE:
Three remainders will be there when we divide a number by 3 (i.e), 0,1,2.
So, we can draw 3 states.



11) Construct a DFA that accepts all string over alphabet $\Sigma = \{0,1\}$ where binary integers are divisible by 4.

$L = \{0, 100, 1000, 1100, \ldots\}$

Possible remainders
$\Rightarrow 0,1,2,3$
No. of states = 4.



NON-DETERMINISTIC FINITE AUTOMATA: (NFA)

NFA is similar to DFA with a difference, it takes a state and input as argument and returns a set of one or more states.

FORMAL DEFINITION:

NFA is defined by five tuples,

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

where, Q - finite set of states.
Σ - set of input symbols.
δ - Transition function
q_0 - Initial (or) starting state.
F - Final (or) Accepting state.

Transition function δ is given by, $Q \times (\Sigma \cup \epsilon) \rightarrow 2^Q$

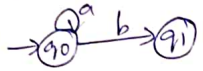
NFA	DFA
* It has a power of moving to more than one state at a time on reading a input symbol. * Easy to represent a model in NFA * Occupies less memory space. * δ is the transition function that takes a state and input symbol as argument and returns one (or) more states. $Q \times (\Sigma \cup \epsilon) \rightarrow 2^Q$ Eg:  $\{q_1, 0\} = \{q_1, q_2\}$	* It has a power to move from one state to another state on reading an input symbol. * Comparatively little bit difficult than NFA. * Occupies more memory space. * The transition function δ returns exactly one state $Q \times \Sigma \rightarrow Q$ Eg:  $\{q_1, 0\} = \{q_1\}$

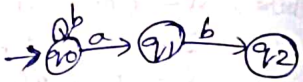
1) construct a NFA with a strings of 0's and 1's that should end with 01.

2) obtain NFA for a language consisting of all strings over a,b.

i) end with b

ii) end with ab

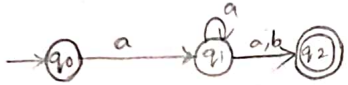
i)


ii)


3) Determine NFA that accepts the language $L = (aa^*(a+b))$

/(or) + ⇒ will use both

* ⇒ self loop

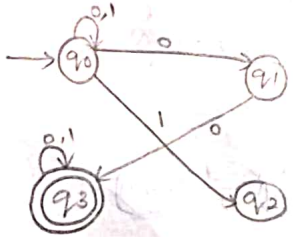


4) consider a finite automata, $M = (\{q_0, q_1, q_2, q_3\}, \{0,1\}, \delta, q_0, \{q_3\})$

with the transition,

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

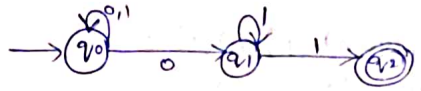
construct NFA and transition table.



Transition Table:

state	I/P symbol	
	0	1
→ q_0	q_0, q_1	q_0, q_2
q_1	q_3	ϕ
q_2	ϕ	ϕ
* q_3	q_3	q_3

5) consider the NFA and check whether the given string {00, 011, 0101, 11100} is accepted or not.



6) construct a DFA for the language $L = \{a^m, b^n\}$ where m and n are positive integers.

sol:

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

sol:

i) 00

$\hat{\delta}(q_0, w) = \hat{\delta}(q_0, 00)$
$= \delta(\hat{\delta}(q_0,0), 0)$
$= \delta(\delta(q_0,0), 0)$
$= \delta(\{q_0, q_1\}, 0)$
$= \delta(q_0,0) \cup \delta(q_1,0)$
$= \{q_0, q_1\} \cup \phi$

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

∴ This automata rejects the acce state. (not ending in final state

ii) 011

$\hat{\delta}(q_0, 011) = \hat{\delta}(q_0, 011)$
$= \delta(\hat{\delta}(q_0,0), 11)$
$= \delta((q_0, q_1), 11)$
$= \delta((q_0,1) \cup (q_1,1), 1)$
$= \delta(\{q_0, q_1, q_2\}, 1)$
$= \{q_0, q_1, q_2, \phi\}$

$\hat{\delta}(q_0, 011) = \{q_0, q_1, q_2\}$

∴ This automata is accepting state.

iii) 0101

$\hat{\delta}(q_0,0101) = \hat{\delta}(q_0,0101)$

$= \delta(\hat{\delta}(q_0,010),1)$

$= \delta(\delta(\hat{\delta}(q_0,01),0),1)$

$= \delta(\delta(\delta(\hat{\delta}(q_0,0),1),0),1)$

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

$= \delta(\delta((\delta(q_0,1)\cup\delta(q_1,1)),0),1)$

$= \delta(\delta((q_0,q_1),0),1)$

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

$= \delta(q_0,q_1,\phi,\phi),1)$

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

$\hat{\delta}(q_0,0101) = \{q_0,q_1,q_2\}$ ∴ Accepting

iv) 11100.

$\hat{\delta}(q_0,11100) = \hat{\delta}(q_0,11100)$

$= \delta(\hat{\delta}(q_0,1),1),100)$

~~$= \delta((q_0,1)\cup(q_0,1)),100)$~~

~~$= \delta(q_0),100$~~

$= \delta(\hat{\delta}(q_0),1),100)$

$= \delta(q_0,1),00)$

$= \delta((q_0),00)$

$= \delta((q_0,0),0)$

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

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

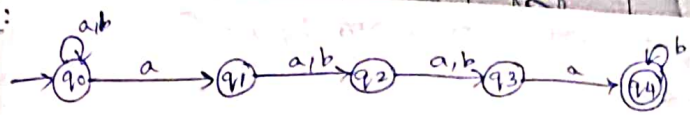
$= \{q_0,q_1,\phi\}$

$\hat{\delta}(q_0,11100) = \{q_0,q_1\}$ ∴ Not accepting.

v) Check whether the string ab and abaab is accepted or not by the below transition table.

State	I/P's	
	a	b
→ q_0	$\{q_0,q_1\}$	$\{q_0\}$
q_1	$\{q_2\}$	$\{q_2\}$
q_2	$\{q_3\}$	$\{q_3\}$
q_3	$\{q_4\}$	$\{\phi\}$
*q_4	$\{\phi\}$	$\{q_4\}$

Sol:



i) ab

$\hat{\delta}(q_0,ab) = \hat{\delta}(q_0,ab)$

$= \delta(\hat{\delta}(q_0,a),b)$

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

$= \delta(q_0,b)\cup(q_1,b)$

$\hat{\delta}(q_0,ab) = \{q_0,q_2\}$ ∴ Not accepted.

ii) abaab

$\hat{\delta}(q_0,abaab) = \hat{\delta}(q_0,abaab)$

$= \delta(\hat{\delta}(q_0,a),baab)$

$= \delta(\{q_0,q_1\},b),aab)$

$= \delta(\{q_0,q_2\},a)\,ab)$

$= \delta(\{q_0,q_1,q_3\},a),b)$

$= \delta(\{q_0,q_1,q_2,q_4\},b)$

$\hat{\delta}(q_0,abaab) = \{q_0,q_2,q_3,q_4\}$ ∴ Accepted.

NFA TO DFA CONVERSION:

→ By using subset construction method.

→ Every DFA is NFA and the class of language is accep by NFA includes the same class of languages accepted by DFA.

→ Every NFA has a equivalent DFA.

i) Let $M=(\{q_0,q_1\},\{0,1\},\delta,q_0,\{q_1\})$ be a NFA where

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

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

$\delta(q_1,0) = \phi$

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

construct its equivalent DFA.

Sol:

Diagram

	0	1
→ q_0	$\{q_0,q_1\}$	$\{q_1\}$
*q_1	ϕ	$\{q_0,q_1\}$

Table

Step 1: Initial state of DFA $[q_0] \rightarrow (1)$

Step 2:

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

$\delta'([q_0,0]) = [q_0,q_1] \rightarrow (2)$

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

$\delta'([q_0,1]) = [q_1] \rightarrow (3)$

Step 3:

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

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

$\delta'([q_0,q_1],0) = [q_0,q_1] \rightarrow (4)$

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

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

$\delta'([q_0,q_1],1) = [q_0,q_1] \rightarrow (5)$

Step 4:

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

$\delta'([q_1],0) = [\phi] \rightarrow (6)$

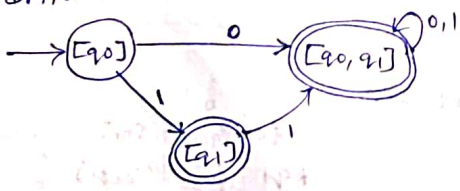
$\delta(\{q_1\},1) = \{q_0,q_1\}$

$\delta'([q_1],1) = [q_0,q_1] \rightarrow (7)$

Transition table:

	0	1
$[q_0]$	$[q_0,q_1]$	$[q_1]$
$[q_0,q_1]$	$[q_0,q_1]$	$[q_0,q_1]$
$[q_1]$	$[\phi]$	$[q_0,q_1]$

DFA:

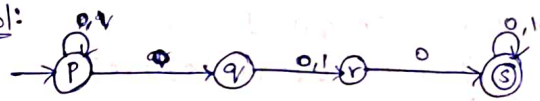


23/8/24
Friday

2) Convert the following NFA to DFA.

State	Inputs	
	0	1
$\rightarrow p$	$\{p,q\}$	$\{p\}$
q	$\{r\}$	$\{r\}$
r	$\{s\}$	ϕ
$*s$	$\{s\}$	$\{s\}$

Sol:



Step 1: Initial state of DFA $[p] \rightarrow (1)$

Step 2:

$\delta(p,0) = \{p,q\}$

$\delta'([p,0]) = [p,q] \rightarrow (2)$

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

$\delta'([p,1]) = [p] \rightarrow (3)$

Step 3:

$\delta(\{p,q\},0) = \{p,q\} \cup \{r\}$

$\delta(\{p,q\},0) = \{p,q,r\}$

$\delta'([p,q],0) = [p,q,r] \rightarrow (3)$

$\delta(\{p,q\},1) = \{p\} \cup \{r\}$

$\delta(\{p,q\},1) = \{p,r\}$

$\delta'([p,q],1) = [p,r] \rightarrow (4)$

Step 4:

$\delta(\{p,q,r\},0) = \{p,q\} \cup \{r\} \cup \{s\}$

$\delta(\{p,q,r\},0) = \{p,q,r,s\}$

$\delta'([p,q,r],0) = [p,q,r,s] \rightarrow (5)$

$\delta(\{p,q,r\},1) = \{p\} \cup \{r\} \cup \{\phi\}$

$\delta(\{p,q,r\},1) = \{p,r\}$

$\delta'([p,q,r],1) = [p,r]$

step 5:

$\delta(\{p,r\},0) = \{p,q\} \cup \{s\}$

$\delta([p,r],0) = [p,q,s]$

$\delta'([p,r],0) = [p,q,s] \rightarrow (6)$

$\delta(\{p,r\},1) = \{p\} \cup \{\phi\}$

$\delta'([p,r],1) = [p]$

step 6:

$\delta(\{p,q,r,s\},0) = \{p,q\} \cup \{r\} \cup \{s\} \cup \{s\}$

$\delta(\{p,q,r,s\},0) = \{p,q,r,s\}$

$\delta'([p,q,r,s],0) = [p,q,r,s]$

$\delta(\{p,q,r,s\},1) = \{p\} \cup \{r\} \cup \{s\} \cup \{\phi\}$

$\delta(\{p,q,r,s\},1) = \{p,r,s\}$

$\delta'([p,q,r,s],1) = [p,r,s] \rightarrow (7)$

step 7:

$\delta(\{p,q,s\},0) = \{p,q\} \cup \{r\} \cup \{s\}$

$\delta'([p,q,s],0) = [p,q,r,s]$

$\delta(\{p,q,s\},1) = \{p\} \cup \{r\} \cup \{s\}$

$\delta'([p,q,s],1) = [p,r,s] \rightarrow (8)$

step 8:

$\delta(\{p,r,s\},0) = \{p,q\} \cup \{s\} \cup \{s\}$

$\delta([p,r,s],0) = [p,q,s]$

$\delta(\{p,r,s\},1) = \{p\} \cup \{\phi\} \cup \{s\}$

$\delta'([p,r,s],1) = [p,s] \rightarrow (8)$

step 9:

$\delta(\{p,s\},0) = \{p,q\} \cup \{s\}$

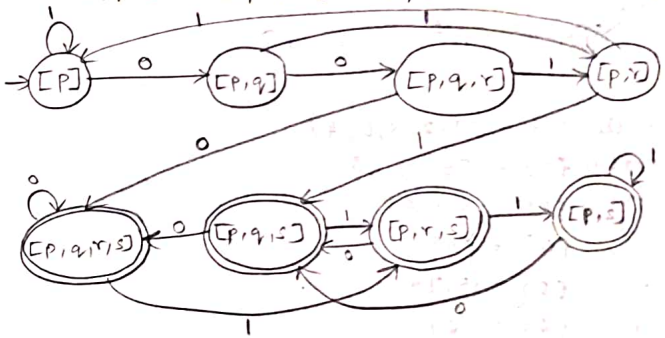
$\delta'([p,s],0) = [p,q,s]$

$\delta(\{p,s\},1) = \{p\} \cup \{s\}$

$\delta'([p,s],1) = [p,s]$

Transition table.

state	0	1
[p]	[p,q]	[p]
[p,q]	[p,q,r]	[p,r]
[p,q,r]	[p,q,r,s]	[p,r]
[p,r]	[p,q,s]	[p]
[p,q,r,s]	[p,q,r,s]	[p,r,s]
[p,q,s]	[p,q,r,s]	[p,r,s]
[p,r,s]	[p,q,s]	[p,s]
[p,s]	[p,q,s]	[p,s]



3.) H.W.

State	I/P	
	0	1
→p	{p,q}	{p}
q	ϕ	{r}
*r	{p,r}	{q}

FINITE AUTOMATA WITH EPSILON TRANSITION: (NFA-ϵ)

* NFA with epsilon can be defined as,

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

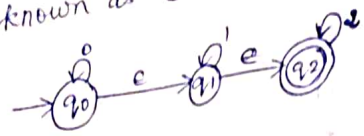
δ has the argument:

i) a state Q.

ii) a new input symbol epsilon is added to the input $(Q \times \Sigma) \cup \epsilon \rightarrow 2^Q$.

ε closure - A set of states reachable from a state on ε can be determined by using a function known as ε closure

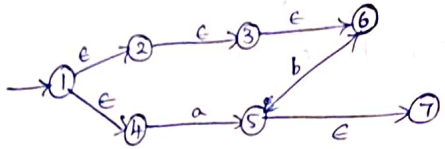
1)



Sol:

ε-closure {q0} = {q0, q1, q2}
ε-closure {q1} = {q1, q2}
ε-closure {q2} = {q2}

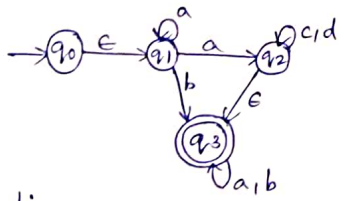
2)



Sol:

ε-closure {1} = {1, 2, 3, 6, 4}
ε-closure {2} = {2, 3, 6}
" {3} = {3, 6}
" {4} = {4}
" {5} = {5, 7}
" {6} = {6}
" {7} = {7}

3)

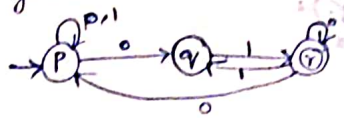


Sol:

ε-closure {q0} = {q0}
" {q1} = {q1}
" {q2} = {q2, q3}
" {q3} = {q3}

H.W Soln:

3) State diagram:



Step 1: Initial state of DFA [p] → (1)

Step 2:
δ(p, 0) = {p, q}
δ'([p, 0]) = [p, q] → (2)
δ(p, 1) = {p}
δ'([p, 1]) = [p]

Step 3:
δ({p, q}, 0) = {p, q} ∪ {ϕ}
δ'([p, q], 0) = [p, q]
δ({p, q}, 1) = {p} ∪ {r}
δ'([p, q], 1) = [p, r] → (3)

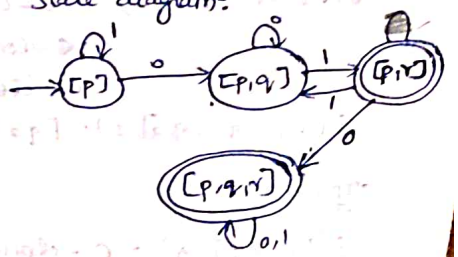
Step 4:
δ({p, r}, 0) = {p, q} ∪ {r, p}
δ'([p, r], 0) = [p, q, r] → (4)
δ({p, r}, 1) = {p} ∪ {q}
δ'([p, r], 1) = [p, q]

Step 5:
δ({p, q, r}, 0) = {p, q} ∪ {ϕ} ∪ {r, p}
δ'([p, q, r], 0) = [p, q, r]
δ({p, q, r}, 1) = {p} ∪ {r} ∪ {q}
δ'([p, q, r], 1) = [p, q, r]

Transition Table:

State	I/p 0	1
[p]	[p, q]	[p]
[p, q]	[p, q]	[p, r]
[p, r]	[p, q, r]	[p, q]
[p, q, r]	[p, q, r]	[p, q, r]

State diagram:



NFA WITH

Ad:

ε-closure {q0} = {q0, q1, q2}
ε-closure {q1} = {q1, q2}
ε-closure {q2} = {q2}

step 1:

Initial state of DFA [q0, q1, q2] ⟶ (1)

step 2:

$\delta([q_0,q_1,q_2],0)$ = ε-closure $(\delta(\hat{\delta}[q_0,q_1,q_2],\epsilon),0)$
= ε-closure $(\delta[q_0,q_1,q_2],0)$
= ε-closure $(\{q_0,\phi,\phi\})$

$\delta([q_0,q_1,q_2],0)$ = [q0, q1, q2] (∵ existing)

step 3:

$\delta([q_0,q_1,q_2],1)$ = ε-closure $(\delta(\hat{\delta}[q_0,q_1,q_2],\epsilon),1)$
= ε-closure $(\delta[q_0,q_1,q_2],1)$
= ε-closure $(\{\phi,q_1,\phi\})$

$\delta([q_0,q_1,q_2],1)$ = [q1, q2] ⟶ (2)

step 3:

$\delta([q_1,q_2],2)$ = ε-closure $(\delta(\hat{\delta}[q_1,q_2],\epsilon),2)$
= ε-closure $(\delta[q_1,q_2],2)$
= ε-closure $(\{\phi,\phi\})$

$\delta([q_0,q_1,q_2],2)$ = ε-closure $(\delta(\hat{\delta}[q_0,q_1,q_2],\epsilon),2)$
= ε-closure $(\delta[q_0,q_1,q_2],2)$
= ε closure $(\{\phi,\phi,q_2\})$

$\delta([q_0,q_1,q_2],2)$ = [q2] ⟶ (3)

step 3:

$\delta([q_1,q_2],0)$ = ε-closure $(\delta(\hat{\delta}[q_1,q_2],\epsilon),0)$
= ε-closure $(\delta([q_1,q_2]),0)$
= ε-closure $(\{\phi,\phi\})$

$\delta([q_1,q_2],0)$ = [ϕ] ⟶ (4)

= ε-closure $(\delta[q_1,q_2],1)$
= ε-closure $(\{q_1,q_2\})$

$\delta([q_1,q_2],1)$ = [q1, q2] (∵ existing)

$\delta([q_1,q_2],2)$ = ε-closure $(\delta(\hat{\delta}[q_1,q_2],\epsilon),2)$
= ε-closure $(\delta[q_1,q_2],2)$
= ε-closure $(\{\phi,q_2\})$

$\delta([q_1,q_2],2)$ = [q2] (∵ existing)

step 4:

$\delta([q_2],0)$ = ε-closure $(\delta(\hat{\delta}[q_2],\epsilon),0)$
= ε-closure $(\delta([q_2]),0)$
= ε-closure $(\{\phi\})$

$\delta([q_2],0)$ = [ϕ] (∵ existing)

$\delta([q_2],1)$ = ε-closure $(\delta(\hat{\delta}[q_2],\epsilon),1)$
= ε-closure $(\delta([q_2],1)$
= ε-closure $(\{\phi\})$

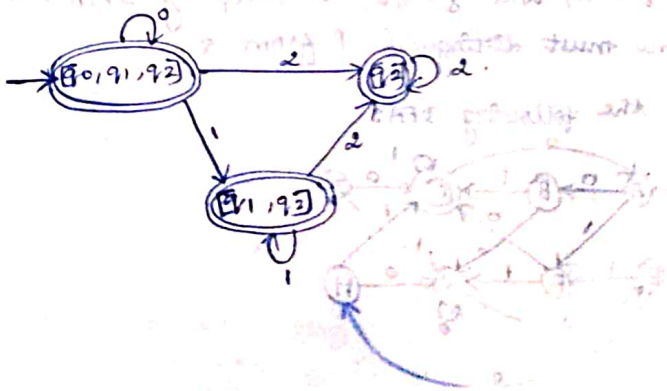
$\delta([q_2],1)$ = [ϕ] (∵ existing)

$\delta([q_2],2)$ = ε-closure $(\delta(\hat{\delta}[q_2],\epsilon),2)$
= ε-closure $(\delta[q_2],2)$
= ε-closure $(\{q_2\})$

$\delta([q_2],2)$ = [q2] (∵ existing)

step 5:

No further new equations, so draw state diagram.



EQUIVALENCE AND MINIMIZATION OF AUTOMATA:

Equivalence of states:

The states P and Q are equivalent if for all input strings w, $\hat{\delta}(P,w)$ is an accepting state if and only if $\hat{\delta}(Q,w)$ is an accepting state.

Non-equivalence state:

If two states are not-equivalent, then it is said to be distinguishable (i.e) state P is distinguishable from state Q. If there is atleast one string w such that $\hat{\delta}(P,w)$ and $\hat{\delta}(Q,w)$ accepting and the other is non-accepting.

TABLE FILLING ALGORITHM:

The algorithm is a recursive discovery of distinguishable pairs in a DFA, $M=\{Q,\Sigma,\delta,q_0,F\}$.

If P is an accepting state and Q is non accepting state then the pair P,Q is distinguishable states.

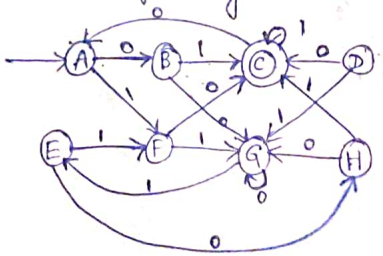
Let P and Q be states such that for some input symbol 'a'

$r=\delta(P,a)$

$s=\delta(Q,a)$ are a pair of states known to be distinguishable, then P,Q is a pair of distinguishable states.

The reason this rule makes sense is that there must be some string w that distinguishes r from s where exactly one of $\hat{\delta}(r,w)$ and $\hat{\delta}(s,w)$ is accepting then the string aw must distinguish P from Q.

1) Consider the following DFA:



Find the equivalent and distinguishable states and draw the minimized DFA.

Sol:

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

not x → x

i) Here, C is the only accepting state so put x in each pair involving C.

ii) Apply Table filling algorithm to find the equivalence and distinguishable states.

AB: $A \xrightarrow{0} B$, $A \xrightarrow{1} F$; $B \xrightarrow{0} G$, $B \xrightarrow{1} C$. Place x in AB since x is already in C, F

not equal B≠G; not equal F≠C; so, distinguishable

AD: $A \xrightarrow{0} B$, $A \xrightarrow{1} F$; $D \xrightarrow{0} C$, $D \xrightarrow{1} G$. Place x is AD since x is already in (B,C)

B≠C; F≠G

AE: $A \xrightarrow{0} B$, $A \xrightarrow{1} F$; $E \xrightarrow{0} H$, $E \xrightarrow{1} F$. Equivalent pair.

F=F

now check, BH is also equivalent if not then it is equivalence

Equivalent pair

AF: $A \xrightarrow{0} B$, $A \xrightarrow{1} F$; $F \xrightarrow{0} C$, $F \xrightarrow{1} G$. Place x is AF since x is already in (B,C)

B≠C; F≠G

AG:
A → B A → F
G → G G → E
B ≠ G distinguishable

AH:
A → B A → F
H → G H → C

BD:
B → G B → C
D → C D → G

BE:
B → G B → C
E → H E → F

BF:
B → G B → C
F → C F → G

BG:
B → G B → C
G → G G → E

BH:
B → G B → C
H → G H → C
Equivalent pair

DE:
D → C D → G
E → H E → F

DF:
D → C D → G
F → C F → G
Equivalent pair

Minimized DFA:

States	I/p 0	1
A, E	B, H	F, D
B, H	G	C
D, F	C	G
C	A, E	C
G	G	A, E

DG:
D → C D → G
G → G G → E

DH:
D → C D → G
H → G H → C

EF:
E → H E → F
F → C F → G

EG:
E → H E → F
G → G G → E

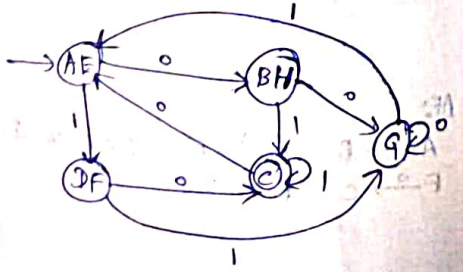
EH:
E → H E → F
H → G H → C

FG:
F → C F → G
G → G G → E

FH:
F → C F → G
H → G H → C

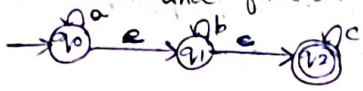
GH:
G → G G → E
H → G H → C

Equivalent states:
There are three equivalent states {A,E}, {B,H} and {D,F}
Distinguishable states: {C}, {G}



Finals

1) consider the NFA and find δ(q0, ab) is accepted or not



Soln:

ε closure of [q0] = {q0, q1, q2}
ε closure of [q1] = {q1, q2}
ε closure of [q2] = {q2}

$\hat{\delta}(q_0, \epsilon)$ = ε-closure(q0) = {q0, q1, q2}

$\delta(q_0, a)$ = ε-closure($\delta(q_0, a)$)
= ε-closure($\delta(\hat{\delta}(q_0, \epsilon), a)$)
= ε-closure($\delta(\{q_0, q_1, q_2\}, a)$)
= ε-closure(q0)

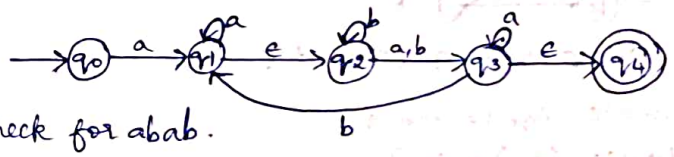
$\delta(q_0, a)$ = {q0, q1, q2}

$\hat{\delta}(q_0, ab)$ = ε-closure($\delta(q_0, ab)$)
= ε-closure($\delta(\delta(q_0, a), b)$)
= ε-closure($\delta(\{q_0, q_1, q_2\}, b)$)
= ε-closure($\phi \cup q_1 \cup \phi$)
= ε-closure{q1}

$\hat{\delta}(q_0, ab)$ = {q1, q2}

∴ Accepted.

2)



check for abab.

ε-closure[q0] = {q0}
ε-closure[q1] = {q1, q2}
ε-closure[q2] = {q2}
ε-closure[q3] = {q3, q4}
ε-closure[q4] = {q4}

$\hat{\delta}(q_0, \epsilon)$ = ε-closure[q0] = {q0}

$\hat{\delta}(q_0, a)$ = ε-closure($\delta(q_0, a)$)
= ε-closure($\delta(\hat{\delta}(q_0, \epsilon), a)$)
= ε-closure($\delta(\{q_0\}, a)$)
= ε-closure({q1, q2})

$\hat{\delta}(q_0, a)$ = {q1, q2}

$\hat{\delta}(q_0, ab) = \epsilon\text{-closure}(\delta(\hat{\delta}(q_0, a), b))$
$= \epsilon\text{-closure}(\delta(\delta(q_0, a)), b)$
$= \epsilon\text{-closure}(\delta(q_1, q_2), b)$
$= \epsilon\text{-closure}(\{q_2, q_3\})$

$\delta(q_0, ab) = \{q_2, q_3, q_4\}$

$\hat{\delta}(q_0, aba) = \epsilon\text{-closure}(\delta(q_0, aba))$
$= \epsilon\text{-closure}(\hat{\delta}(\delta(q_0, a), ba))$
$= \epsilon\text{-closure}(\hat{\delta}(q_1, q_2), b), a)$
$= \epsilon\text{-closure}(\hat{\delta}(q_2, q_3), a))$
$= \epsilon\text{-closure}\{q_3\}$

$\hat{\delta}(q_0, aba) = \{q_3, q_4\}$

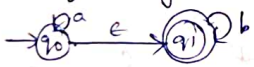
$\delta(q_0, abab) = \epsilon\text{-closure}(\delta(q_0, a), bab))$
$= \epsilon\text{-closure}(\delta(\hat{\delta}(q_1, q_2), b), ab)$
$= \epsilon\text{-closure}(\delta(\{q_2, q_3\}, a), b)$
$= \epsilon\text{-closure}(\delta(\hat{\delta}(q_3), b)$
$= \epsilon\text{-closure}\{q_1\}$

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

∴ Not Accepted.

NFA WITH ε to NFA without ε:

1) convert the following NFA without ε.



NFA with ε to NFA without ε.

$\epsilon\text{-closure}\{q_0\} = \{q_0, q_1\}$
$\epsilon\text{-closure}\{q_1\} = \{q_1\}$

$\delta(q_0, a) = \epsilon\text{-closure}(\delta(\hat{\delta}(q_0, \epsilon), a))$
$= \epsilon\text{-closure}(\{q_0, q_1\}, a)$
$= \epsilon\text{-closure}(\{q_0\} \cup \phi\})$
$= \epsilon\text{-closure}\{q_0\}$

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

$\delta(q_0, b) = \epsilon\text{-closure}(\delta(\hat{\delta}(q_0, \epsilon), b))$
$= \epsilon\text{-closure}(\{q_0, q_1\}, b)$
$= \epsilon\text{-closure}\{\phi \cup q_1\}$
$= \epsilon\text{-closure}\{q_1\}$

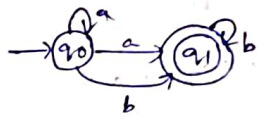
$\delta(q_0, b) = \{q_1\}$

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

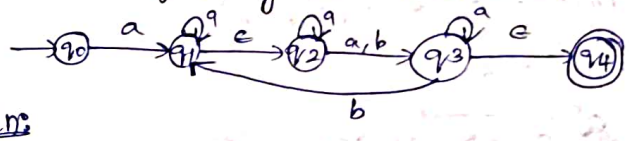
$\delta(q_1, a) = \{\phi\}$

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

$\delta(q_1, b) = \{q_1\}$



2) convert the following NFA without ε.



soln:

$\epsilon\text{-closure}\{q_0\} = \{q_0\}$
$\epsilon\text{-closure}\{q_1\} = \{q_1, q_2\}$
$\epsilon\text{-closure}\{q_2\} = \{q_2\}$
$\epsilon\text{-closure}\{q_3\} = \{q_3, q_4\}$
$\epsilon\text{-closure}\{q_4\} = \{q_4\}$

$\delta(q_0, a) = \epsilon\text{-closure}\{\delta(\hat{\delta}(q_0, \epsilon), a))$
$= \epsilon\text{-closure}(\{q_0\}, a)$
$= \epsilon\text{-closure}\{q_1\}$

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

$\delta(q_0, b) = \epsilon\text{-closure}(\{q_0\}, b)$
$= \epsilon\text{-closure}\{\phi\}$

$\delta(q_0, b) = \phi$

$\delta(q_1, a) = \epsilon\text{-closure}(\{q_1, q_2\}, a)$
$= \epsilon\text{-closure}\{q_1, q_2, q_3\}$

$\delta(q_1, a) = \{q_1, q_2, q_3, q_4\}$

$\delta(q_1, b) = \epsilon\text{-closure}(\{q_1, q_3\}, b)$
$= \epsilon\text{-closure}\{q_3\}$

$\delta(q_1, b) = \{q_3, q_4\}$

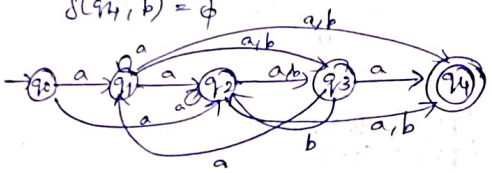
$\delta(q_2, a) = \epsilon\text{-closure}(\{q_2\}, a)$
$= \epsilon\text{-closure}\{q_2, q_3\}$

$\delta(q_2, a) = \{q_2, q_3, q_4\}$

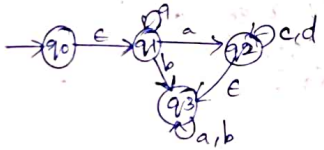
$\delta(q_2,b) = \epsilon\text{-closure}(\dots$
$= \epsilon\text{-closure}\{q_3\}$

$\delta(q_2,b) = \{q_3,q_4\}$
$\delta(q_3,a) = \epsilon\text{-closure}(\{q_3,q_4\},a)$
$= \epsilon$ closure $(\{q_{33},\phi\}$

$\delta(q_3,a) = \{q_3,q_4\}$
$\delta(q_3,b) = \epsilon$ closure $(\{q_3,q_4,b)$
$= \epsilon\text{-closure}\{q_1\}$
$\delta(q_3,b) = \{q_1,q_2\}$
$\delta(q_4,a) = \epsilon\text{-closure}(\{q_4\},a)$
$= \phi$
$\delta(q_4,a) = \phi$
$\delta(q_4,b) = \epsilon\text{-closure}(\{q_4\},b)$
$= \phi$
$\delta(q_4,b) = \phi$



H.W NFA with ϵ to DFA



$\epsilon\text{-closure}\{q_0\} = \{q_0,q_1\}$
$\epsilon\text{-closure}\{q_1\} = \{q_1\}$
$\epsilon\text{-closure}\{q_2\} = \{q_2,q_3\}$
$\epsilon\text{-closure}\{q_3\} = \{q_3\}$

step1 : Initial state of DFA : $\epsilon\text{-closure}[q_0] = [q_0,q_1] \rightarrow (1)$

step2:

$\delta([q_0,q_1],a) = \epsilon\text{-closure}(\delta(\hat{\delta}([q_0,q_1],\epsilon),a)$
$= \epsilon\text{-closure}(\delta([q_0,q_1],a))$
$\delta([q_0,q_1],a) = \epsilon\text{-closure}([q_1,q_2])$
$\delta([q_0,q_1],a) = [q_1,q_2,q_3] \rightarrow (2)$

$\delta([q_0,q_1],b) = \epsilon\text{-closure}(\delta(\hat{\delta}([q_0,q_1],\epsilon),b)$
$= \epsilon\text{-closure}(\delta([q_0,q_1],b))$
$\delta([q_0,q_1],b) = \epsilon\text{-closure}([q_3])$
$\delta([q_0,q_1],b) = [q_3] \rightarrow (3)$

$\delta([q_0,q_1],c) = \epsilon\text{-closure}(\delta(\hat{\delta}[q_0,q_1],\epsilon),c)$
$= \epsilon\text{-closure}(\delta[q_0,q_1],c))$
$= \epsilon\text{-closure}([\phi])$
$= \phi$
$\delta([q_0,q_1],d) = \epsilon\text{-closure}(\delta[q_0,q_1],d) = \phi$

step 3:

$\delta([q_1,q_2,q_3],a) = \epsilon\text{-closure}([q_1,q_2,q_3],a)$
$= \epsilon\text{-closure}([q_1,q_2,q_3])$
$= [q_1,q_2,q_3]$

$\delta([q_1,q_2,q_3],b) = \hat{\delta}(\delta([q_1,q_2,q_3],\epsilon),b)$
$= \hat{\delta}([q_1,q_2,q_3],b)$
$= \epsilon\text{-closure}([q_3])$
$= [q_3]$

$\delta([q_1,q_2,q_3],c) = \delta(\hat{\delta}([q_1,q_2,q_3],\epsilon),c)$
$= \epsilon\text{-closure}(\delta[q_1,q_2,q_3],c)$
$= \epsilon\text{-closure}([q_2])$
$= [q_2,q_3] \rightarrow (4)$

$\delta([q_1,q_2,q_3],d) = \epsilon\text{-closure}(\delta(\delta([q_1,q_2,q_3],\epsilon),d)$
$= \epsilon$ closure $(\delta[q_1,q_2,q_3],d)$
$= \epsilon$ closure $([q_2])$
$= [q_2,q_3]$

step 4:

$\delta([q_3],a) = \epsilon$ closure $(\delta(\hat{\delta}[q_3],\epsilon),a)$
$= \epsilon$ closure $(\delta([q_3],a)$
$= \epsilon$ closure $([q_3])$
$= [q_3]$

$\delta([q_3],b) = \epsilon$ closure $(\delta([q_2],b)$
$= [q_3]$

$\delta([q_3],c) = \epsilon$ closure $[\delta[q_2],c]$
$= \phi$

$\delta([q_3],d) = \epsilon$ closure $(\delta[q_3],d)$
$= \phi$

step 5:

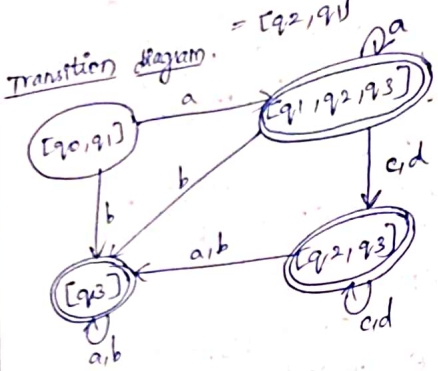
$\delta([q_2,q_3],a) = \epsilon$ closure $(\delta(q_2,q_3),a)$
$= \epsilon$ closure $([q_3])$
$= [q_3]$

$\delta[q_2,q_3,b) = \epsilon$ closure $(\delta(q_2,q_3),b)$
$= \epsilon$ closure $([q_3])$
$= [q_3]$

$\delta([q_2,q_3],c) = \epsilon$ closure $(\delta(q_2,c)) \cup \delta(q_3,c)$
$= \epsilon$ closure (q_2,q_3)
$= [q_2,q_3]$

$\delta[q_2,q_3],d) = \epsilon$ closure $(\delta[q_2,q_3),d)$
$= \epsilon$ closure (q_2,q_3)
$= [q_2,q_3]$

Transition diagram.

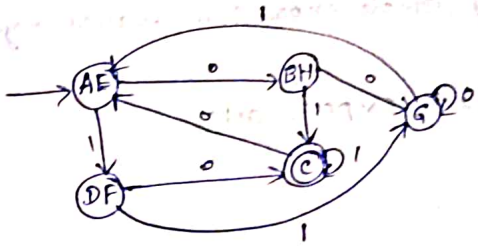


Transition table

$\delta(A,0) = B$	$\delta(A,1) = F$
$\delta(E,0) = H$	$\delta(E,1) = F$
$\delta(G,0) = G$	$\delta(G,1) = E$

Step 5: on considering the input 0 and 1 the states A and E goes to another group BH and F. Therefore,

$\Pi new = (AE)(BH)(DF)(G)(C)$



29/6/23 Thursday

GROUPING METHOD: (for minimization)

Step 1: Initially the states are divided into two groups, final and non-final.

Final	Non-final
C	ABDEFGH

Step 2: To find the Πnew group, the group (ABDEFGH) is checked on transition 0 and 1

$\delta(A,0) = B$	$\delta(A,1) = F$
$\delta(B,0) = G$	$\delta(B,1) = C$
$\delta(D,0) = C$	$\delta(D,1) = G$
$\delta(E,0) = H$	$\delta(E,1) = F$
$\delta(F,0) = C$	$\delta(F,1) = G$
$\delta(G,0) = G$	$\delta(G,1) = E$
$\delta(H,0) = G$	$\delta(H,1) = C$

Step 3: on considering the input 0, the states D and F goes to another group C. Therefore, (ABEGH) (DF) (C)

Step 4: on considering the input 1, the states B and H goes to another group C. Therefore,

$\Pi new = (AEG)(HB)(DF)(C)$