ed layers
so, fully
features

ter Convol-

Word2vec

* CBOW – Continuous bag of words: finds target from Context
* Skip gram – find context from target.

* MLE for Normal distribution:-

$$\mu = \frac{(x_1 + x_2 + x_3 + x_4 + \cdots + x_n)}{n}$$

where $x_1, x_2, \ldots x_n$ are data points
n = total data points

$$\sigma = \sqrt{\frac{(x_1-\mu)^2 + \cdots + (x_n-\mu)^2}{n}}$$

LNM 19/8/24

Operant Conditioning :- when the agent action leads to outcome.
outcome → something the agent wish to happen

* what is reinforcer?:-
⇒ Negative reinforcers can diassociate between response and outcome (ie. outcome may be different than response)

→ Cost function
↓
ck propagation

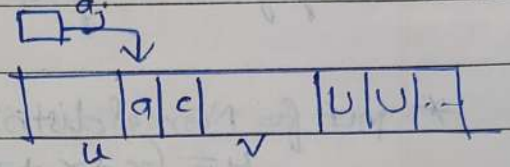
MCT 20/8/24

Turing Machine:
* Unrestricted memory (Tape size)
* Head can both read or write
* Tape is read-write.
* Output of TM can be accept/reject/looping
* Positions of head movement :- L, R, Same position (S)
* Configuration of TM :- < q, tape content, head location >
current state

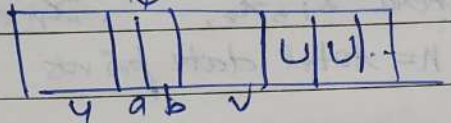
Date
Page

* Configuration of TM:- $u\ q\ v$ (Head pointing to first char of v)
  (u, v: strings; q: state)

* $C_2 = uq_j acv$



$C_1 = uaq_i bv$



$\delta(q_i, b) = (q_j, c, L)$ moving in 1 step.

* Start Configuration of TM:- $q_0 w$ TM
* Turing decidable:- A TM is decidable if it gives output for that input & don't loop.

Design a DTM, say M, that decides an input string $w = a^i b^j c^k$ such that $i*j=k$ and $i, j, k \geq 1$

Ex:- $i=2, j=3, k = i*j = 2\times3 = 6$

Tape :- [aa bbb cccccc U U U ...]
(blank symbols)

M = "On input string w:
1. Scan the input from left to right in the tape to determine whether it is a member of $a^+b^+c^+$ and "reject" iff it is not so.
2. Return the head to the left hand end of the tape
3. Cross off an "a" and scan to the right until a "b" occurs
4. Shuffle between b's and c's crossing of each other until all b's are gone. If all c's have been crossed-off and some b's remain, "reject".

Bookbird
Date
Page

5. Restore the crossed-off b's and repeat stage 3 and 4 if there is another a to cross-off. If "yes", 'accept', otherwise, "reject".

EST 22/8/24

* LST - Land Surface Temprature
* LULC - Land used Land Cover
  - land used for agriculture
  - land covered naturally on surface of earth
* Carbon is absorbed by vegetation and water bodies. When vegetation is degraded, carbon remains in

Azolla effect
in antartica region.
Azolla → doubles in mass 2-3 days, takes more CO_2, lives in fresh water → leads to intolerant freezing → After death it settles in ocean & converts to Peat. As it does not get decomposed
Peat → vegetation which does not degrade.
Polar motion:- Shift in earth axis & poles, because of mass distribution in & on the planet.

Quiz 1 → 2, 3, 4

LNM 22/8/24 Predicting reward

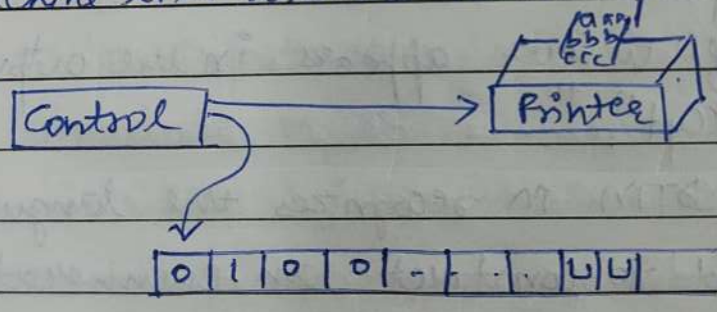
* Ventral Tegmental are :- Contains dopamine.

MCT 23/8/24

Non-deterministic Turing Machines (NTM)

$\delta : Q \times \Gamma \rightarrow P(Q \times \Gamma \times \{L, R\})$

Multitape Enumerator :- It is a 2 tape Turing machine that uses its second tape as the printer.



Formal Definition :- An enumerator, E, is a 7-tuple $E = (Q, \Sigma, \Gamma, \delta, q_0, q_{print}, q_{accept})$ where

1) Q is a finite set of states.
2) Γ is the work tape (finite)
3) Σ is the output tape alphabet (finite)
4) $\delta : Q \times \Gamma \rightarrow Q \times \Gamma \times \{L, R\} \times \Sigma_\epsilon$ is the transition function, $\Sigma_\epsilon = \Sigma \cup \{\epsilon\}$

5.) $q_{print} \in Q$ is the print state.

6) $q_{accept} \in Q$ is the accept state, where $q_{accept} \neq q_{print}$

for ex, $\delta(q,a) = (r, b, L, c)$ means the current state is $q \in Q$ reading a, the enumerator E enters to state r, writes b on the work tape, moves the work tape head to the left (L), writes c on the output tape (printer) and moves the output tape head to right if $c \neq \epsilon$

Theorem:- A language is Turing-recognizable if and only if some enumerator enumerates it.

($\Rightarrow$) RTP:- If we have an enumerator, E, that enumerates a language A then a Turing Machine (DTM) M recognizes A.

M = "On input w":

1.) Run E. Every time E outputs a string, compare it with w.

2) If w ever appears in the output of E. "accept" "

($\Leftarrow$) RTP:- If DTM M recognizes the language A, then we need to construct an enumerator E for A we call that $s_1, s_2, s_3, \ldots$ is a list of possible strings in Σ^*.

E = "Ignore the input

1. Repeat the following for $i = 1, 2, 3, \ldots$
2. Run (simulate) M for i steps on each input $s_1, s_2, \ldots s_i$
3. If any computations accept, print the corresponding string s_i in the output tape (printer)."

Problem:-

a)

b)

Problem:-

* Decidable means no looping. If will accept | reject

Problem:- Show that the collection of decidable languages is closed under the operation of
a) Complementation
b) Intersection.

$L = \{w \mid$ string w is accepted by a DTM, $M\}$

a) $\bar{L} = \{w \mid \bar{w}$ is accepted by $M\}$

For any decidable language, say L, let M be the DTM that decide it.
Goal is to construct a DTM say, M' that decides $\bar{L}$.

M' = "On input string w:
1. Simulate M on input string w.
2. If M accepts w, "reject"; otherwise "accept."

b) For any two decidable languages L_1 and L_2 let M_1 and M_2 the DTM's that decide them
Goal is to construct a DTM, M', that decides $L_1 \cap L_2$

M' = "On input string w:
1. Simulate M_1 on w. If M_1 rejects it, reject
2) Simulate M_2 on w. If accepts, "accept"
Otherwise, "reject" ".

Problem: Design a DTM, say M_1, that decides language $A = \{0^k 1^k \mid k \geq 0\}$

M_1 = { on input string $w = 0^* 1^*$:
1. S can across tape and "reject" if a "0" is found to the right of a "1"

2) Repeat if both 0's and 1's remain on the tape:

3 Scan across the tape, crossing off a single 0 and single 1.

4. If 0's still remain after all 1's have been crossed-off, or, if 1's still remain after all 0's have been crossed-off "reject".

5) If neither 0's nor 1's remain on the tape, "accept".

Time Complexity :- n = length of $w = |w|$

Stage 1 :→ $O(n)$

Stage 2 & 3 → $\frac{n}{2} O(n) = O(n^2)$

Stage 4 & 5 :→ $O(n)$

Total time $O(n) + O(n^2) + O(n) = O(n^2)$

EST 29/8/24

* Grasshopper effect along Global distillation:
POP - Persistent obscuring pollutants → disease

* Global dimming → due to Aerosols
↳ block radiation → leads to low temprature
Global dimming Currently in India & China.
[illegible] atmosphere &

2) Mantle
3) Crust

upper
Crust + Mantle = Lithosphere

* Continental drift Theory :- by Alfred wagner
Earth was single mass called "pangea".
Over the years it broken into parts/continents.

* Earth system Science :-

* Volcanoes (geosphere) → Ash
↓ affects ↓ affects
Biosphere ~~affects~~ Atmosphere
Hydrosphere

* Hurricanes (Atmosphere) → Hydrosphere
→ Biosphere
→ Lithosphere (causes land slides)

* Earthquake (Geosphere) → Fire accident (Atmosphere)
↓ → Human being
water (Tsunami)

MCT 30/8/24 Chapter Time Complexity

Definition [Running time or time Complexity) :-
Let M be a DTM that halts on all inputs
{Decidable TM}
The running time (time complexity) of M is the function
$f: N \to N$, where N is the set of natural (positive numbers

$f(n)$ is the maximum number of steps or iterations that the turing machine M uses on any input of length (size) n. If $f(n)$ is running time of M, we say that M runs in time $f(n)$ and that M is an $O(f(n))$ time DTM.
$n = |\alpha|$ = length or size of input α

Definition (Time Complexity class)
Let $t: N \to R^+$ be a function, where N = set of natural number, R^+ = set of positive real numbers. Define the time complexity class, TIME $(t(n))$, to be the collection of all languages (problems) that are decidable by an $O(t(n))$ -time DTM.

$$\text{TIME}(t(n)) = \{L \mid L \text{ is a language (problem) decided by } O(t(n))\text{-time DTM}\}$$

Solution1:- $A \in \text{TIME}(n^2)$
Solution2:- $A \in \text{TIME}(n \log_2 n)$
Solution3:- $A \in \text{TIME}(n)$

The class P (Polynomial Time class)
$P := \{L \mid L \text{ is a language decided by polynomial time (poly-time) DTM}\}$

$$:= \bigcup_{k \in N} \text{TIME}(n^k)$$

Problem:- Let $G=(V,E)$ be a directed graph, where V is the set of vertices (nodes), and E is the set of edges (links) of G. Define $\langle G \rangle$ as the (reasonable) encoding of the graph G. $\langle G \rangle$ can be adjacency matrix or adjacency linked list.
Define the following formal problem:-
PATH := $\{ \langle G, s, t \rangle \mid$ G is a directed graph having an (s,t) path$\}$

Theorem :- PATH $\in$ P

Proof:- Input: $\langle G, s, t \rangle$
Output:- Accept if there is an (s,t) path;
Reject, otherwise

Steps (Iterations/stages):-
1.) Place a mark ✓ on the node "s".
2.) Repeat the following until one fails to mark an additional node in G.
3.) Scan all the edges of G. If there is an edge $(u,v) \in E(G)$ from the already marked "u" to an unmarked node "v", then marks "v"
4.) If the node "t" is marked, then "accept", otherwise "reject".

Time Complexity
Step 1:- $O(1)$
Step 2 & 3 :- $n = |V|$
$|E| \leq \frac{n(n-1)}{2} = O(n^2)$

Total time $= O(n^2) =$ poly-time
Hence, PATH is in P.

Bookbird
Date
Page

Problem:- Define the following formal Problem:-
RELPRIME := $\{\langle x,y\rangle \mid x,y \in N$ are relatively primes, where N is the set of natural numbers $\}$

Claim:- RELPRIME $\in P$

$x,y \in N$ are relatively primes if and only if $gcd(x,y)=1$. Since the gcd of 2 integers a and b is always positive.

$gcd(a,b) = gcd(-a,b) = gcd(a,-b) = gcd(-a,-b)$

In general, $gcd(a,b) = gcd(|a|,|b|)$

* DTM(M_1) for deciding gcd(x,y) [Euclid's gcd algorithm].

Input:- $\langle x,y\rangle$

output:- $\langle gcd(x,y)\rangle$

Steps:- 1) Repeat until $y=0$
2) $x := x \bmod y$
3) Exchange x and y
4) output : x

Time Complexity:- $O(\min(\log_2 x, \log_2 y))$

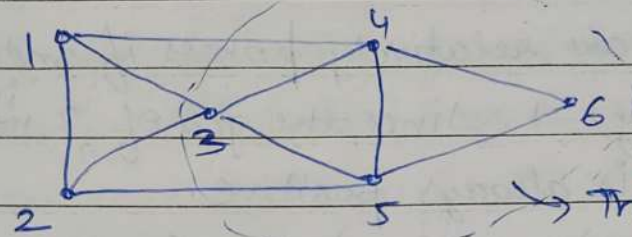
* DTM M_2 for deciding whether x and y are relatively primes.

Input:- $\langle x,y\rangle$

output:- Accept if $gcd(x,y)=1$,
Reject otherwise

Stages:- 1) Run(simulate) M_1 on $\langle x,y\rangle$
2) If the tape ($\langle gcd(x,y)\rangle$) contains 1 then "accept"; otherwise "reject".

Problem :- A K-clique in an undirected (directed) graph $G = (V, E)$
= a subgraph of G which is isomorphic to the Complete graph K_k.



→ This is a 4-clique ($\{3,4,5,6\}$) of G.

Define the following problem:
TRIANGLE = $\{\langle G \rangle \mid G$ Contains a triangle$\}$

Claim: TRIANGLE $\in P$.
A triangle in an undirected graph is a 3-clique
Construct a DTM, M that decides TRIANGLE in poly-time
M = "On input $\langle G \rangle$, where G is a (undirected) graph:
1.) For each triple of vertices (v_1, v_2, v_3) in G:
2) If the edges (v_1, v_2), (v_2, v_3), (v_1, v_3) are all edges in G, "accept"
3) No triangle has been found in G, so "reject"

Time Complexity: $n = |V|$ = no. of vertices in G
$n_{C_3} = O(n^3)$

2/9/24 EST
Anything which is in access amount from the

* Incidental information is difficult to capture. goes.
(when paying attention to multiple tasks)
* Phonological loop, Episodic buffer, visuo-spatial sketchpad.
* Central executive → responsible for evolution

Q where is working memory located in brain? ~~In all~~
Ans:- In all parts of brain. 3 regions

* PFC is necessary for working memory
(Prefrontal Cortex)

MCT 3/9/24

PROBLEM [P]:- Show that P is closed under (i) union, (ii) Concatenation, (iii) Complement

Proof: (i) For any 2 P-languages L_1 and L_2, let M_1 and M_2 be the corresponding DTM's that the DTM's decide them in poly-time.

Goal:- Construct a DTM, say, M', that decides $L_1 L_2$ in poly-time.

M' = " on input string w:

1. For each way to cut w into 2 substrings w_1 and w_2, that is, $w = w_1 w_2$:

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

2. Run (simulate) M_1 on w_1 and M_2 on w_2. If both M_1 and M_2 accept, then "accept".
3) If w is not accepted after trying all possible cuts then "reject" it.

Hence, $L_1, L_2 \in P$

(iii) For any P-language L, let M be the DTM that decides L in poly-time

Goal:- Construct a DTM, M', that decides the Complement of L, $\bar{L}$ in poly-time.

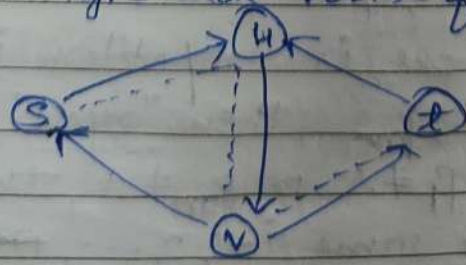
M' = " on input string w:
1. Simulate M on w.
2. If M accepts w, then "reject", otherwise "accept".

Hence, $\bar{L} \in P$.

The formal problem for Hamiltonian path in a (directed) graph $G=(V,E)$ is as follows:-

HAMPATH := $\{ \langle G, s, t \rangle \mid$ there is a (directed) Hamiltonian path from vertex "s" to vertex "t" in $G \}$

* An (s,t)-hamiltonian path is an (s,t)-path that goes through each vertex of G exactly once.



$\langle s,u,v,t \rangle$ is an hamiltonian path

The class EXP

$$EXP := \bigcup_{k \in N} TIME(2^{n^k}) = \{ L \mid L \text{ is a language decidable by an exp-time DTM} \}$$

M = "On input $\langle G, s, t \rangle$

1. Construct all the (s,t) paths in G
2. Verify if any (s,t) path is Hamiltonian

If so, "accept", otherwise, "reject".

The class NP → (This is not non-polynomial)

$NTIME := \{L \mid L$ is a language that is decided by $O(f(n))$ - time NTM, $N\}$

NP = Class NP = Collection of all languages that are decided by non-deterministic algorithms in poly-time.

$$= \bigcup_{k \in N} NTIME(n^k)$$

Theorem: HAMPATH $\in$ NP.

Goal: Construct a NTM, N, that decides HAMPATH in poly-time.

N = "On input $\langle G, s, t \rangle$:

$O(m)$ → 1) Non-deterministically (randomly) generate a sequence of m nodes, say, $p_1, p_2 \cdots p_m$; where each node p_i is chosen from the vertex set $V = \{1, 2, \cdots, m\}$

$O(m)$ → 2) If there is repetition in the chosen $p_1, p_2, \cdots p_m$, then "reject".

$O(1)$ → 3) If $p_1 \neq s$ or $p_m \neq t$, then "reject".

$O(m^2)$ → 4) If for some $i = 1, 2, \cdots, m-1$, the edge (p_i, p_{i+1}) is NOT an edge of G, then "reject".

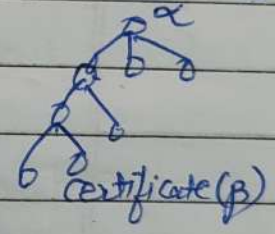
5) Otherwise "accept".

Bookbird
Date
Page

Defn:- (Poly time Verifier) :- Let L be a language. A verifier for L is a DTM, V, such that
$L = \{\alpha \mid \langle \alpha, \beta \rangle$ is accepted by V for some string β in poly time$\}$
For every $\alpha \in L$ there exists a certificate B such that V accepts $\langle \alpha, \beta \rangle$.

Theorem:- $L \in NP$ if and only if L has a poly-time verifier.

Proof:- [Necessary condition]
We are given that $L \in NP$
R.T.P: L has a poly time verifier, V
Let $\alpha \in L$, and N be a polytime NTM that decides L
There is a branch of computation that leads to acceptance of input string α.
∴ The certificate 'β' would be the prescription for that branch.



Claim:- A poly-time verifier V for L.
Input:- $\langle \alpha, \beta \rangle$
Output:- Accept / Reject.
Stages:- 1) Simulate N on α with the choices of transitions prescribed (deterministically) by β.
2) If N accepts α on this branch of computation, "accept", otherwise "reject".

[Sufficient Condition]
Let L has a poly-time verifier "V".
Assume that L runs $\leq n^k$ time (steps) for some Constant $k \geq 0$.

Construct a NTM, N

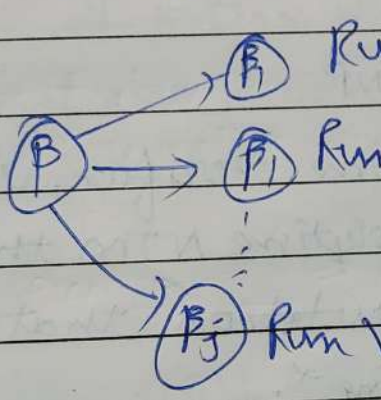
Input: α

output: Accept/Reject

Stages: 1) Non-deterministically generate strings β of length $\leq n^k$

2) Run V in $\langle \alpha, \beta \rangle$

3) If any branch accepts "accept", otherwise "reject".



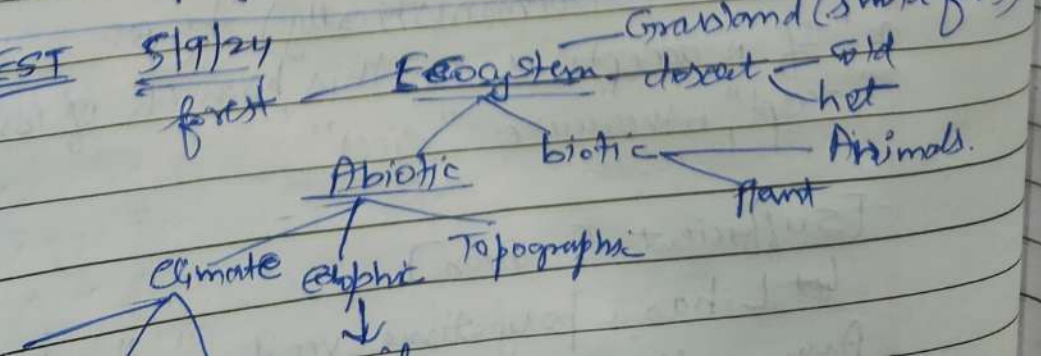
$L \in \bigcup_{k \in N} NTIME(n^k)$

LENP

To prove L is in NP:-

1. A poly-time verifier for L
2. A poly-time NTM for L

ESI 5/9/24



* Yellow stone national park (1995)
⇒ vegetation loss due to "Elk" eating grasslands so they introduced "wolves".
⇒ Hooves
⇒ Cold water has high O_2.

Lindeman 10% law
* The energy transferred from one to another organisms is utilized only 10% by another organism.

* ① Natural Nitrogen fixation:- by lightning in sky
② Biological fixation → bacteria

LNM 5/9/24
* The human brain is not designed for multitasking as it need to switch between tasks which needs to reorient the attention.

MCI 6/9/24
Problem:- CLIQUE = $\{\langle G, K\rangle \mid G$ is an undirected graph with a K-clique where $k>0$ is an integer constant$\}$

Theorem:- CLIQUE is in NP

Proof:- [Poly-time verifier] :-
Poly-time Verifier, V, for CLIQUE
Input:- $\langle\langle G, K\rangle, T\rangle$ where $T \subseteq G$ is subset [T = certificate]
Output:- Accept, if G has a K clique;
Reject, otherwise
Stages:- 1) If T does not contain K nodes, then "reject". $O(k)$
2) If G does not contain an edge (u, v) with $u, v \in T$ then "reject" → $O(m^2)$ (m = no. of nodes)
3) otherwise, "accept".

Problem: Consider the following problem:
SUBSET-SUM = $\{ \langle s, t \rangle \mid s$ has a subset T' with $t = \sum_{x \in T'} x \}$

Theorem: SUBSET-SUM $\in$ NP.

$\alpha = \langle s, t \rangle$

$\beta = \langle T' \rangle$ is a certificate of α.

Poly-time verifier, V, for SUBSET-SUM

Input: $\langle \langle s, t \rangle, T' \rangle$ where T' is subset of s

Output: Accept, if $t = \sum_{x \in T'} x$; Reject, otherwise

Stages: 1) If $T' \not\subseteq S$, then "reject".

2) Compute $t' = \sum_{x \in T'} x$

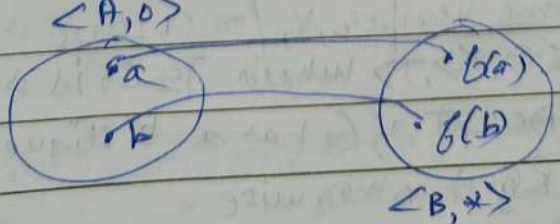
3) If $t' = t$, then "Accept" otherwise "Reject"

Problem: Graphs G and H are called "isomorphic" if the nodes of G may be re-ordered so that it is identical to H.

Let ISO-GRAPH = $\{ \langle G, H \rangle \mid$ G and H are isomorphic graphs $\}$.

Claim: ISO-GRAPH $\in$ NP.

Homomorphism: $f: A \to B$ is homomorphism or morphism if



$f(a \circ b) = f(a) * f(b) \quad \forall a, b \in A$

If G and H are isomorphic graphs, there exists a bijective map $f: G \to H$ that respects adjacency in the two graphs.

NTM, N, for ISO-GRAPH

N = "on input $\langle G, H \rangle$:

1) Let m be the number of nodes of G and H ($m = |G| = |H|$)

2) If G and H does not contain same number of nodes, then "reject".

Note:- $S = \{1, 2, \ldots, m\}$

A permutation $\pi: S \to S$ is a bijective map.

Total no. of permutations $= m!$

2) Non deterministically (randomly) select a permutation π of m elements.

3) For each pair of nodes, say x and y of G, check that (x, y) is an edge of G if and only if $(\pi(x), \pi(y))$ is an edge of H.

If all agree, then "accept".

If any differs, then "reject".

Definition (Poly-time Reduction):

language A is polynomial-time mapping reducible (poly-time reducible) to language B, written $A \leq_p B$, if a poly-time computable function $f: \Sigma^* \to \Sigma^*$ exists where for every input w

$w \in A$ if and only if $f(w) \in B$

($\Leftrightarrow$)

$\leq_p$ is a relation.

Lemma: If L_1, L_2 and L_3 are languages such that $L_1 \leq_p L_2$ and $L_2 \leq L_3$ then $L_1 \leq_p L_3$ (Transitivity)

Theorem: If $A \leq_p B$ and $B \in P$, then $A \in P$.

Theorem: If $A \leq_p B$ and $B \in NP$, then $A \in NP$.

→ Proof:- Let M be the DTM for deciding B and $f: \Sigma^* \to \Sigma^*$ be the polynomial time reduction from $A \to B$

By def^n for every input w,

$w \in A \iff f(w) \in B$

Goal:- "To Construct a DTM, N, deciding A as follows:

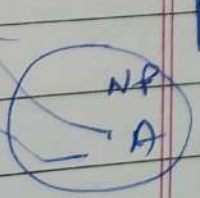
N = " On input w:

1.) Compute $f(w)$ using the poly-time reduction function $f: \Sigma^* \to \Sigma^*$.

2) Simulate M on input $f(w)$, and return the output whatever M output"

NP-hard: A problem "B" is called NP-hard, if every problem $A \in NP$ is poly-time reducible to B, ie. B is NP-hard.

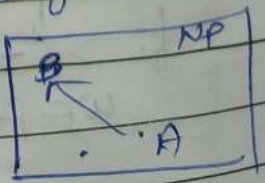
if $A \leq_p B \ \forall A \in NP$ { Here B can be P, exponential time machine



NP-Complete :- A problem "B" is called NP-Complete if following conditions are satisfied.

1) $B \in NP$

2) B is NP hard



* $P = \bigcup_{k \in N} TIME(n^k)$

$NP = \bigcup_{k \in N} NTIME(n^k)$

NP-hard = NPH

NP-Complete = NPC

$coNP = \{L \mid \bar{L} \in NP\}$

Complement of NP problems

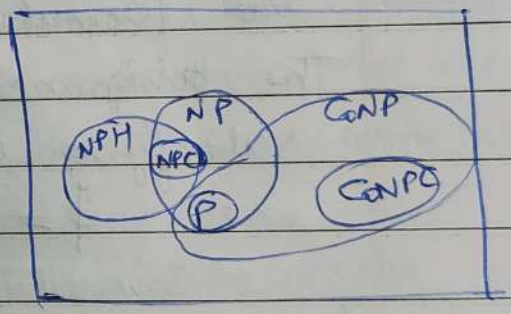
coNPC = coNP-Complete

EXP = exponential time

$= \bigcup_{k \in N} TIME(n^{2^k})$

* Conjectured relationships among different classes:

$P \subseteq NP$, $P \neq NP$



Boolean satisfiability (SAT) problem

Cook-Levin's Theorem:-

Boolean satisfiability is in P iff P=NP

SAT = Boolean satisfiability problem.

Let $x_1, x_2, \ldots$ denote the boolean variables (their value is either TRUE(1) or FALSE(0))

Let $\bar{x}_i$ denote the negation or complement of the boolean variable x_i.

* A literal is either a boolean variable (x_i) or its negation ($\bar{x}_i$)

A boolean formula in the propositional logic is an expression that can be constructed using literals and the operations: $\wedge$ (and), $\vee$ (OR) and $-$ (NOT).

AND ($\wedge$)			OR ($\vee$)			NOT ($-$)	
x	y	$x \wedge y$	x	y	$x \vee y$	x	$\bar{x}$
0	0	0	0	0	0	0	1
0	1	0	0	1	1	1	0
1	0	0	1	0	1		
1	1	1	1	1	1		

for ex:- $\phi(x_1, x_2, x_3) = (x_1 \vee \bar{x}_2) \wedge x_3$ is a boolean formula. ϕ is "satisfiable" if some assignment of truth values to the boolean variables let the formula ϕ evaluates to TRUE (1)
The assignment $(x_1=0, x_2=0, x_3=1)$ is a satisfying assignment because
$\phi = (0 \vee \bar{0}) \wedge 1 = 1$

Let SAT = $\{ \langle \phi \rangle \mid$ Boolean formula ϕ is satisfiable $\}$

Theorem: SAT $\in$ NP.

EST 9/9/24

Climate system

klima + logos

impact this area.

RCP:- Representative Concentration pathways (IPCC)
RCP60 & RCP85

LTM 9/9/24

Long Term Memory

* Interference:- old memories can affect new memories similar to it.
* Place fields:- Neuron cells that fire when we are in particular physical environment.
* Space & temporal information is stored in hippocampus.
* factual information does not depend on hippocampus. It depends on Cortex regions.

NCI

Theorem:- SAT is in NP.

Proof:- (Poly-time verifier)

Input:- $\langle\langle \phi \rangle, \beta \rangle$ | where $\phi = \phi(x_1, x_2, \ldots x_n)$ be a boolean formula with n variables $x_1, x_2 \ldots x_n$ and β is certificate (an assignment) >

Output:- Accept, if ϕ is satisfiable under β
Reject, otherwise.

Stages 1) If β does not contain 'n' boolean variables assignment, then "reject"
2) Evaluate ϕ on β.
3) If ϕ evaluates to 1, then "accept", otherwise "reject".

CNF- satisfiability (CNF-SAT) :- A boolean formula ϕ is in Conjunctive Normal formal (CNF) if and only if $\phi = \bigwedge_{i=1}^{k} C_i$, where $C_1, C_2, \ldots C_k$ are clauses and each $C_i = \bigvee l_{ij}$, l_{ij}'s are literals (x_i or $\bar{x}_i$)

Ex: $\phi = (x_3 \vee \bar{x}_4) \wedge (x_1 \vee \bar{x}_2)$ is in CNF.

Theorem: CNF-SAT $\in$ NP.

DNF- satisfiability (DNF-SAT) :- A boolean formula ϕ is in Disjunctive normal form (DNF) iff $\phi = \bigvee_{i=1}^{k} C_i$, where each clause $C_i = \wedge l_{ij}$.

Ex: $\phi = (x_1 \wedge \bar{x}_2) \vee (x_3 \wedge \bar{x}_4)$ is DNF formula.

Theorem: DNF-SAT $\in$

We have, $\phi = C_1 \vee C_2 \vee C_3 \vee \ldots \vee C_k$

where $C_i = \wedge l_{ij}$

DNF-SAT = $\{ \langle \phi \rangle \mid \phi$ is a satisfiable DNF-formula$\}$

Now, ϕ is satisfiable

$\Leftrightarrow$ atleast one of its clause is satisfiable.

It is worth noticing that a clause C_i is satisfiable iff for every atomic formula ϕ, C_i does not contain both x_i and $\bar{x}_i$ as literals.

Satisfiable: $\phi_1 = (\bar{x}_2 \wedge x_1 \wedge x_2) \vee (\bar{x}_1 \wedge x_3)$

Unsatisfiable: $\phi_2 = (x_1 \wedge \bar{x}_1 \wedge x_3) \vee (x_3 \wedge \bar{x}_3)$

* A CNF-formula is said to be a 3-CNF formula if every clause contains exactly 3 literals.

Ex: $\phi = (x_1 \vee \bar{x}_2 \vee \bar{x}_3) \wedge (\bar{x}_1 \vee \bar{x}_4 \vee x_1) \wedge (x_2 \vee x_3 \vee \bar{x}_4)$ is a 3-cnf formula.

Bookbird
Date
Page

Define: 3SAT = { <ϕ> | ϕ is a satisfiable 3-cnf boolean formula }

Theorem: 3SAT is in NP.

Theorem: If B is NP-Complete and $B \leq_p C$ for $C \in NP$, then C is also NP-Complete.

Proof: Given that C is in NP.

RTP:- C is NP-Complete

ie. $A \leq_p C$, $\forall A \in NP$ (C is NP hard)

since B is NP-Complete, every language in NP is poly-time reducible to B, and $B \leq_p C$; we have:- if $A \leq_p B$ and $B \leq_p C$ Then $A \leq_p C$

Theorem: (COOK-LEVIN's Theorem) :- SAT is NP-Complete

Proof:- To prove SAT is NP-complete, we must demonstrate the following 2 things:-

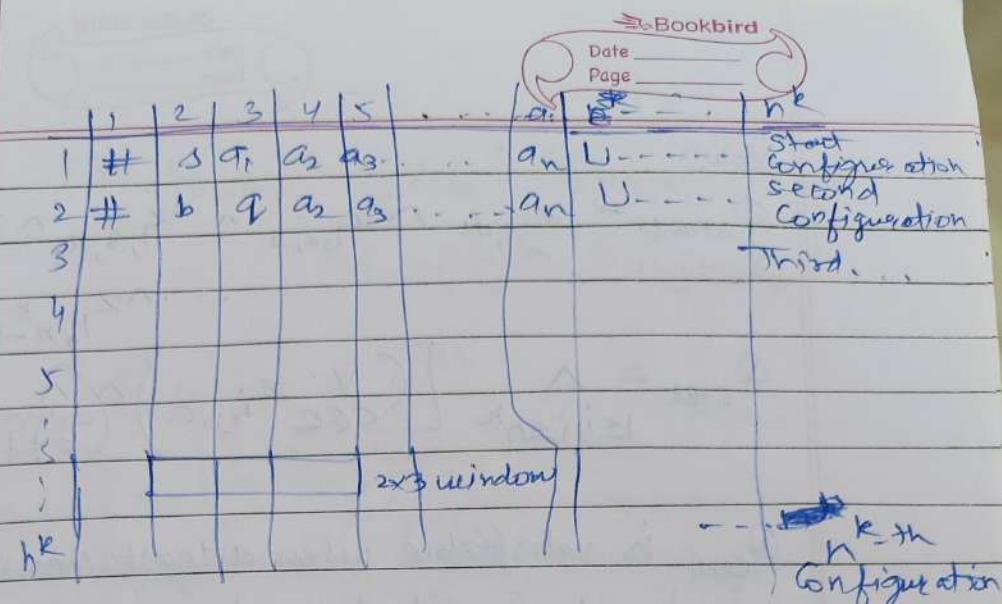
1) SAT is in NP
2) SAT is in NP-hard, i.e, $A \leq_p SAT$, $\forall A \in NP$

⇒ Assume $A \in NP$. Then, there exists a NTM, say, N which decides A in poly-time. If there is an input string α of length n, that is, $n = |\alpha|$ then N decides α in atmost n^k transitions (steps) where k is a constant.

Goal: Construct a boolean formula ϕ from N and α, such that ϕ is satisfiable iff α is accepted by N.

* Let $\alpha = a_1 a_2 a_3 \cdots a_n$ be an input string of size n

Table :- A tableau is an $n^k \times n^k$ table of configuration



Let us define:-

Q := set of states of N

Γ := the tape alphabet of N

F := the set of final states of N

C := the set of configurations

$:= Q \cup \Gamma \cup \{\#\}$

$|C| = |Q| + |\Gamma| + 1$ Constant $=$ constant

Let the boolean formula be

$$\phi = \phi_{start} \wedge \phi_{cell} \wedge \phi_{accept} \wedge \phi_{move}.$$

For each cell position (i,j) in the table and for $c \in C$ define a boolean variable $x_{i,j,c}$ such that

$$x_{i,j,c} = \begin{cases} 1 & \text{if cell } (i,j) \text{ contains } c \\ 0 & \text{otherwise.} \end{cases}$$

Ex: 1) $x_{1,1,\#} = 1$, $x_{1,1,c} = 0$ for all $c \in C \setminus \{\#\}$

Total number of boolean variables in $\phi = n^k * n^k * |C|$

$= n^{2k} * |C| = O(n^{2k})$

$= $ polynomial in n.

$$\sum_{\substack{1 \le i \le n^k \\ 1 \le j \le n^k \\ c \in C}}$$

Now,

$\phi_{start} = x_{1,1,\#} \wedge x_{1,2,s} \wedge x_{1,3,q_1} \wedge \ldots \wedge x_{1,n^k-1,\sqcup} \wedge x_{1,n^k,\#}$

$Q_{cell} = \bigwedge_{1 \le i,j \le n^k} \left[\left(\bigvee_{c \in C} x_{i,j,c} \right) \wedge \left(\bigwedge_{\substack{c,d \in C \\ c \neq d}} \left(\overline{x}_{i,j,c} \vee \overline{x}_{i,j,d} \right) \right) \right]$

Q_{cell} is satisfiable when atleast one variable is turned on and at most one variable must be turned on, that is, when EXACTLY one variable must be turned on.

$Q_{Accept} = \bigvee_{f \in F} x_{i,j,f}$
$1 \le i,j \le n^k$

For Q_{move} = we define a 2×3 window as "legal" if it is part of 2 successive rows such that the lower row follows the upper one of N's transitions.

for ex:-

a	q_1	b
q_2	a	c

legal

b	q_1	b
q_2	b	q_1

illegal

$Q_{move} = \bigwedge_{1 \le i \le n^k} (\text{the } (i,j)^{th} \text{ window is legal})$
$1 \le j \le n^k$

where, the $(i,j)^{th}$ window is legal

	j	$j+1$	$j+2$
i	b_1	b_2	b_3
$i+1$	c_1	c_2	c_3

2×3 window

$$= V\Big(x_{i,j,b_1} \wedge x_{i,j+1,b_2} \wedge x_{i,j+2,b_3} \wedge x_{i+1,j,c_1} \wedge x_{i+1,j+1,c_2} \wedge x_{i+1,j+3,c_3}\Big)$$

$$|\Phi_{start}| = O(n^k)$$

$$|\Phi_{cell}| = O(n^{2k})$$

$$|\Phi_{accept}| = O(n^{2k})$$

$$|\Phi_{move}| = O(n^{2k})$$

$$\therefore |\Phi| = |\Phi_{start}| + |\Phi_{cell}| + |\Phi_{accept}| + |\Phi_{move}|$$
$$= O(n^{2k})$$

EST

Tipping point :- A threshold after which changes cannot be reverted back.

* El-nino :- warming up of oceans. ENSO :- El nino southern oscillation
↓ variation in wind / sea surface temperature (SST)
(warm) (happens in eastern pacific ocean)

* La-nina :- Ocean becomes cold.

* Glacier melts → sea level rise → water → Thermal expansion
↓ CO_2
High temperature

* When Hypercane causes ozone layer depletion
* Glacier melts it causes decompressed tectonic plates, this creates magma giving rise to volcanos. ~~Eyja~~ Katla - iceland.
* Lightning causes forest fire.

LNM 12/9/24

Model of motor learning.

1) Cognitive stage 2) Associative stage
3) Autonomous stage.

MCT 13/9/24 Corollary: 3SAT is also NP-Complete.

3SAT := {<ϕ> | ϕ is a satisfiable 3 cnf-formula}

1. 3SAT is in NP (similar to SAT)
2) 3SAT is in NPhard.

Theorem:- If B is NP-Complete and B ≤p C for C is in NP, then C is also NP-Complete.

B = SAT, C = 3SAT } RTP SAT ≤p 3SAT

SAT : ϕ = ϕstart ∧ ϕcell ∧ ϕAccept ∧ ϕmove

(i) First, Convert ϕ to a cnf-formula

* ϕstart is already in CNF
* ϕcell " " "
* ϕaccept " " "
* ϕmove : using the distributive laws:

(a∧b)∨(c∧d) = (a∨c)∧(a∨d)∧(b∨c)∧(b∨d)

∴ ϕmove is in CNF

⇒ ϕ is now converted to CNF

(ii) Next, convert cnf-formula to 3 cnf formula in polynomial time
Using the rules: x = x∨x∨x
x∨y = x∨x∨y = x∨y∨y

we do not have any issues to convert ϕ to 3 cnf. If a clause contains exactly 3 literals, we are done. So, we can easily convert each clause of ϕ to 3 cnf when the no. of literals in a clause ≤ 3.

when the number of literals in a clause ≥ 4. A clause of the form $(x_1 \vee x_2 \vee x_3 \vee \dots \vee x_k)$ can be replaced by $(x_1 \vee x_2 \vee z_1) \wedge (\bar{z}_1 \wedge x_3 \vee z_2) \wedge (\bar{z}_2 \vee \bar{x}_4 \vee x_3) \wedge \dots \wedge (\bar{z}_{k-4} \vee x_{k-2} \vee \bar{z}_{k-3}) \wedge (\bar{z}_{k-3} \vee x_{k-1} \vee x_k)$ where z_i's are new boolean variables. Then $(x_1 \vee x_2 \vee x_3 \vee \dots \vee x_k)$ is satisfiable $\Leftrightarrow (x_1 \vee x_2 \vee z_1) \wedge \{\dots \wedge (\bar{z}_{k-3} \vee x_{k-1} \vee x_k)$ is also satisfiable.

Ex: $(x_1 = 1, x_2 = x_3 = \dots = x_k = 0)$ assignment is satisfiable.

Then, $z_1 \wedge (\bar{z}_1 \vee z_2) \wedge (\bar{z}_2 \vee z_3) \wedge \dots \wedge \bar{z}_{k-3} = 1$

$\Rightarrow z_1 = 1, z_2 = z_3 = 1, z_{k-4} = 1 \dots z_{k-3} = 0.$

Problems: Two computational problems L_1 & L_2 are poly-time equivalent if there exist poly-time reductions: $L_1 \leq_p L_2$ and $L_2 \leq_p L_1$.

Prove or disprove:- Every two NP-Complete problems are poly-time equivalent.

* Let P_1 and P_2 be two NP-complete problems.

Since P_2 is NP complete, so $P_2 \in NP$ and $\forall A \in NP$, $A \leq_p P_2$. Let $A = P_1$

$\therefore P_1 \leq_p P_2$ --- (1)

Similarly, P_1 is NP-complete, so, $P_1 \in NP$ and $P_2 \leq_p P_1$ --- (2) From (1) & (2): $P_1 \leq_p P_2$ and $P_2 \leq_p P_1$ $\therefore$ P_1 and P_2 are poly-time equivalent (TRUE)

Problem:- CLIQUE $= \{\langle G, K \rangle \mid G$ is an undirected graph with a K-clique, $K > 0$ is constant$\}$

Claim:- CLIQUE is NP Complete.

(1) CLIQUE $\in$ NP 2) CLIQUE is NP-hard. Take a known NP-Complete problem, say 3SAT (is NP-Complete). RTP. 3SAT $\leq_p$ CLIQUE

$\exists f: \Sigma^* \to \Sigma^*$ s.t. $\phi \longrightarrow f(\phi) = \langle G, K \rangle$

Goal: Given a 3cnf-boolean formula ϕ, construct an undirected graph $G = (V, E)$ such that it will a clique of "some size".

* Let $\phi = C_1 \wedge C_2 \wedge \dots \wedge C_k$ be a 3cnf-boolean formula where $C_i = (a_i \vee b_i \vee c_i)$ and a_i, b_i, c_i are the literals.

Construction of G:

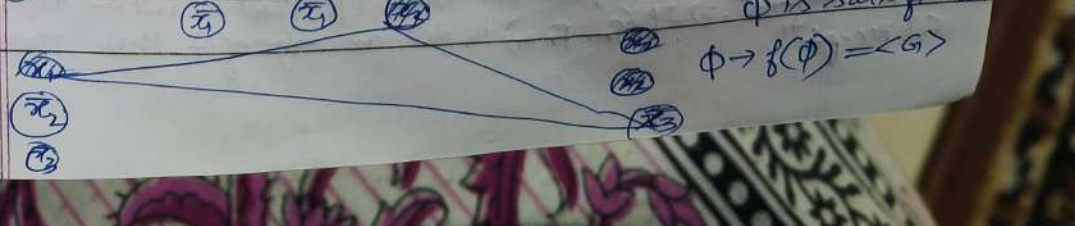
* For each clause $C_i = a_i \vee b_i \vee c_i$, create 3 nodes in G and label the nodes by the literals. $|V|$ = total no. of vertices in $G = 3k$.

* Add all possible edges among these $3-k$ nodes with following exceptions

(i) No edge between nodes in a clause (triple) is allowed, i.e. $(a_i, b_i), (b_i, c_i)$ (a_i, c_i) are not allowed.

(ii) No edge between nodes with contradictory labels is allowed i.e. $(x_i, \bar{x}_i)$ is not allowed.

Ex: $\phi = (x_1 \vee \bar{x}_2 \vee \bar{x}_3) \wedge (\bar{x}_1 \vee \bar{x}_1 \vee x_2) \wedge (x_2 \vee x_2 \vee \bar{x}_3)$ $\Rightarrow x_3 = 0$ $(\bar{x}_3 = 1)$



ϕ is satisfiable.

$\phi \to f(\phi) = \langle G \rangle$

* Directed broadcast address
* If any other network sends DBA to another network then the router drops it due to security reasons; as the attacker can perform DOS attack by sending excessive amount of messages.
* In classless addressing, 180.16.0.0 or 180.16.0.255 can also be IP of a host but in classful this cannot be used as host IP.

HAMPATH : $\{\langle G,s,t\rangle \mid$ there is a directed Hamiltonian path from node "s" to node "t" in the directed graph G$\}$

UHAMPATH =

Claim: HAMPATH is NP-Complete. 1) HAMPATH $\in$ NP
2) HAMPATH is NP-hard.

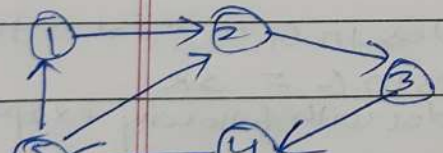
3SAT $\leq_p$ HAMPATH ; $\phi \longrightarrow f(\phi) = \langle G,s,t\rangle$

HAMCYCLE := $\{\langle G\rangle \mid$ there is a Hamiltonian cycle in the directed graph G$\}$

UHAMCYCLE := $\{\langle G\rangle \mid$ .. .. .. undirected graph G$\}$

Theorem: HAMCYCLE is NP-Complete.

* A directed Hamiltonian cycle in a directed graph G=(V,E) of length n=|V|. So the cycle goes through every vertex exactly once and then returns to starting vertex.



<1,2,3,4,5,1> is hamiltonian cycle
1) HAMCYCLE is in NP
2) HAMCYCLE is NP-hard; HAMPATH $\leq_p$ HAMCYCLE

1) NTM for HAMCYCLE: Input: <G> output: Accept/Reject
Steps: 1) Non deterministically generate a sequence of (n+1) nodes (n=|V|) say $P_1, P_2, \ldots, P_{n+1}$ from which each $P_i \in \{1,2,3,\ldots,n\}$
2) If $P_1 \neq P_{n+1}$ then "reject". 3) If there is a repetition in $P_1, P_2, \ldots P_n$ then "reject".
4) If for some $i=1,2,\ldots,n-1$ the edge (P_i, P_{i+1}) is not an edge of G then "reject". 5) If (P_n, P_{n+1}) is NOT an edge of G, then "reject".
6.) "Accept".