13/10/16

Unit - 5

V-Imp Questions :-

① Chomsky Hierarchy :- (classes of languges)

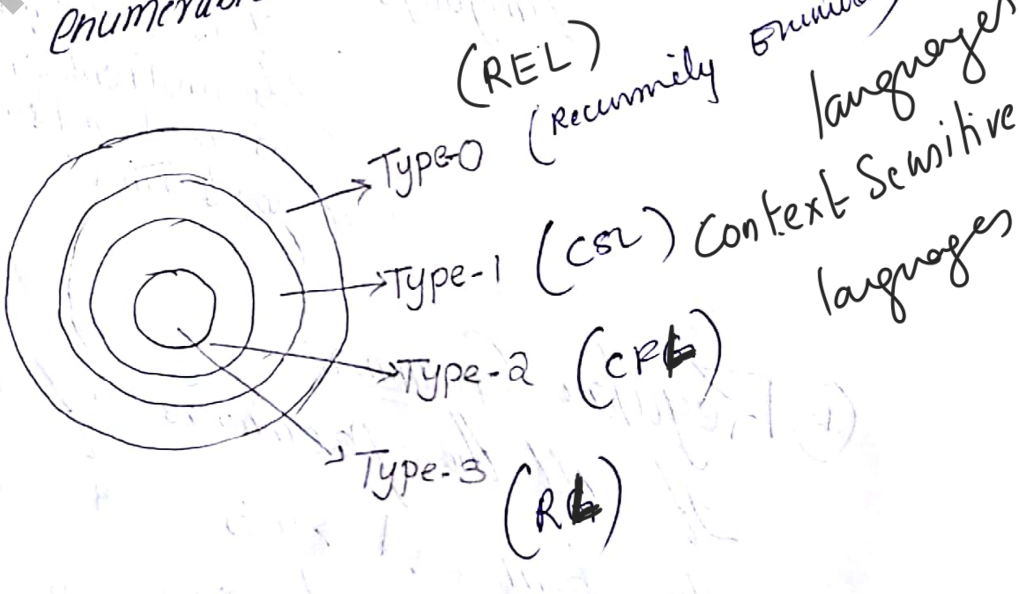
The theorem states that except for the empty string type-i languages properly include type (i+1) languages for i=0,1,2.

So There are 4 classes of languages

1. Recursively Enumerable Language (type-0)
2. Context Sensitive Language (CSL) (type-1)
3. Context Free language (CFL) (type-2)
4. Regular Language (type-3)

The regular sets are properly contained in CFL's. The CFL's are contained in CSL's. The CSL's are contained in recursively enumerable set

* Type i includes type (i+1) for i = 0, 1, 2.



Chomsky hierarchy table:-

Type of language	Name	Production $X \rightarrow Y$	Acceptor or Automaton
type-0	Recursively enumerable language	X is any string with nonterminals & Y is any string.	Turing Machine
type-1	Context Sensitive Language	X is any string with non-terminals Y is any string & longer than X or equal	Linear Bounded Automaton
type-2	Context Free Language	X is the single non-terminal Y is any string	Push down Automata
type-3	Regular Language	X is the single non-terminal Y is of the form terminal nonterminal (or) non-terminal terminal (or) terminal	Finite Automata.

① Regular Grammar.

If all the productions of the grammar are of the form $A \rightarrow wB$ or $A \rightarrow w$ where A & B are non-terminals & w is a string of terminals then we say the grammar is right linear

If the productions are of the form $A \to Bw$ & $A \to w$ where A & B are nonterminals w is a string of terminals then we say the grammar is left linear.

A language is said to be regular if & only if it has a left linear grammar & a right linear grammar.

ex:-

FA:- Write few lines with diag.

(2) CFG :- Def: CFGs defines CFLS.

ex:-

PDA:- Write few lines & diag.

(3) CSL :- For the productions $\alpha \to \beta$ where α & β are strings of grammar symbols with $\alpha \neq \epsilon$. If β is atleast as long as or longer than α then the resulting grammar is called CSG represented by an LBA

ex:-

LBA:- write few lines with diag.

(4) Recursively Enumerable Grammars

The largest family of grammars where productions are of the form $\alpha \to \beta$ where α & β are strings of grammar symbols $\alpha \neq \epsilon$ then the grammar is called

REG & semi-thue grammars & phrase structured grammars & unrestricted grammar represented by Turing machine.

ex::

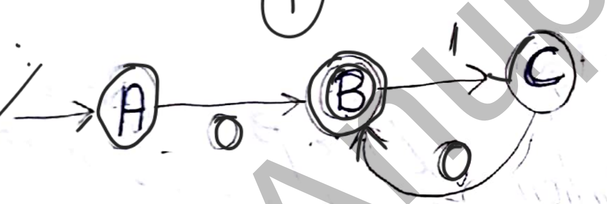
TM:- write few lines & diag.

---

① Convert the following into right linear and then into left linear

① $0(10)^*$ Min string is 0.

①

FA 

②

Right linear:- (RLG)

$NT \rightarrow tNT / t$

$A \rightarrow 0B / 0$

$B \rightarrow 1C$

$C \rightarrow 0B / 0$

RLG

$A \rightarrow 0B / 0$

$B \rightarrow 1A$

LLG

$A \rightarrow B0 / 0$

$B \rightarrow A1$

To get a left linear grammar, we reverse the right linear grammar.

(LLG) $A \rightarrow B0 / 0$

$B \rightarrow C1$

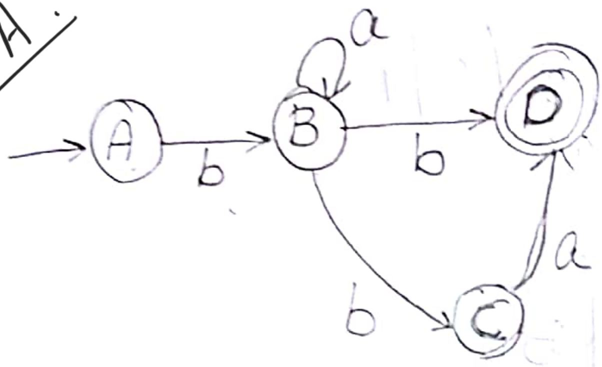
$C \rightarrow B0 / 0$

② $ba^*(b+ba)$

assume a^* is ϵ, then

min string: $bb + bba$

FA:



else

$ba^*b +$

ba^*ba

R(G)

$A \to bB$

$B \to aB \mid bC \mid bD \mid b$

$C \to aD \mid a$

$D \to \epsilon$

LLG:

$A \to Bb$

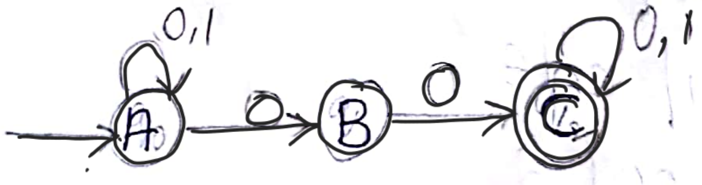
$B \to Ba \mid Cb \mid Db \mid b$

$C \to Da \mid a$

$D \to \epsilon$

③ $(0+1)^*00(0+1)^*$

FA:



RLG:

$A \rightarrow 0A \mid 1A \mid 0B$

$B \rightarrow 0C \mid 0$

$C \rightarrow 0C \mid 1C \mid 0 \mid 1$

LLG:-

$A \rightarrow A0 \mid A1 \mid B0$

$B \rightarrow C0 \mid 0$

$C \rightarrow C0 \mid C1 \mid 0 \mid 1$

Without FA:-

4) $((((01+10)^*11)^*00)^*$

RLG

$S \rightarrow 01S \mid 10S \mid B \mid \epsilon$

$B \rightarrow 11B \mid C \mid \epsilon$

$C \rightarrow 00C \mid \epsilon$

LLG:

$S \rightarrow S10 \mid S01 \mid B \mid \epsilon$

$B \rightarrow B11 \mid C \mid \epsilon$

$C \rightarrow C00 \mid \epsilon$

(5) $10 + (0+11)0^*1$

RLG: $S \rightarrow 10 \mid B$

$B \rightarrow 0C \mid 11C$

$C \rightarrow 0C \mid 1$

LLG:-

$S \rightarrow 01 \mid B$

$B \rightarrow C0 \mid C11$

$C \rightarrow C0 \mid 1$

(6) $0^*(1(0+1))^*$

RLG: $S \rightarrow 0S \mid B \mid \epsilon$

$B \rightarrow 10B \mid 11B \mid \epsilon$

LLG:

$S \rightarrow S0 \mid B \mid \epsilon$

$B \rightarrow B01 \mid B11 \mid \epsilon$

Recursively enumerable Languages & their Properties:

A language L is said to be recursively enumerable if there is a turing machine that accepts L. and is recursive if there is a TM that recognizes L (TMs halt regardless of whether or not they accept — recursive)

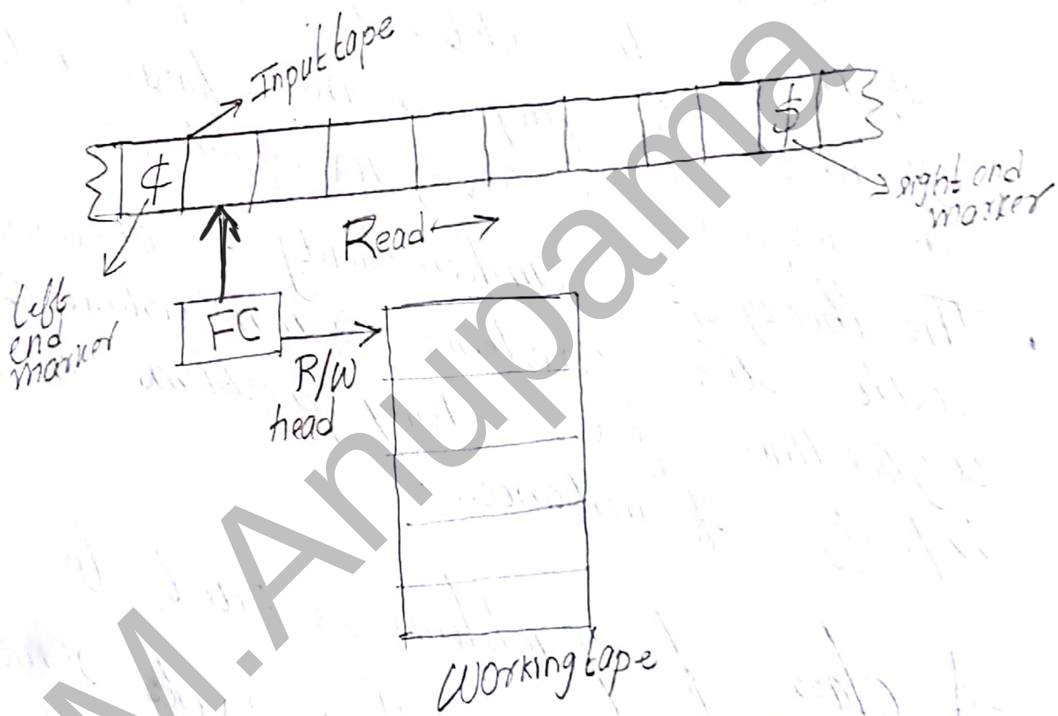
Properties:

1. The Complement of a recursive language is recursive.
2. The Union of two recursive languages is recursive. The union of two recursively enumerable language is recursively enumerable.
3. If the language L & its compliment L' are both recursively enumerable then L & L' are recursive.

Linear Bounded Automata

LBA is an automata which is a non deterministic TM & defines context sensitive languages. These languages lie b/w Recursively Enumerable languages & ~~Con~~ CFG. There is a single tape whose length is bounded by a linear function which is the length of the input string. Hence it is called linear bounded. The tape is restricted by the portion containing input x bounded by left end marker '¢' & '\$' is the right end marker. Other parameters are same as the TM. Hence it is given by

$$M = (Q, \Sigma, \Gamma, \delta, q_0, B, ¢, \$, F)$$



- Input tape can only be read only & tape head never prints on the tape. There is an additional working tape which performs both read write operations.

17/10/16

Undecidability:-

There are some types of problems that cannot be solved with the computer. Such problems are called undecidable problems. Problems that a computer cannot solve are as follows:-

1. Positive Integer solutions to the equation

$$x^n + y^n = z^n$$

If $n = 2$ the solution is 3, 4, 5

If $n > 2$ the programs never finds a triple of +ve integers to satisfy the equation.

- The theory of undecidability is concerned with the existence or non-existence of the algorithms for solving problems with an infinity of instances.

- A class of problems is said to be decidable if their exists some definite algorithm which always terminates with a outputs (Yes or No).

- It is undecidable or unsolvable if there is no algorithm that takes as input & instance of the problem & determines whether the answer is yes or no.

Eg:- examples of undecidable problem

Eg:- The halting problem of TM's is unsolvable.

① Whether a given program can loop forever on some input.

② Whether a turing machine halts on all inputs.

③ Given 2 programs & an input do the programs produce the same output for the given input.

Universal Turing Machine :-

~~The~~ UTM It is a general purpose TM which accepts 2 inputs.

1. Input data
2. A description of the computation (algorithm)

- This is more powerful than a TM & is designed by making certain modifications to the arbitrary TM. It is used to show that a particular problem is undecidable.
- The language accepted by a UTM is denoted as Lu (Universal language)

Modifications:-

1. It has more than one read write heads.
2. ~~Tape~~ Tape is 2 or 3 dimensions.

3. It has special purpose memory such as the stack or special purpose register.
4. These modifications increase the speed of operations.

→ The UTM is a multi tape TM where the transactions of M are stored initially on first tape along with the input string W.

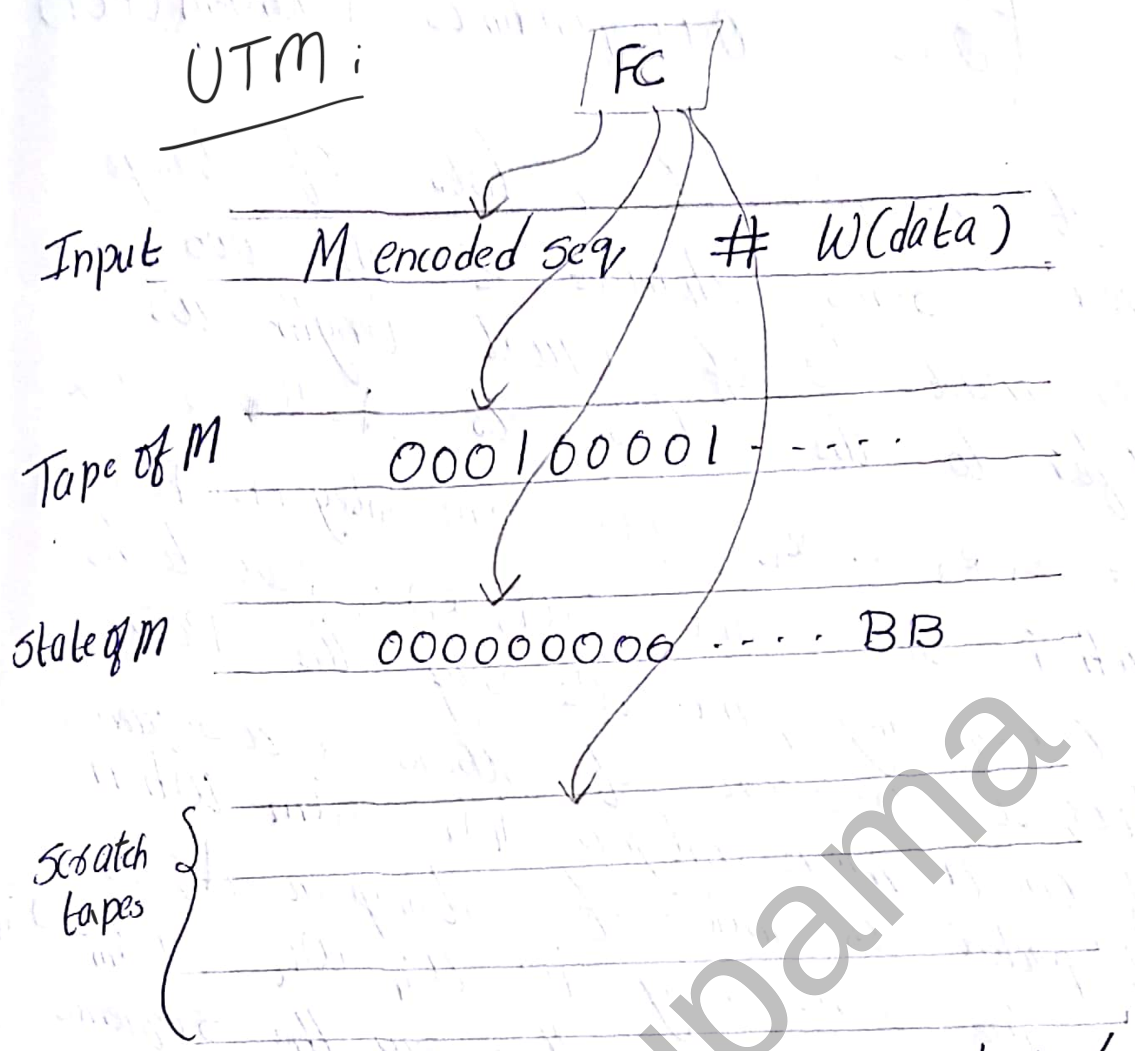
→ The second tape is used to hold simulated tape of M. i.e the tape symbol X_i of M will be represented by 0^i seperated by 1's.

→ The third tape holds the state of TM with q_i represented by i 0's etc.

→ The i/p is a string composed of 2 parts i.e the encoded program of TM followed by a special marked & followed by a string called data.

→ The UTM operates on the data depending on the encoding ~~scream~~ scheme used.

UTM :



Questions

1. How does the concept of UTM help to ~~helps to show~~ show the problem is Undecidable.

Ans: The TM ~~Quotes~~ codes are generated in the beginning with restricted alphabets & the universal language is Constructed. The UTM is then Constructed which accepts this Universal language. At first, we assume an algorithm for Universal language exists. This assumption taken is finally used to prove that no algorithm exists for Universal language. Hence a UTM helps in showing that the problem is undecidable.

Post Correspondance Problem (PCP)

It Consists of 2 lists of strings over some alphabet Σ. The two lists must be of equal length. We refer to the lists as $A = w_1 w_2 \dots w_k$ $B = x_1 x_2 \dots x_k$ for some integer k. For each i the pair (w_i, x_i) is said to be a Corresponding pair. We say the PCP has a solution if there is a sequence of one or more integers $i_1 i_2 \dots i_m$ when interpreted as indexes of strings in A & B is the same string i.e $w_{i_1} w_{i_2} \dots w_{i_m} = x_{i_1} x_{i_2} \dots x_{i_m}$ we then say the sequence $i_1\, i_2 \dots i_m$ is solution of PCP.

(1) Does the given PCP has a solution

(1) $X = (b, bab^2, ba)$ (list 1) length = 3

$Y = (b^2, ba, ba)$ (list 2) length = 3

Let $M = 4$ $i_1 = 2$ $i_2 = 1$ $i_3 = 1$ & $i_4 = 3$

Applying this sequence to list Sequence X

$X = \underbrace{bab^2}_{2}\,\underbrace{b}_{1}\,\underbrace{b}_{1}\,\underbrace{ba}_{3}$ } Seq is: (2,1,1,3)

Applying the same sequence to list Y

$$Y = \underset{2}{\underline{bab}}\ \underset{1}{\underline{b^2}}\ \underset{1}{\underline{b^2}}\ \underset{3}{\underline{ba}} \quad \text{Seq is } (2,1,1,3)$$

Since both the strings generated are the same the PCP has a solution given by (2,1,1,3). Try and solve with some sequences and finally arrive at the correct sequence

② $X = (\overset{1}{ba}, \overset{2}{abb}, \overset{3}{bab})$

$Y = (\overset{1}{bab}, \overset{2}{bb}, \overset{3}{abb})$

Let ~~M=~~ $i_1 = 1$ $i_2 = 3$ $i_3 = 3$ (try some attempts & show that the solution is not possible.)

no solution

③

	list A	list B
	A	B
i	w_i	x_i
1	1	111
2	10111	10
3	10	0

$\simeq$ $A = (\overset{1}{1}, \overset{2}{10111}, \overset{3}{10})$

$B = (111, 10, 0)$

$M = 4$ $i_1 = 2$ $i_2 = 1$ $i_3 = 1$, $i_4 = 3$

$A = \underline{10111}\,\underline{1}\,\underline{1}\,\underline{10}$

$B = \underline{10}\,\underline{111}\,\underline{111}\,\underline{0}$

the PCP has a solution

(2, 1, 1, 3)

(4)

	A	B
i	w_i	x_i
1	10	101
2	011	11
3	101	011

$A = (\overset{1}{10}, \overset{2}{011}, \overset{3}{101})$

$B = (\overset{1}{101}, \overset{2}{11}, \overset{3}{011})$

No solution

(5) $X = (01, 1, 1)$ $Y = (01^2, 10, 1^2)$

Every string of X is less than the correspondance string of Y

So we never get a sequence.

(PTO to the last ~~pages~~ for more probs) solved on PCP

Imp Turing Machine as Enumerator

Ans TM is used to enumerate a set. i.e it lists out the elements or the O/p's of strings of a language one at a time. i.e there is an algorithm for the

TM to act as an enumerator.

ex:- $(a+b)^* aa (a+b)^*$

enumerations are : 1. ab aa ab

2. a aa ab

3. aa and so on.

~~Q8~~ Halting Problem of TM

Ans:- We divide the class of problems that can be solved by a TM as follows.

1. Those problems that have a TM which halts whether or not it accepts the i/p.

2. Those problems that are only solved by TM's. However TM's may not halt at all if they donot accept the i/p.

The statement 2 raises the fundamental issue "Whether the TM 'T' applied to input x halts or doesnot halt". This is known as halting problem of TM.

* (Add Answer of reasons for TM for not accepting i/p given in TM notes.)

This problem is undecidable i.e there exists no algorithm that can correctly answer this question.

③ Diagonalization Language (2m Bit)

Read from TB

pg-346

pg-5.8

④ MPCP:- (Modified PCP)

PCP is undecidable like halting problem of TM. A modified PCP is designed which has two lists A & B similar to a PCP but the solution requires the sequence $i_2, i_3, \ldots i_k$ with a match starting with the first domino.

1st one should 1st string of list A and B.

ex:- A = (ab, ba, b, abb, a) — ① ② ③ ④ ⑤

B = (aba, abb, ab, b, bab) — ① ② ③ ④ ⑤

Seq is : (5, 2, 3, 4, 4, 3, 4)

When we apply the sequence to both the lists we apply it has follows

A = $\underline{ab}\ \underset{5}{a}\ \underset{2}{ba}\ \underset{3}{b}\ \underset{4}{abb}\ \underset{4}{abb}\ \underset{3}{b}\ \underset{4}{abb}$

B = $\underline{aba}\ \underset{5}{bab}\ \underset{2}{abb}\ \underset{3}{ab}\ \underset{4}{b}\ \underset{4}{b}\ \underset{3}{ab}\ \underset{4}{b}$

1st domino of A is ab and B is aba; Applying the seq to A and B with the respective 1st dominos, the resulting strings are same. Hence the MPCP has a solution.

5 Rice Theorem :-

Statement :-

Let P be any non-trivial property of regular expression languages then P is undecidable. It is an example of unsolvable problems & by using this theorem we can determine ~~the prop~~ a property of RE is decidable & undecidable.

The following properties of RE's like Context Freeness, Finiteness, emptiness & regularity are determined to be undecidable.

The Rice theorem is not applicable if the property is trivial. (decidable) trivial means the Decision problems are answer is either yes or no.

non-trivial: means "maybe".

6 2 examples of recursively enumerable language.

ex: for $a^{2n}b^n$ $n \geq 1$

For a palindrome

7. 2 examples for unrestricted grammar.

The grammar generates the language of strings over ~~(ab)*~~ $\{a, b\}^*$ with equal no: of a's & b's.

ex:- ① $S \rightarrow aBS/\epsilon$

$aB \rightarrow Ba$

$Ba \rightarrow aB$

$B \to b$

(i) $S \to aBS \to aB \to Ba \to \underline{ba}$

$\quad\quad\quad \downarrow$

$\quad\quad\quad \epsilon$

② $S \to aB$

$aB \to aaB$

$aB \to a$

$\to$ (ii) $S \to aBS \to aBaBS \to$

$aBaB \to aBBa \to abba$

ϵ

(Satisfiability problem)

~~Satisfy~~

⑦ SAT problem:-

Show that it is NP (non polynomial)

* { Given a boolean expression $f(x_1, x_2 \ldots x_n)$ involving logical ~~problems~~ variables, ~~thus~~ does there exist an assignment of either true or false values to each of these variables such that f is true?

We call the boolean expression f to be satisfiable if the above statement holds good.

Sol:- Consider the boolean expression $f(x_1, x_2, x_3)$ which is given as follows

$$f(x_1, x_2, x_3) = (\bar{x}_1 \vee x_2) \wedge (x_1 \vee x_3)$$

We can find the existence of an assignment to each of the variable such as $x_1 =$ false $x_2 =$ true $x_3 =$ true

$$f(x_1, x_2, x_3) = (\overline{\text{false}} \vee \text{true}) \wedge (\text{false} \vee \text{true}) = \text{true}$$

For the above assignment the answer is true. Hence we say the given expression is satisfiable. If it is false it is not satisfiable.

The SAT problem is clearly NP because we can guess a truth assignment & verify that it satisfies boolean exp in polynomial time. SAT is NP complete is given by Cook's theorm.

---

⑧ RSAT: (Restricted SAT)

It is also called 3 SAT. It is much easier than SAT but the expression has regular form. They are the 'and' of clauses each of which is the Or of exactly 3 variables. We need to convert each expr. in a SAT into the above format for a RSAT problem.

---

⑨ P & NP proplems or (Theory of Intractability)

A problem is said to be solvable if it has an algorithm to solve it.
Problems are categorized into 2 groups depending on time taken for their execution.

(a) problems whose solution times are bounded by polynomials ex: Bubble sort
$P(n) = n^2 - 2n + 1$

(b) Problems whose best known algorithms are non polynomial

ex:: travelling sales men

The complexity of this is $O(n^2 2^n)$

i.e exponential.

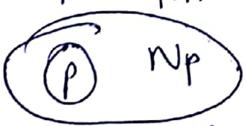
These problems require large amounts of time to execute & hence are difficult to solve.

Problems of 1st kind are called tractable or easy problems & problems of 2nd kind are called Intractable or hard ~~difficult~~ problems.

(P-problem, np-problem, np-Complete & np-hard problems) ⟶ short notes.

P– problem: P means deterministic polynomial time. A language L is said to be in class P if there exists a deterministic TM M such that M is of time complexity P(n) for some polynomial P and M accepts L. This class of problems are solvable in polynomial time. P– problems are usually decision problems. All the problems in the P-class are tractable. Eg: Kruskal's Algo

NP– problem: NP stands for Non deterministic polynomial time. The class of NP consists of those problems that are verifiable in polynomial time. A language L is in class NP if there is a Nondet. TM such that M is of time complexity P(n) for some polynomial P and M accepts L. Eg: 4 color problem, Hamiltonian ~~Circuit~~ Circuit problem etc., Travelling Salesman problem,

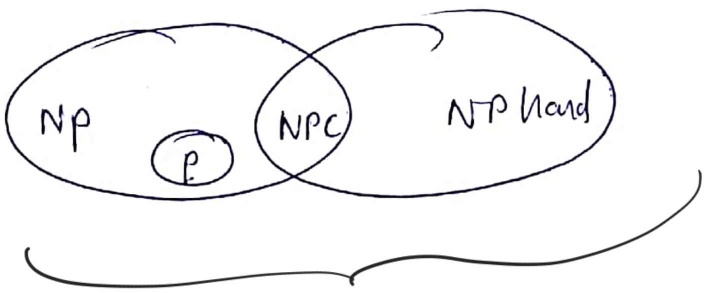


$P \subseteq NP$

Any problem in P is also in NP but it is not yet known that P = NP. NP problems are classified into 2 types. NP-complete & NP Hard

NP Complete: A class of problems is known as NP complete problems whose status is unknown. No polynomial time has yet been discovered for NP complete problems nor has anyone been able to prove that no polynomial time exists for any of them. These are the hardest of NP problems. Eg: Satisfiability problem. (SAT)

NP-Hard: These has computational times which are exponential or even worse than that. These are at least as hard as the NP-Complete. or harder.



4) Diagonalization:

The process of complementing the diagonal to construct the characteristic vector

---

PCP ⑥ $X = (11, 100, 111)$

$Y = (111, 001, 11)$

let $m = 3$ $i_1 = 1,\ i_2 = 2,\ i_3 = 3$

Seq: $(1, 2, 3)$ or $(1,3,1,3)$

$\cancel{X_1}\cancel{X_2}X_3 = Y_1Y_2Y_3 =$

11100111

11100111 ✓

---

⑦ $X = (110, 0011, 0110)$

$Y = (11\,0110, 00, 110)$

Seq: $(2, 3, 1)$ ~~$X_i = Y_i$~~

$0011\,0110\,110 = 00\,110\,110\,110$

$\therefore\ X_2X_3X_1 = Y_2Y_3Y_1$

(8) $X_i = (011, 11, 1101)$

$Y_i = (101, 011, 110)$

No solution.

$i_1 = 1$ is ruled out $i_1 = 2$ is ruled out

if $i_1 = 3$, then $i_2 = 1$ is possible ----

(9)

$X_i = (1, 110, 0)$

$Y_i = (10, 0, 11)$

MPCP is

X_1, X_{i_1}, X_{i_2} ----

Y_1, Y_{i_1}, Y_{i_2} ----

Seq: (1, 3, 2)

$\underset{1}{\underline{1}}\ \underset{3}{\underline{0}}\ \underset{2}{\underline{110}}$ and $\underset{1}{\underline{10}}, \underset{3}{\underline{11}}, \underset{2}{0}$