Polynomial Time Reduction

Definition
Π is a decision problem if it only has a “yes” or “no” answer.

Definition
Given two decision problems Π₁, Π₂ we say Π₂ is polynomial time reducible to Π₁ iff there exists a polynomial time algorithm f that transforms any instance X of Π₂ to an instance f(X) of Π₁ such that:

\((X \text{ is a “yes” instance of } \Pi_2) \iff (f(X) \text{ is a “yes” instance of } \Pi_1)\)

We write Π₂ ≤ₚ Π₁ to denote “Π₂ is polynomial time reducible to Π₁”.
Some Examples

- **3-SAT** $\leq_p$ **CLIQUE**
  - We saw a reduction in the last lecture.

- **INDEPENDENT-SET** $\leq_p$ **CLIQUE**
  - A graph $G$ has an independent set of size $k$ iff the complement of the graph $\bar{G}$ has a clique of size $k$.

- **VERTEX-COVER** $\leq_p$ **SET-COVER**
  - A graph has a vertex cover of size $k$ iff $(S_1, S_2, \ldots, S_n)$ has a set cover of size $k$ where $S_i$ is all edge incident to node $i$.

- **VERTEX-COVER** $\leq_p$ **INDEPENDENT-SET**
  - A graph $G$ has a vertex cover of size $k$ iff $G$ has an independent set of size $n - k$. (This follows because all the nodes not in a vertex cover form an independent set.)
Outline

NP Completeness
**P and NP Definitions**

**Definition (P)**

$\Pi \in P$ iff there exists a polynomial time algorithm $A$ such that:

$$(X \text{ is a “yes” instance of } \Pi) \iff (A(X) = \text{“yes”})$$

**Definition (NP)**

$\Pi \in NP$ iff there exists a polynomial time algorithm $A$ whose input has two parts, the input to $\Pi$ and some extra “advice” (also known as a “certificate” or “witness”), such that:

$$(X \text{ is a “yes” instance of } \Pi) \implies (\exists Y : |Y| = \text{poly}(|X|), A(X, Y) = \text{“yes”})$$

$$(X \text{ is a “no” instance of } \Pi) \implies (\not\exists Y : |Y| = \text{poly}(|X|), A(X, Y) = \text{“yes”})$$
Example: Clique

- Input: Given graph $G = (V, E)$ and integer $k$.
- Question: Does $G$ contain a clique of size $k$?

Lemma

*Clique is in NP.*

Proof.

1. Suppose the witness $Y$ encodes a set of $k$ nodes in $V$ and $A(G, Y)$ checks if the induced graph on $Y$, $G[Y]$ is a clique.
2. $A$ is a polynomial time algorithm.
3. If there exists a clique of size $k$, there exists $Y$ of size $k$ such that $A(G, Y)$ outputs “yes”
4. If there doesn’t exist a clique of size $k$, there doesn’t exist $Y$ of size $k$ such that $A(G, Y)$ outputs “yes”

Example for a problem that is not known to be in NP: Is a quantified boolean formula, e.g., $\forall x \exists y \exists z , ((x \lor z) \land y)$, true?
NP-Completeness

Definition
A decision problem $\Pi$ is NP-Hard iff for all $\Pi' \in NP$, $\Pi' \leq_P \Pi$.

Definition
A decision problem $\Pi$ is NP-Complete iff it is both NP-Hard and in NP.

Remark 1: If $\Pi$ is NP-Complete and $\Pi \in P$ then $P = NP$

Remark 2: In 1971, Cook showed 3-SAT is NP-Complete. Because

\[ \text{CLIQUE} \in NP \text{ and } 3\text{-SAT} \leq_P \text{CLIQUE} \]

we now know CLIQUE is NP-Complete.
Summary of NP Completeness and Reductions

1. Decision problem $\Pi$ is in $P$ if there is a polynomial time algorithm that correctly answers $\Pi$.
2. Decision problem $\Pi$ is in $NP$ if there is a polynomial time algorithm that takes advice:
   - If the answer should be “yes”, then there exists advice that leads the algorithm to output “yes”
   - If the answer is “no”, then there doesn’t exist advice that would lead the algorithm to output “yes”
3. A problem $\Pi$ is NP-hard if for any $\Pi' \in NP$: $\Pi' \leq_P \Pi$
4. A problem $\Pi$ is NP-complete if $\Pi \in NP$ and $\Pi$ is NP-hard.
5. To show $\Pi$ is NP-complete it suffices to show that
   - $\Pi$ is in $NP$
   - $\Pi' \leq_P \Pi$ for some $\Pi'$ that is already known to be NP-hard
6. It’s widely believed the $P \neq NP$ but finding a polynomial time algorithm for any NP-hard problem would prove $P = NP$. 