# NP-hardness and NP completeness.

NP: Problems that can be efficiently verified

P: Problems that can be solved eff.

Circuit SAT: NP-hard

NP-hard: Problems when solved in PTIME

$\Downarrow$

NP = P.

$\Pi$ is NP-hard if any problem in NP can be solved with $\Pi$ as a subroutine.

"Subroutine".

## Reductions

$\Pi' \leq_p \Pi$

k-clique

Hamiltonian cycle

$\phi \in \Pi'$ $\longrightarrow$ $\phi' \in$ Ckt-SAT.

instance $n$

$$G^{\,n} \xrightarrow{T} \text{Ckt-SAT problem}^{\,n^c} \; C_G$$

Say $N$ sized ckt has a $f(N)$ time ago.

" $G$ has a vertex cover of size at most $k$ if and only if $C_G$ has a satisfiable assignment".

" $|C_G|$ is at most poly(n)".

$\Rightarrow$ $\Pi'$ w.r.t insta G can be solved in

$$|T| + f(n^c).$$

Further if $|T|$ is poly(n); and $f(N) = N^{O(1)}$

then $G \in \Pi'$ can be solved in $n^{O(1)}$.

"Ckt SAT is NP-hard"

$\Updownarrow$

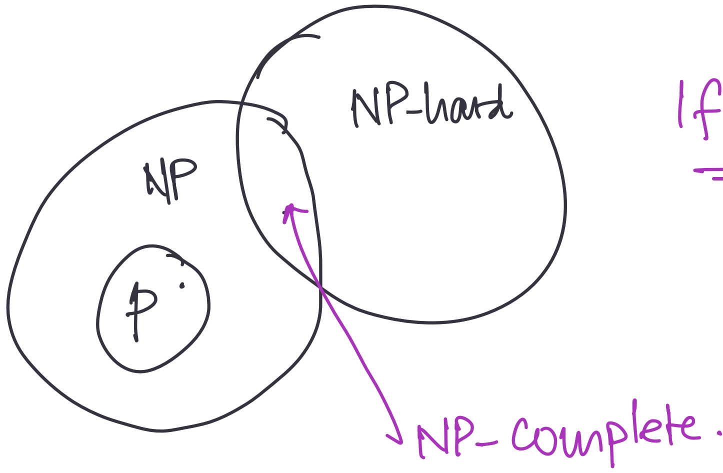
" Every instance of every problem in NP can be reduced to an instance of ckt satisfiability of size at most poly(|instance|), and reduction takes at most poly(|instance|) time".

- Traveling Salesperson problem
- Minimum vertex Cover $\equiv$ Maximum Independent Set
- k-Vertex Cover, k-Clique.

NP-complete : Problems that are NP-hard and also in NP.

" Ckt SAT is NP-complete"

To show $P \neq NP$:
- Suff to show that $\exists$ problem $NP \setminus P$ that admits no polytime algos.




<u>If $P \neq NP$</u>

<u>Suff</u>:

- Polynomial time algorithms for any NP-hard problem $\Rightarrow P = NP$.

- Proving that a NP-complete problem has no polytime algorithms $\Rightarrow P \neq NP$.

- Proving that a problem in $NP \setminus (P \cup$ NP-complete$)$ admits no polytime algo $\Rightarrow P \neq NP$

- Proving that a problem in $NP \setminus (P \cup$ NP-complete$)$ admits a polytime algo $\not\Rightarrow P = NP$

whereas

- Proving that a NP-complete problem has a polytime algorithm $\Rightarrow P = NP$.

<u>3-SAT</u> (3-CNF): A Boolean formula that can be expressed as a conjunction of clauses each of which is a disjunction of 3 literals.

$C_1 \wedge C_2 \wedge C_3 \cdots$

$C_i = (x_{i_1} \vee \bar{x}_{i_2} \vee x_{i_3})$

# Reduction:

**Ckt SAT**

Boolean circuit C
$\overset{?}{\exists}$ a satisfying assignment

**3SAT**

Given a 3-CNF, is there a satisfying assignment.

$$S_1, S_2, \ldots, S_k \subseteq \{1, 2, \ldots, n\}$$

$$L_i := \left. \bigwedge_{j \in S_i} x_j = 0 \right\} \equiv \bigvee_{j \in S_i} \bar{x}_j \equiv 1$$

$$\bigwedge_{i=1}^{k} \left( \bigvee_{j \in S_i} \bar{x}_j \right)$$

**Claim:** 3-SAT is NP-hard.

- Maximum Independent set is NP-hard.

"Algorithms" — Jeff Erickson.
(UIUC)