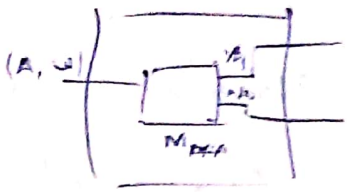
$\Rightarrow L_{NFA} = \{ \langle A, w \rangle : A$ is an NFA that accepts $w \}$

convert A to DFA & then use M_{DFA}



$\Rightarrow L_{EMPTY\_DFA} = \{ A : L(A) = \phi \}$

is Given a DFA A, Is L(A) empty

Algorithm:

(1) Mark the initial state Q.

(2) Repeat until no further states can be marked in one round
(i.e., mark states that can be reached from some other state)

(3) If atleast one final state is ~~unmarked~~ marked
Ans = NO

$\Rightarrow$ it can be reached via another states which are all marked

$\Rightarrow$ final state can be reached for atleast one I/P

$\Rightarrow$ L(A) is not empty

→ ⟨A, w⟩ the i/p seq that contain all i/p sequence seq for A

→ ⟨G, w⟩ : Is G connected

(1, 2, 3, 4), ((1,2), (2,3), (1,3), (3,1))

set of vertices — set of edges.

Algo:

① Mark first vertex (here 1)

② Repeat until no further vertices are marked in one round

- If for each vertex v, mark if there is an edge to v from a marked vertex

③ If all vertices are marked. Yes else NO

---

c] $L_{EQ.DFA} = \{ \langle A, B \rangle : L(A) = L(B) \}$

- Given 2 DFA A & B, do they accept the same set of strings

if: NO if $w \in L(A)$ & $L(B)$ or vice versa

$\Rightarrow L' = (L(A) \cap \overline{L(B)}) \cup (L(B) \cap \overline{L(A)})$

$\therefore$ construct a DFA : L'

$\therefore$ Given DFA A with $L(A)$

DFA with $\overline{L(A)} = L(\bar{A})$ i interchange final & non-final

& to make $\cap$ we know the method. discussed previously

so how to make $L(A) = L'$

or $L(A) \cap L(B)$

" DFA that accept $w \in L(A)$ & $L(B)$

$\Rightarrow$ let C: set of states $Q_c = Q_A \times Q_B$

$\Rightarrow$ transitions are those from both A & B.

$\Rightarrow$ final state (q_1, q_2) if $q_1 \in$ final states (Q_A)
& $q_2 \in$ final states (Q_B)

$\therefore$ Construct L' using above algo.

then Run M_{empty_DFA} on L' i if L' is empty

$\Rightarrow L(A) = L(B)$ $\Rightarrow$ say YES else NO

Theorem : L_{TM} is undecidable

Proof : ① create L_d : lang that is not turing decidable

② Now, if L_{TM} is decidable

show that M_d exists using M_{TM} which contradicts ①

∴ L_{TM} not decidable

Note

→ halting problem & Is empty problem all not turing decidable but its turing recognisable

Proof :

Reducibility

→ reducing one problem to another.

→ given TM (or algo) for problem A
how we can use it for
solving problem B

eg: L_{HALT} is undecidable

→ if L_{HALT} is decidable use
M_{HALT} to make M_{TM}
(not possible)

(1) $L_{EMPTY} = \{\langle M \rangle : L(M) = \phi\}$

i.e. M does not accept any string

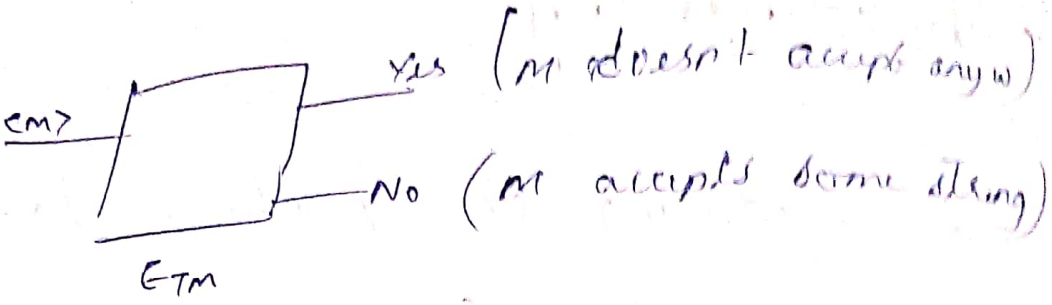
Is L_{EMPTY} decidable (does E_{TM} exists?)

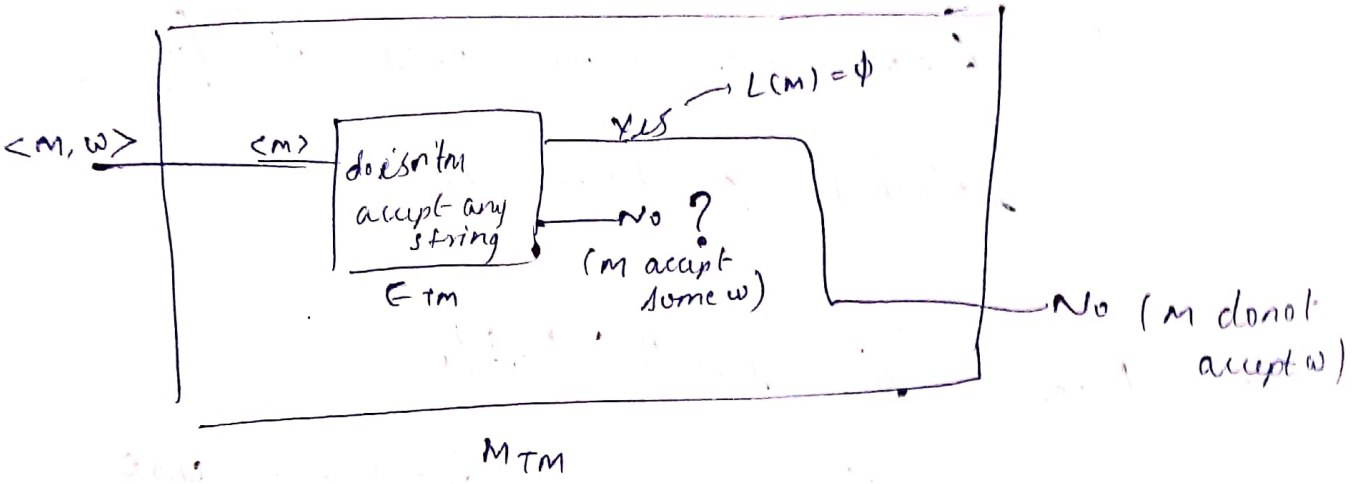
Theorem: L_{EMPTY} is undecidable.

Proof: We know L_{TM} is undecidable

∴ if L_{EMPTY} is decidable, then $\exists M_{TM}$

how?





∴ modify M and give it to E_{TM}

IMP

M_w :

if $x = w$ (if I/P is w)

run M on I/P x. (if M accepts w → yes)

if $x \neq w$ (if I/P to M_w is not x) → No

reject

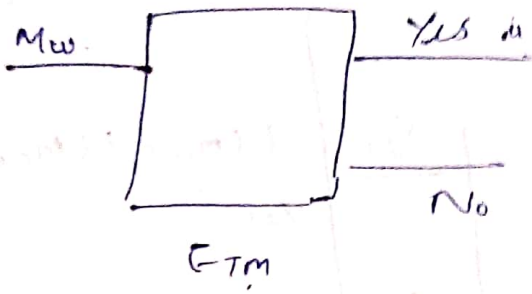
$L(M_w) \neq \phi$

$\Rightarrow L(M_w) = \{w\}$: if M_w doesn't accepts any string, then either $x \neq w$ or $x = w$ & M_w runs ∞

$L(M_w) = \phi$ (even when $x = w$, M_w does not accept)

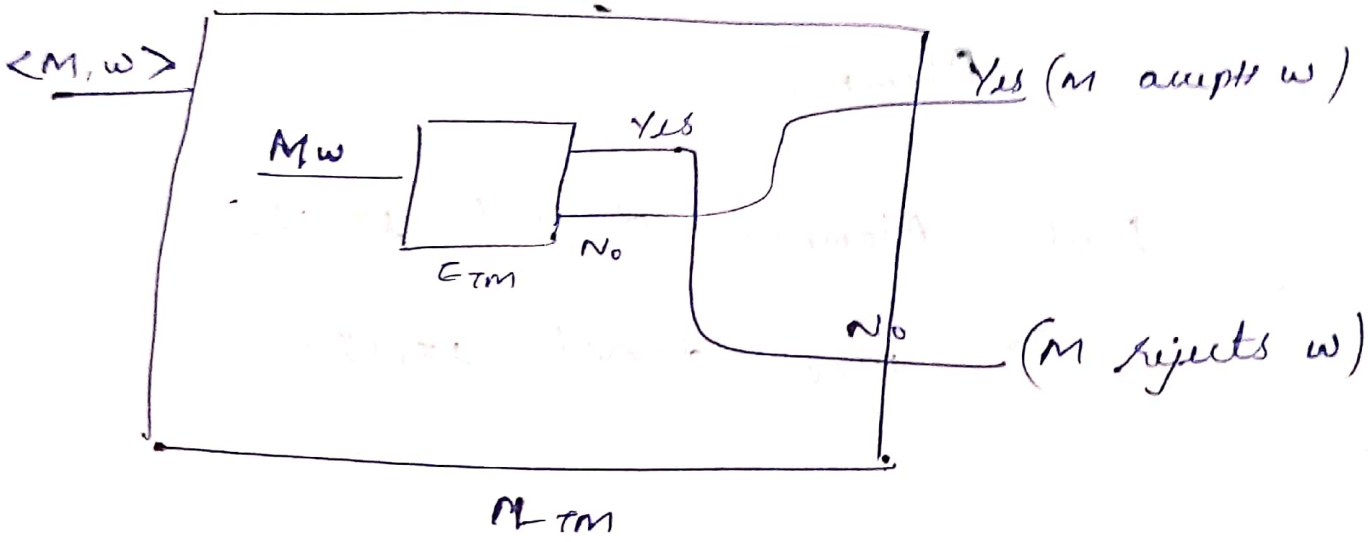
$\Rightarrow$ if M_w accepts some string, then it can only be $x = w$ & M accepts w

M_ω



M_ω doesn't accept w
⇒ M also doesn't accept w

⇒ M_ω accepts w (just it rejects)
⇒ M accept w

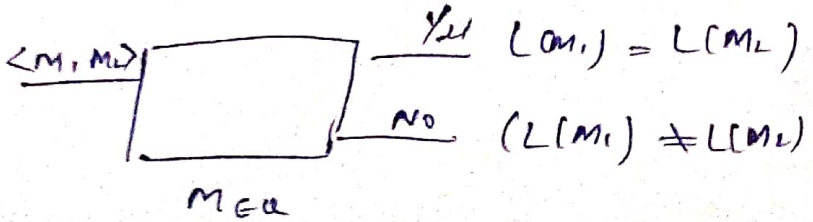


⇒ M_{TM} exists but we know L_{TM} is undecidable

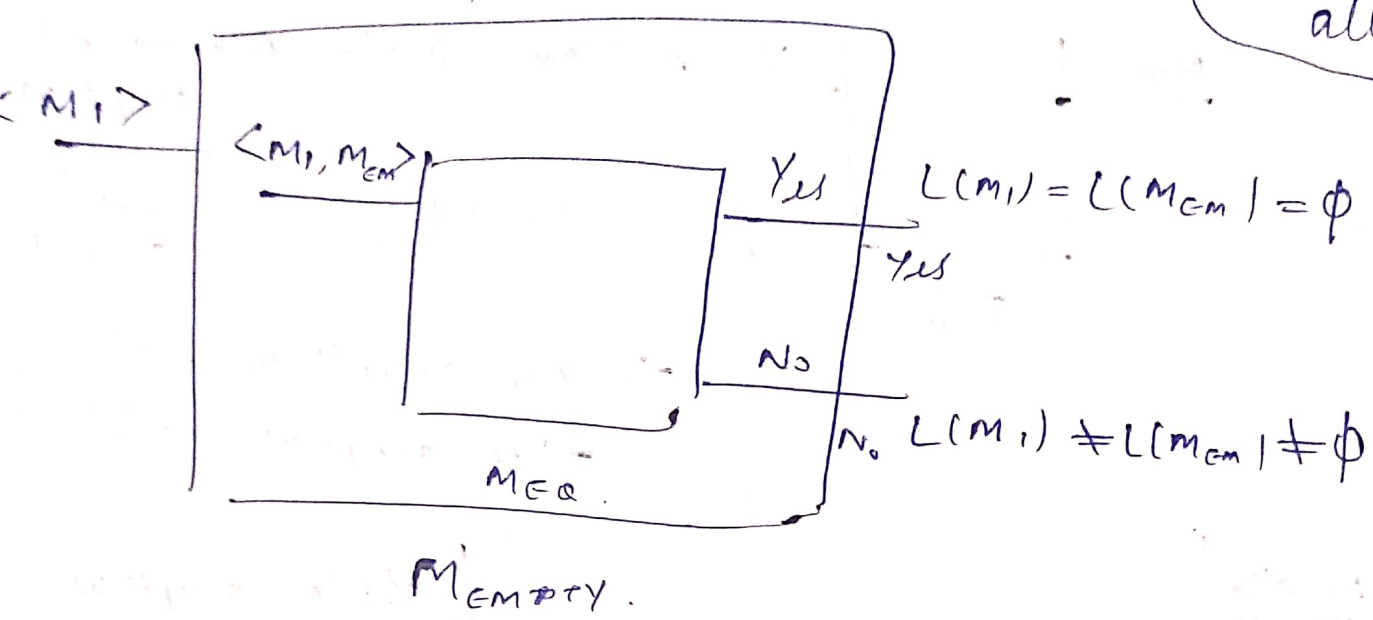
⇒ E_{TM} doesn't exists.

(2) $L_{EQ\_TM} = \{ \langle M_1, M_2 \rangle : L(M_1) = L(M_2) \}$

Is L_{EQ} decidable



Assume M_{EQ} exists. let M_{EM} rejects all strings



But M_{EMPTY} doesn't exists

$\Rightarrow$ M_{EQ} donot exists

Q. Given a TM M, is the lang recognised by M i.e., L(M) regular.

i.e. $L_{REG} = \{ M : L(M) \text{ is regular} \}$

eg: M_1 = accepts string of the form 0^n1^n

M_2: accepts string with odd no of 0s

M_3: Accepts w#w

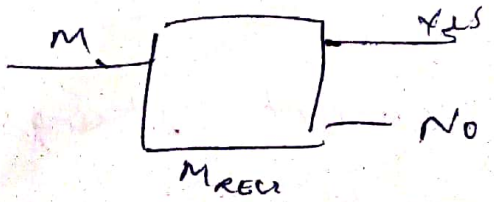
M_4: accepts all string

Here $M_2 \& M_4 \in L_{REG}$ can make DFA for them.

[pumping lemma can be used to check if a lang is regular.]

S.T L_{REG} is undecidable

Proof: Let M_{REG} decide L_{REG}.



Build M'_w s.t based on whether M'_w is regular we can say M accepts w or not

M'_w : Accept non regular language (eg: $0^n 1^n$) if M does not accept w

Accept regular lang if M accept w

If M'_w : $\begin{cases} x \neq w & \text{reject} \\ x = w & \text{run M with w} \end{cases}$

no irregular lang : $\begin{cases} \text{if M accepts w, is } M'_w \text{ accept regular lang} \\ \quad L(M'_w) = \{w\} \\ \text{if M doesnot accept w, } M'_w \text{ accept no string } \{\} \end{cases}$

$\Rightarrow$ let, if M accept w, M'_w accept all string

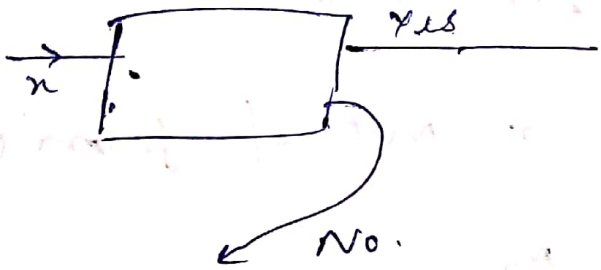
else

It accept strings of the form $0^i 1^i$ not regular

M_w

Step 1 :

→ Accept all strings that are of the form 0^n1^n



→ Step 2

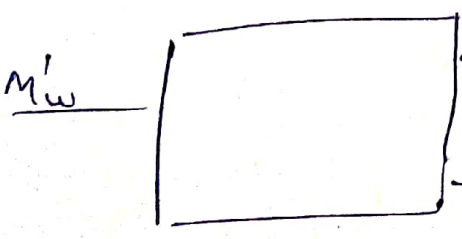
if $x \neq 0^n1^n$, run w on M, ⟶ Yes accept x.

∴ if m accepts w, then M_w accepts all string

regular.

if m does not accept w. it accepts only strings of the form 0^n1^n

not regular



Yes ⇒ L(M'w) = reg
⇒ M accept w

No ⇒ L(M'w) = non reg
⇒ M rej. w

$M_{1\_w_1}$: if $x = 0^n1^n$ accept

if not

run M with w_1.

if x not of form 0^n1^n

M_1 :

$L(M_{1\_w_1})$: 0^n1^n + all other strings ⟶ all string

M_2 accept all string either in step 1 or in step 2.

if

Time complexity of TM

NP — Non-deterministic polynomial

$P \subseteq NP$

As all det. TM can be seen as a non-deter. TM with exactly one choice (or no choice → move to Rej)

eg: ⊙ checking if 2 nos are relatively prime $\in P, \in NP$.

⊙ $0^n1^n \in P, \in NP$

⊙ ww $\in P, \in NP$.

⊙ check if a given no. is prime $\in NP$; $\in P$ (complicated proof)

eg: check if graph is connected

$\langle G \rangle$ encoded as (matrix, adjacency list)

size - $O(n^2)$ n = # vertices.

NP class (alternate defn)

A lang. $L \in NP$ if we can be

verified with the help of a certificate (whose size is polynomial to $|w|$) with a polynomial time soln.
to $|w|$

(Q1). does k-Colouring $\in NP$

$L = \{G$ that are 3-colourable$\}$

at present no known poly-time algo to check if a given $G \in L$

But given an assignment of colours to each vertex as certificate we can verify it in $O(m)$ m - # edges time.

(Q2) k-clique certificate - clique vertices
verify - if all are connected

$L \in NP$, if $\exists$ non-det. TM that recognises L in $O(|w|^c)$ i.e. it gives yes answer

is polynomial time. ('no' answer need not be in pol. time)
If 'no' answer can also be verified in polynomial time.

$L \in co\text{-}NP$ or $\bar{L} \in NP$

If $L \in P \Rightarrow L \in NP$ & $L \in co\text{-}NP$
(decides L)

T.P the 2 defn for NP is same.

(1) Use the path of non-det TM 'N' as our certificate. Now since N reaches Qaccept in pol. time, we can verify certificate in $O(|w|^2)$

(2) Guess the certificate & run the verifier algorithm. It guess the path in the non-det TM and combining all these path gives the non-det TM

Travelling salesman problem

It's to finding a least weighted hamiltonian

cycle → decision version is ∈ NP

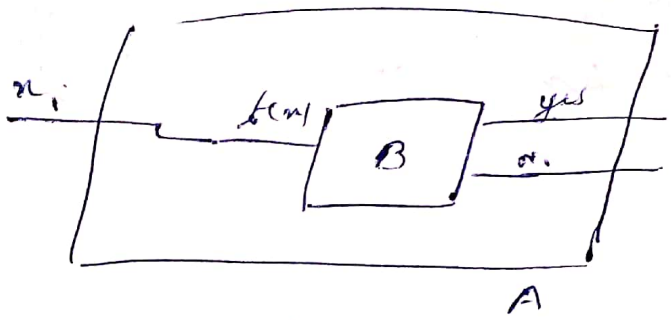
---

if prob A is not in P, then B is also not in P

$\Rightarrow$ B is in P $\Rightarrow$ A is in P

$A \leq_m^P B$

i.e., assume $B \in P$ (∃ an algo in pol. time



$f(x)$ in $|x|^c$

CNF - SAT

It ∈ NP as given a certificate we can verify it in polynomial time

$\Rightarrow$ if CNF-SAT ∈ P, then P = NP ?

NP-H class

Problems that are atleast as hard

as any problem in NP

i $\forall X : X \in NP\text{-H}$, $X \leq_m^p L$ where $L \in NP\text{-H}$.

$\therefore$ if $L \in P$, then $X \in P \Rightarrow NP = P$!

---

① clique :- Is there a clique of size k

② independent set :- Is there an ind set of size k.

Assume clique $\in NP\text{-H}$. Then does ind set $\in NP\text{-H}$?

let ① $= L_1$ & ② $= L_2$

$\forall X : X \in NP, X \leq_m^p L_1$

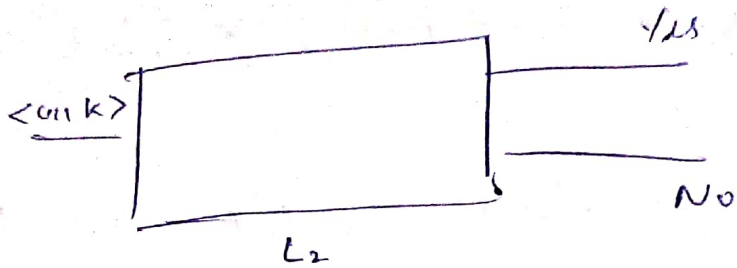
if $L_1 \leq_m^p L_2$ then $\forall X : X \in NP, X \leq L_2$

& $L_2 \in NP\text{-H}$

To show $L_1 \leq_m^p L_2$.

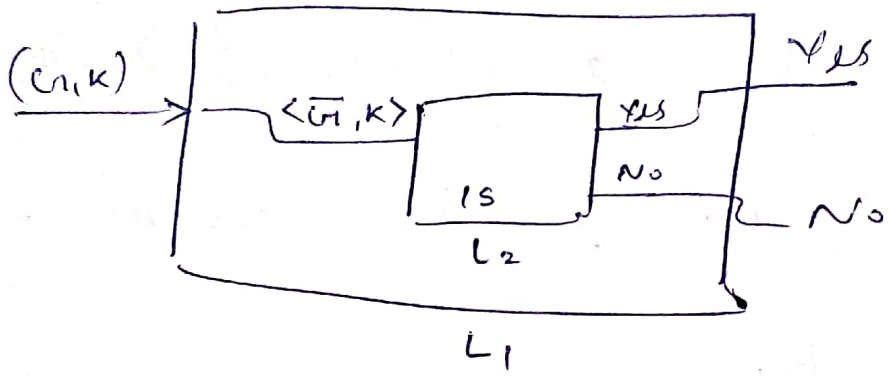
$\Rightarrow$ make L_1 using L_2 in polynomial time.

$\therefore$ if just prove that if L_2 runs in poly. time, L_1 also runs in pol. time



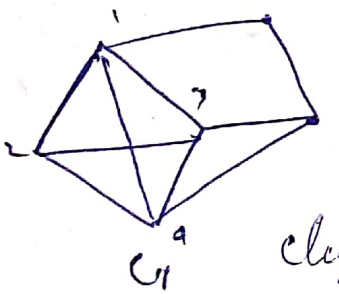
G has is of size k

(IS)

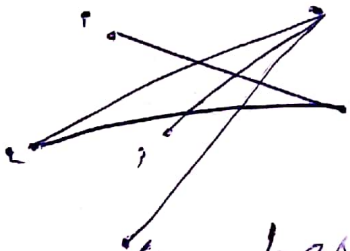


clique

G has clique of size k



$\Rightarrow \bar{G}$ =



clique of size 4

G1

4 has ind. set of size 4

If $\bar{G}$ has clique
$\bar{G}$ has ind set

$\therefore$

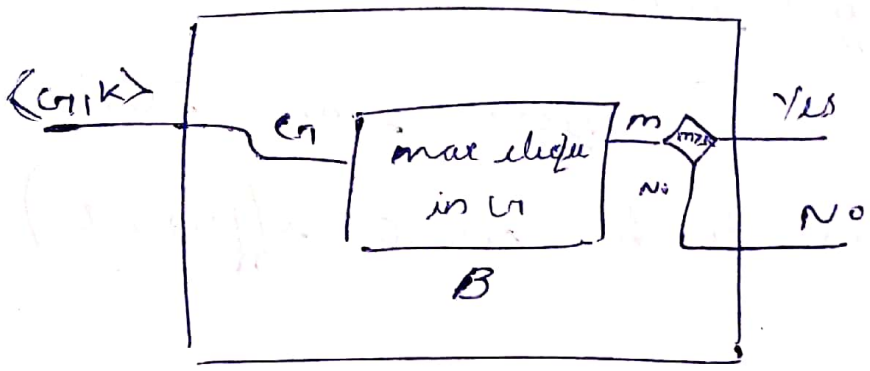
$f(G) = \bar{G}$ can be done in polynomial time

$L_1 \leq_p^n L_2$

(A) CLIQUE (decision problem)

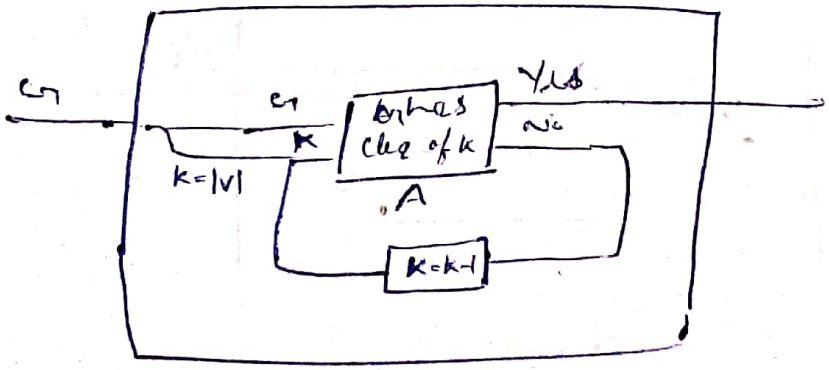
(B) CLIQUE (optimisation) - what is the max sized clique?

p.T that both are polynomialy equivalent



A

if B has $O(n^c)$ then A has $O(n^{c'})$



if A has $O(n^c)$ then B has $O(kn^c)$

(C) Find a biggest clique in G.
Remove an edge, check if it has

the same sized clique as before.

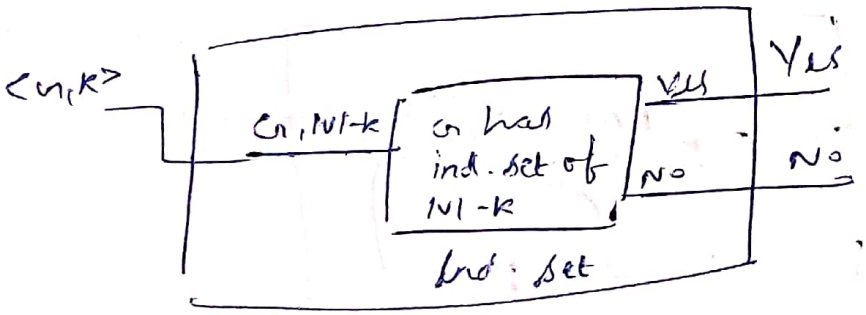
if NO. keep edge & remove another

if YES, remove another

repeat this till only clique edges remain

(or you can remove vertices to be more efficient)

① Vertex cover: set of vertices s.t all edges have atleast on vertex in it.



vertex cover

if G has vertex cover of size k, then G also has ind. set of size |V|-k ?

& vice versa