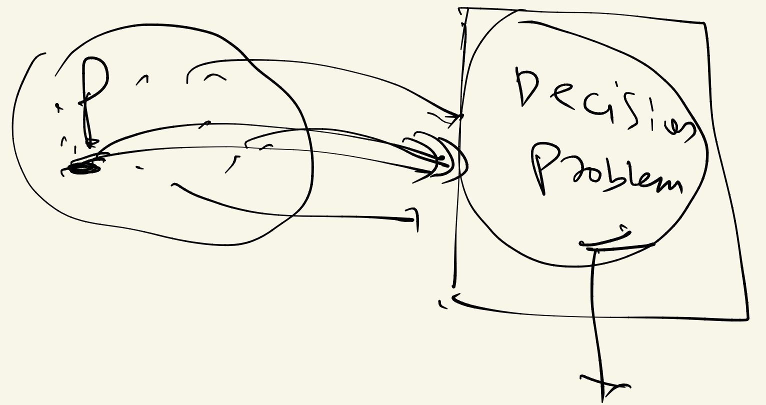
Quick Recap

- overview
- OPT ↔ Decision
- OPT → DECISION



Framework

Formal Lang Theory

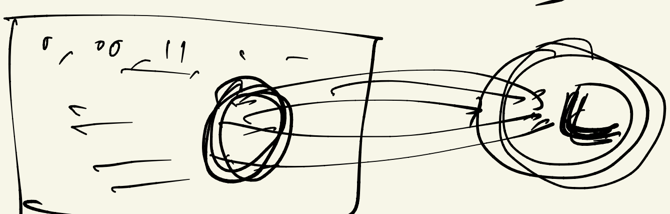
- alphabet

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

- $U = \Sigma^*$ $\quad L \subseteq \Sigma^*$

$$= \{0, 1, 00, 01, 11, 000, \ldots\}$$

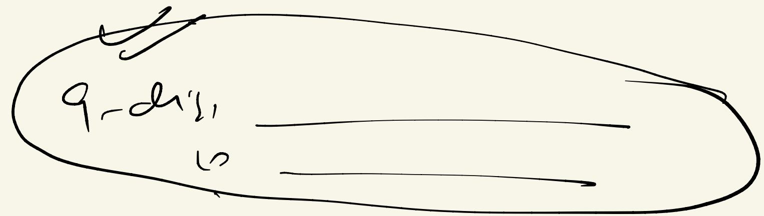
$U = \Sigma^*$



SSN

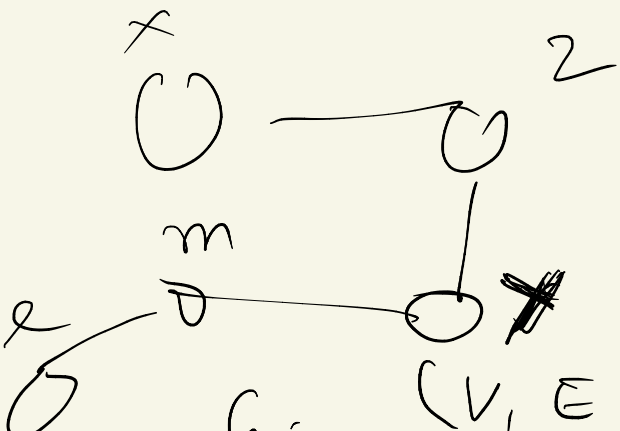
$$\Sigma = \{0, 1, 2, 3, 4, \ldots, 9\}$$

$$\Sigma^* = \{0, 1, \ldots, 9, \; 01, 00, 02, \ldots, \; 011, 012, \ldots\}$$



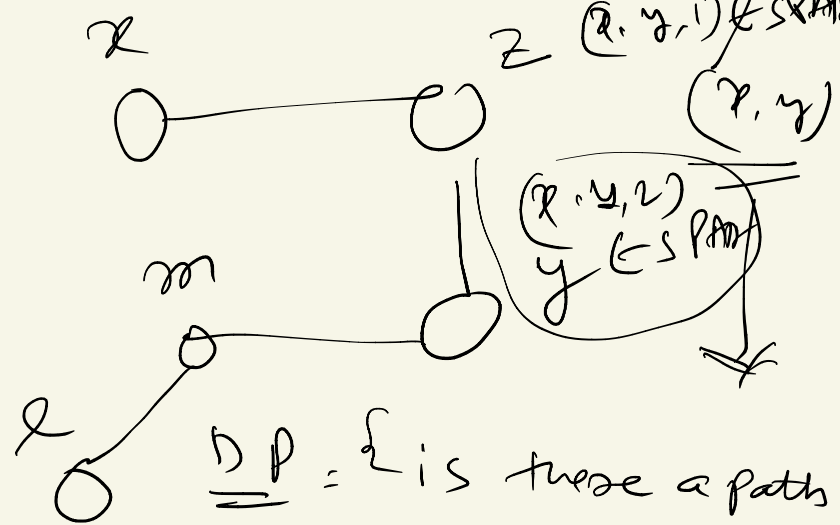
$$L = \{\text{9-digit string } \Sigma^* - 0\}$$

Encoding an instance of a Decision Problem



$G = (V, E)$

Find the Shortest path (x, y)



DP: {is there a path of length at most K betn x, y}

$\{G = (V, E), x, y, K\}$

SPATH

$\boxed{\{\{(x, z, y, e, m), \{(m, z)\,(z, y)\,(y, m), (m, e)\}, x, y, K\}}$

Complexity Class

P

Class of all Problem decided in Poly-time

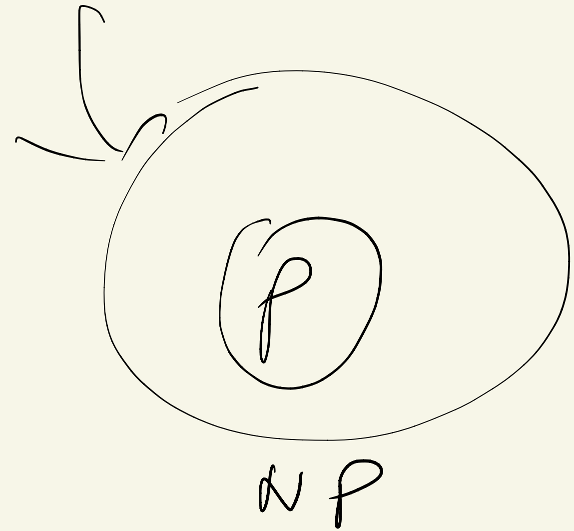
solve (actual Sol^n)

NP

verified in Poly-time

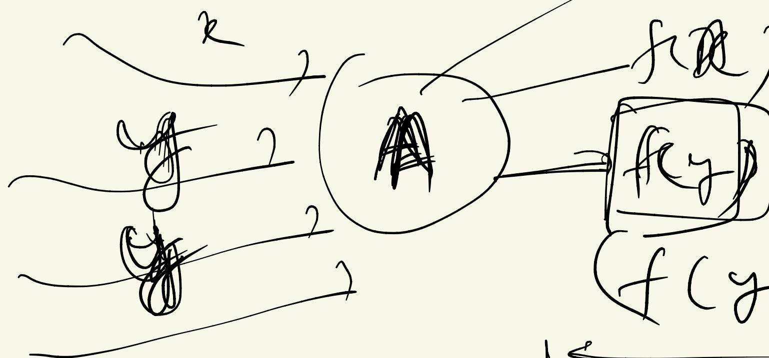
may or may not Solve them in P

P ∈ NP



Deterministic

$\boxed{\text{K-means}}$



undeterministic

May or may not know the sol

$\boxed{\text{HALTING PROB}}$

Stochastic

Alan Turing ~1930's

NP - completeness

= Ability to scale

- Find soln to known problems

UNKNOWN problems

GROUPING / CLASSIFYING PROBLEMS ?

Problem

time to solve

resource to solve

complexity of the problem

Prob Stmt

Sol^n Stmt

P

Decided in Polytime

✓

Find Soln

$\boxed{x \in P}$

NP

Verified in Polytime

✓

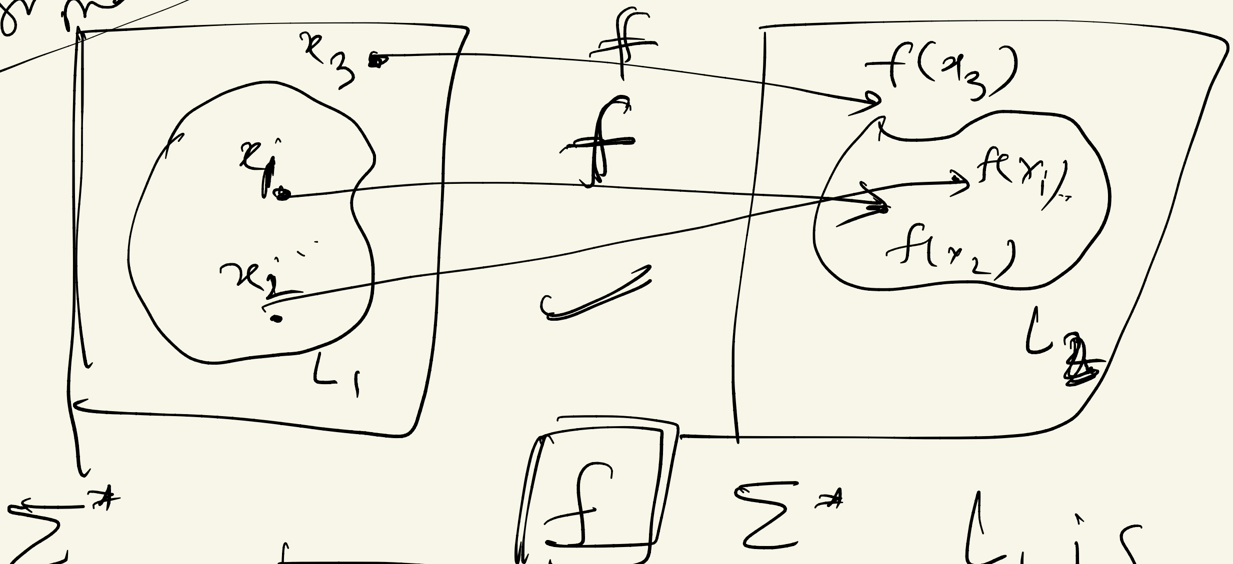
$Y \subseteq NP$

Problem Space

mapping is constructed / designed manually

iff $\forall x \in L_1, f(x) \in L_2$

$x \notin L_1, f(x) \notin L_2$



L_1 is no harder to solve than L_2

$$\boxed{L_1 \leq L_2}$$

L_1 is reducible to L_2

multiplication   Polytime   addition

L_1 $\xrightarrow{f}$ L_2 (addition)

Polytime

new problem $\boxed{x * y}$

known

Don't know the sol^n

$\underbrace{x + x + \cdots + x}_{y \text{ times}}$

Sol^n is also known

$\boxed{\text{multi} \in P}$

$\boxed{\text{MULTI} \le_p \text{ADD}}$

$L_1 \leq_p L_2$

L_1 is Poly-time reducible

$f \rightarrow O(n^k) \quad \forall \underline{\underline{k}}$

ADD( )

$\boxed{f \in P}$

$2 \cancel{+} 3$
$+ \cancel{4}$
-7

$2 + 2 + 2 + 4 + (-7)$

$(+)$

$\text{multi} \le_p \text{add}$

```
def add(x, y):
    ...
    return (x+y)
```

$f \sim$ Polytime

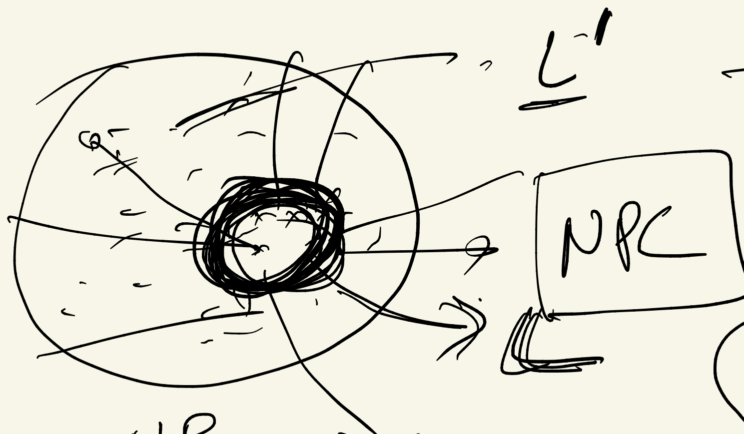
```
def multi(a, b)
    t = 0
    for i in range(0, b)
        t = add(t, a)
    return t
```

(n)

- Reductions
  - What is reduction
  - " " Poly time reduction
- Use reduction to solve an unknown Problem L_1 using a known Soln to Problem L_2

NP COMPLETENESS

- ✓ What is Reduction
- ✓ P, NP
- ✓ ~~NP is~~ more interesting



NP

Multi Add

$$\boxed{L' \leq_p L}$$

L' is no harder than L

Representation of NP

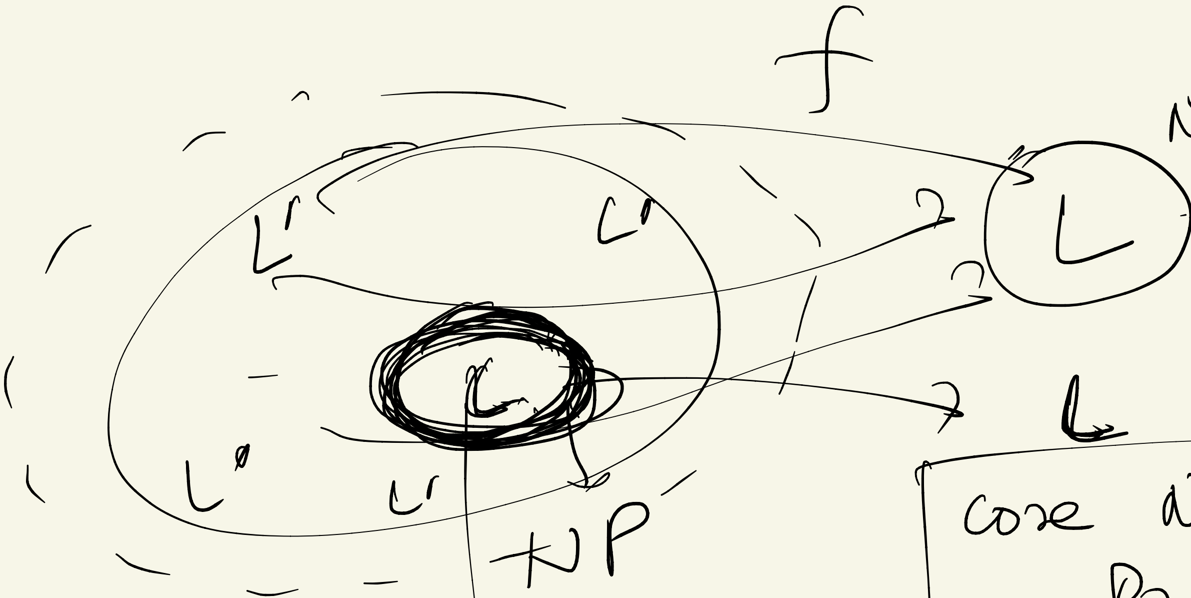
L is NPC

<u>iff</u> $\forall L' \in NP$

$L' \leq_p L$

then L is NPC

(NP-complete)



$\frac{NP\text{-hard}}{NP}$

core NP Problems

NP + (NP-hard) = NPC

$L' \in NP$

$\boxed{L' \le_p L}$

$L \in NP$- completness

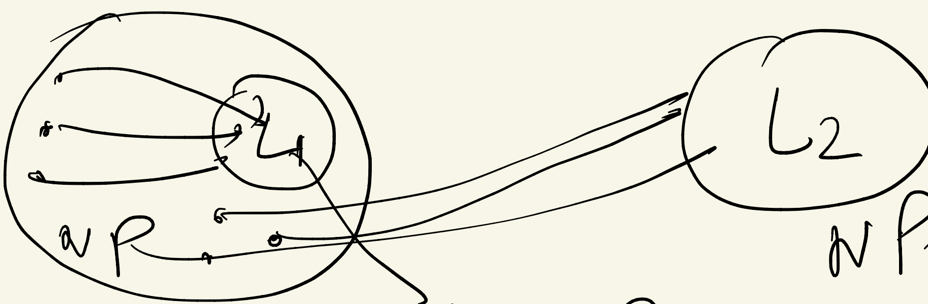
① $\boxed{L \in NP}$ ✓

② $\forall L'' \in NP \quad L'' \leq_p L$

NP Reduction Property

$\rightarrow$ $\boxed{\text{NP- HARD}}$

$$NPC = NP + NP\text{-}HARD$$



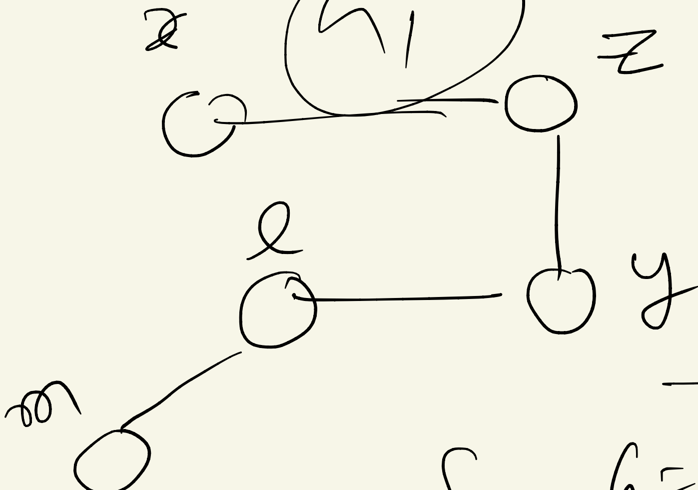
$L_2 =$ NP-hard

$L_1 \in NP$

$L_1 =$ NP-hard hence $\boxed{L_1 \in NPC}$

SPATH

G_1

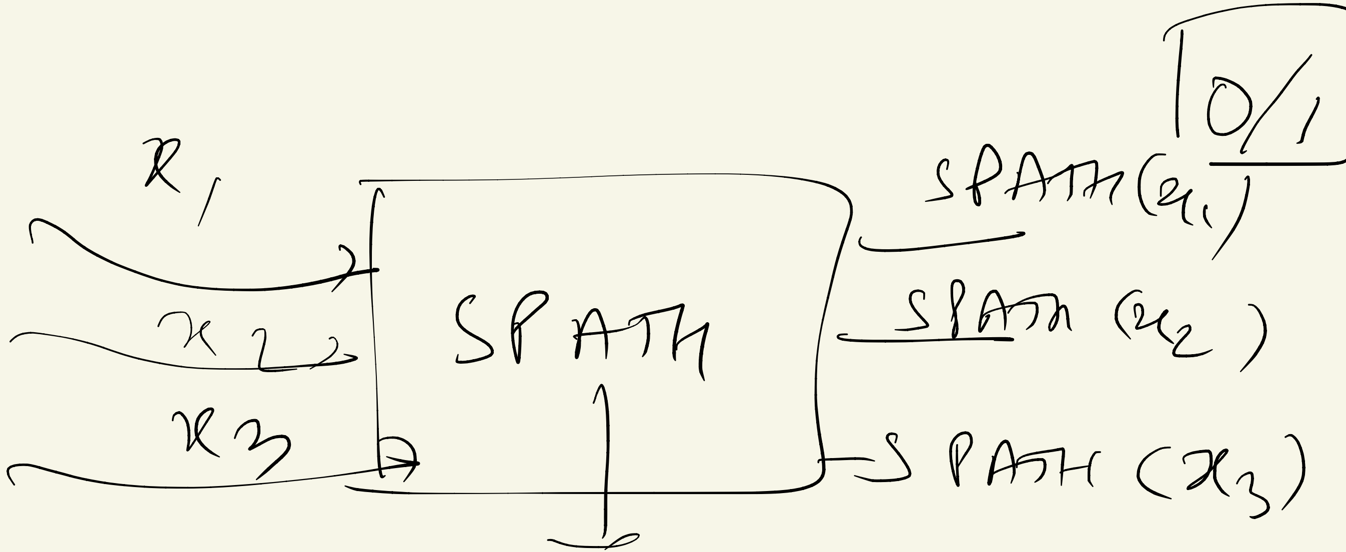


$(x, y, 1) \notin SPATH$

$(x, y, 2) \in SPATH$

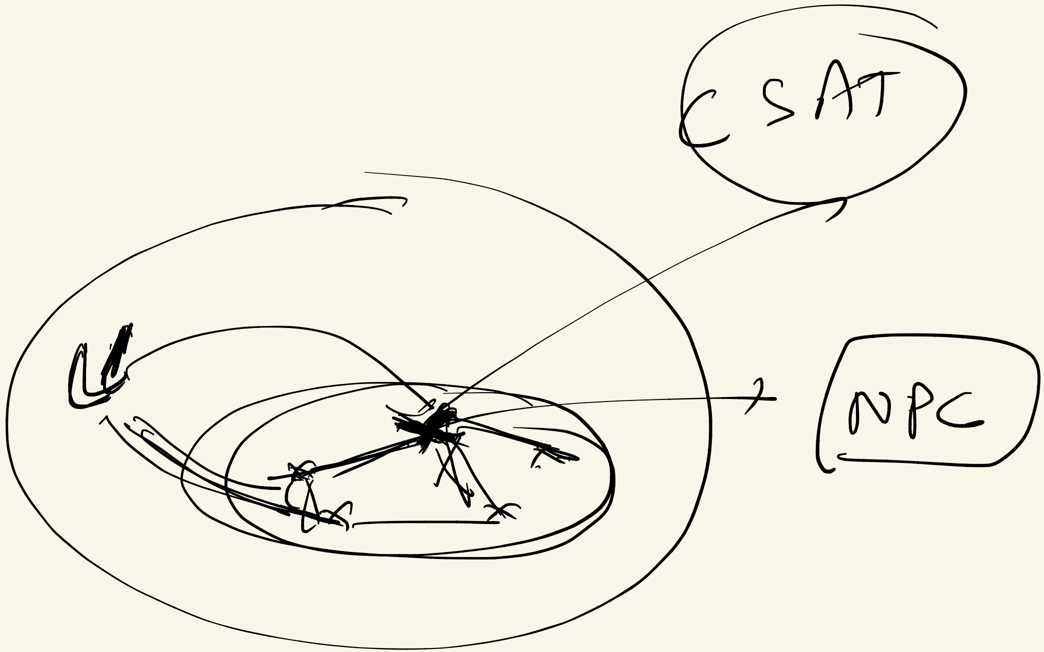
G_2

$$SPATH = \left\{ \begin{array}{l} \{G_1 = (V, E), x, z, 1\} \\ \vdots \\ \{G_1 = (V, E), x, y, 2\} \end{array} \right\}$$



L Decision --

$$\left\{ \forall (x, \mathrm{SPATH})\ \mathrm{SPATH}(x) = 1 \right\}$$



$L' \leq_p CSAT$

$L' \leq_p CSAT$

$$SAT \leq_p CSAT$$

$$3CNF\ SAT \leq CSAT$$

First NPC Problem

COOK- LEVIN THEOREM

Stephen COOK

Levin

$\boxed{1962}$

U Toronto

$\boxed{\text{CIRCUIT-SAT}}$

$\boxed{\text{C SAT}} \in \boxed{NPC}$

$2^3 = 8$

x_1	0	0	0	0	1	1	1	1
x_2	0	0	1	1	0	0	1	1
x_3	0	1	0	1	0	1	0	1

2^n $\boxed{n=3}$

$\boxed{\text{CSAT}}$ (8) $\boxed{x_1 \quad x_2 \quad x_3}$

$\boxed{\text{CSAT} \notin P}$

$(1, 0, 0)$

(x_i, x_j, x_k) ~ $\boxed{O = 1}$

$\boxed{O \text{ output} = 1}$

$CSAT \in NPC$

(1) * $CSAT \in NP$

- Polytime verifi

(2) * $CSAT \in NP$-hard

$L' \in NP$

$\boxed{L' \leq_P CSAT}$

L'

NP

NPC_0

NPC_0

$L' \le_p \underline{NPC_0}$

$L \in$ NP-hard $\qquad NPC_0 \le_p L$

Prove $L \in NPC$

① $L \in NP$ (Prove first!)

② $L \in$ NP-hard

$NPC_0 \leq_p L \to L \in$ NP-hard

SAT is NPC

1. SAT $\in$ NP ✓

2. SAT $\in$ NP-hard

$\mathrm{NPC}_0 \leq_p \mathrm{SAT}$

$\boxed{\mathrm{CSAT} \leq_p \mathrm{SAT}}$ ✓

$$\phi = (x_1, x_2 \cdots x_n)$$

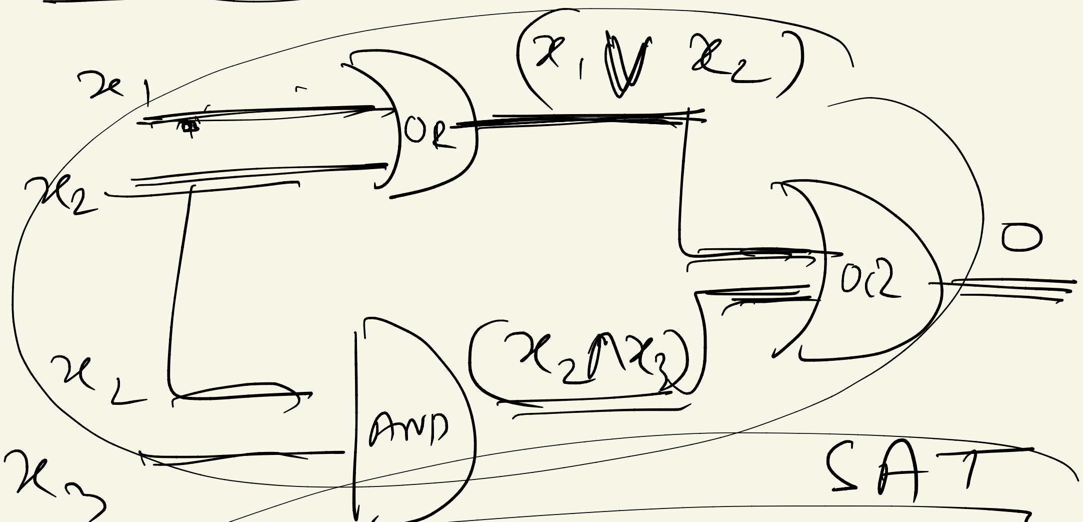
"m" clauses

K literal

$$\underbrace{K + K + K \cdots + K}_{m}$$

$$\boxed{K^m} \qquad \phi \approx (n^m)$$

CSAT $\le_p$ SAT



SAT

$$O = \underbrace{(x_1 \vee x_2) \vee (x_2 \wedge x_3)}_{\phi}$$

3-CNF

$\phi =$ 1 AND of ORs

2 each clause 3 literals

$$\phi_1 = \underbrace{(\bar{x}_1 \vee \bar{x}_2)}_{2} \wedge \underbrace{(\bar{x}_2 \vee \bar{x}_1)}_{2}$$

$\phi_1 \notin$ 3CNF

$$\phi = (x_1 \lor \overline{x_2} \land \overline{x_3}) \times 3cr$$

$$\land \; (x_1 \lor \overline{x_2} \lor x_3)$$

$$\phi \in \text{3-CNF}$$

3-CNF $\in$ NPC

1 3-CNF $\in$ NP

2 3-CNF $\in$ NP-hard

$$\boxed{NPC_0 \leq_p \text{3-CNF}}$$

$\Phi = 1/0$

$$\phi = \underbrace{\qquad}_{3} \wedge \underbrace{\qquad}_{3} \wedge \underbrace{\qquad}_{3}$$

(x_1, x_2, x_3)

$= O(3^k) \; \ldots$

$$NPC_0 \le_p \text{3-CNF}$$

$$\overset{?}{=}$$

$$SAT \le_p \text{3-CNF}$$

CSAT

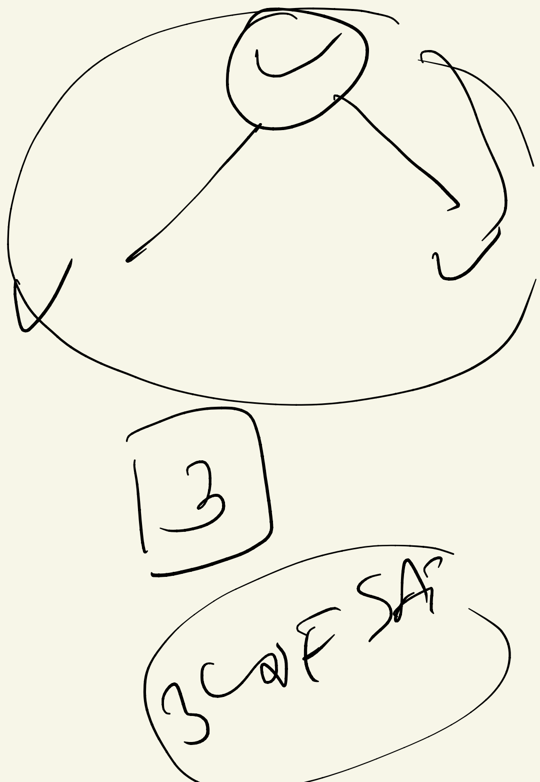
$\downarrow$

SAT

$$\phi = (x_1)\ (x_1 \cup x_2)\ (x_1 \cup \_\_)$$

AND of OR's

3 literal


3

CLIQUE

$\{ G = (V, E), k$

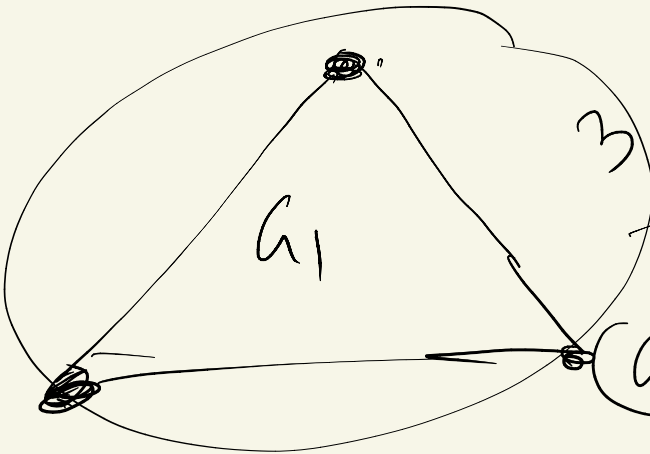
is there a Clique

of "k" nodes in G $\}$

Clique = set of nodes

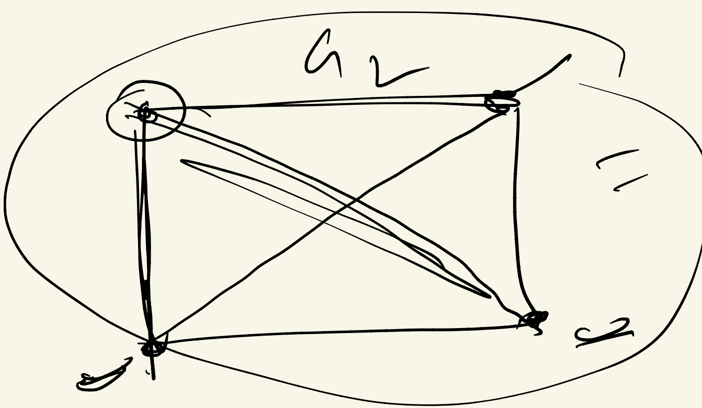
$\forall$ all nodes are connected

$(G_1, 3) \in$ CLIQUE



3 CLIQUE

$(G_2, 4) \in$ CLIQUE



4 CLIQUE

CLIQUE $\in$ NPC

1. CLIQUE $\in$ NP

2. CLIQUE $\in$ NP-hard

$$\boxed{NPC_0 \leq_p \text{CLIQUE}}$$

3CNF-SAT $\leq_p$ CLIQUE $\subset$ SAT

$\phi = \underbrace{(x_1 \lor x_2 \lor \neg x_3)}_{1} \text{ AND } \underbrace{(x_1 \lor x_2 \lor x)}_{2} \text{ AND } (-) \cdots \underbrace{(-)}_{m}$

OR

OR

SAT

3CNF-SAT

"m" clauses = 3 literal

Formula $\rightarrow$ Graph

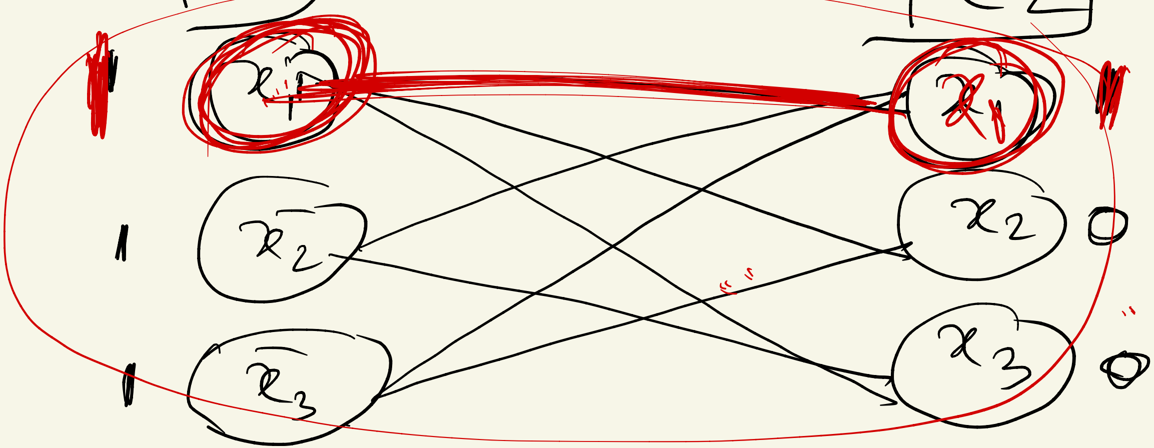
$\phi = 2$- clauses

$x_1 \quad x_2 \quad x_3$

$(x_1 \text{ OR } x_2 \text{ OR } x_3)$ AND

$(x_1 \text{ OR } x_2 \text{ OR } x_3)$

C_1 $\quad$ C_2



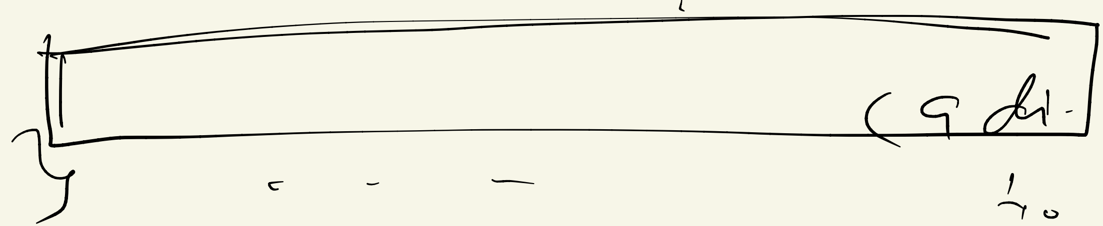
$\rightarrow C(2)$

$f(x)$

USING FCT, describe the Language of SSW

1. $\Sigma = \{0, 1, \ldots 9\}$

2. $\Sigma^* = \{0, 1, 2, \ldots 9 \ (1\text{-di}),$
$00, 11, 01, 12, \ldots (2\text{-di}), \ldots\}$

3. 

L_{Phone}

$= \{$ all 10 digit #s $\subseteq \Sigma^*$

s.t. the number does

not start with '0' $\}$

Using Formal Language Theory, describe the language that <u>accepts</u> all PHONE NUMBERS in USA.

<u>Step 1</u>: Define alphabet : $\Sigma = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$

<u>Step 2</u>: Define Universe of strings

$$\Sigma^* = \{0, 1, 2, 3 \ldots 9 \longrightarrow \text{length } 1$$

$$00, 01, 02, \ldots 99 \longrightarrow \text{length } 2$$

$$\vdots$$

$$\underbrace{0000000000}, \ldots, \longrightarrow \text{length } 10$$

$$\vdots \qquad \text{INFINITE STRINGS}$$

$$\}$$

Step3: Language L-PHONE can be defined as

$$\text{L-PHONE} = \{ s \in \Sigma^* : s \text{ does not start with } 0 \text{ AND length of } s\ |s| = 1$$