Last time
- Class NP
- The P vs. NP question

Today
- Polynomial-time reductions
- NP-completeness
Classify Problems

- **Desiderata:** classify problems according to those that can be solved in polynomial-time and those that cannot.
- Some problems *provably require exponential time* (Chapter 9):
  - Given a Turing machine, does it halt in at most $k$ steps?
  - Given a board position in an $n$-by-$n$ generalization of chess, can black guarantee a win?
- **Frustrating news:** huge number of fundamental problems have defied classification for decades.
- **Chapters 7.4-7.5 (NP-completeness):** Show that these fundamental problems are "computationally equivalent" and appear to be different manifestations of one really hard problem.
Given languages $A$ and $B$, $A \leq_p B$ if there is a *poly-time* computable function $f$, such that for all strings $w$, $w \in A$ iff $f(w) \in B$.

Polynomial-time reductions are the major tool we have to understand P and NP.
Implication of poly-time reductions

**Theorem.** If $A \leq_p B$ and $B \in \mathbf{P}$ then $A \in \mathbf{P}$.

(So, if $A \leq_p B$ and $A \notin \mathbf{P}$ then $B \notin \mathbf{P}$.)

**Theorem.** If $A \leq_p B$ and $B \leq_p C$ then $A \leq_p C$.

(Poly-time reductions compose.)
Basic reduction strategies

- Reduction by simple equivalence.
- Reduction from special case to general case.
- Reduction by encoding with gadgets.
Independent Set

Given an undirected graph $G$, an independent set in $G$ is a set of nodes, which includes at most one endpoint of every edge.

\[
\text{INDEPENDENT SET} = \{ (G, k) \mid G \text{ is an undirected graph which has an independent set with } k \text{ nodes} \}
\]

- Is there an independent set of size $\geq 6$?
  - Yes.

- Is there an independent set of size $\geq 7$?
  - No.
Given an undirected graph $G$, a **vertex cover** in $G$ is a set of nodes, which includes at least one endpoint of every edge.

$\text{VERTEX COVER} = \{ (G, k) \mid G \text{ is an undirected graph which has a vertex cover with } k \text{ nodes} \}$

- Is there vertex cover of size $\leq 4$?  
  - Yes.  
- Is there a vertex cover of size $\leq 3$?  
  - No.
Claim. S is an independent set iff V – S is a vertex cover.

• \( \Rightarrow \)
  – Let S be any independent set.
  – Consider an arbitrary edge \((u, v)\).
  – S is independent \( \Rightarrow u \notin S \) or \( v \notin S \) \( \Rightarrow u \in V - S \) or \( v \in V - S \).
  – Thus, \( V - S \) covers \((u, v)\).

• \( \Leftarrow \)
  – Let \( V - S \) be any vertex cover.
  – Consider two nodes \( u \in S \) and \( v \in S \).
  – Then \((u, v) \notin E \) since \( V - S \) is a vertex cover.
  – Thus, no two nodes in S are joined by an edge \( \Rightarrow S \) independent set. •
**Theorem.** INDEPENDENT-SET $\leq_p$ VERTEX-COVER.

**Proof.** “On input $\langle G, k \rangle$, where $G$ is an undirected graph and $k$ is an integer,

1. Output $\langle G, n - k \rangle$, where $n$ is the number of nodes in $G$.”

**Correctness:**

- $G$ has an independent set of size $k$ iff it has a vertex cover of size $n - k$.
- Reduction runs in linear time.
Basic reduction strategies

- Reduction by simple equivalence.
- Reduction from special case to general case.
- Reduction by encoding with gadgets.
Set Cover

Given a set $U$, called a *universe*, and a collection of its subsets $S_1, S_2, \ldots, S_m$, a set cover of $U$ is a subcollection of subsets whose union is $U$.

- **SET COVER**$= \{ \langle U, S_1, S_2, \ldots, S_m; k \rangle \mid U \text{ has a set cover of size } k \}$

U has a set cover of size $k$

- Sample application.
  - $m$ available pieces of software.
  - Set $U$ of $n$ capabilities that we would like our system to have.
  - The $i$th piece of software provides the set $S_i \subseteq U$ of capabilities.
  - Goal: achieve all $n$ capabilities using fewest pieces of software.

$$U = \{1, 2, 3, 4, 5, 6, 7\}$$

$$k = 2$$

$$S_1 = \{3, 7\} \quad S_4 = \{2, 4\}$$

$$S_2 = \{3, 4, 5, 6\} \quad S_5 = \{5\}$$

$$S_3 = \{1\} \quad S_6 = \{1, 2, 6, 7\}$$
**Theorem.** \textsc{vertex-cover} \( \leq \text{P} \) \textsc{set-cover}.

**Proof.** “On input \( \langle G, k \rangle \), where \( G = (V, E) \) is an undirected graph and \( k \) is an integer,

1. Output \( \langle U, S_1, S_2, \ldots, S_m; k \rangle \), where \( U = E \) and

\[ S_v = \{ e \in E \mid e \text{ incident to } v \} \]

**Correctness:**

- \( G \) has a vertex cover of size \( k \) iff \( U \) has a set cover of size \( k \).
- Reduction runs in linear time.
Basic reduction strategies

• Reduction by simple equivalence.
• Reduction from special case to general case.
• Reduction by encoding with gadgets.
• **Boolean variables:** variables that can take on values T/F (or 1/0)

• **Boolean operations:** $\lor$, $\land$, and $\neg$

• **Boolean formula:** expression with Boolean variables and ops

SAT = $\{\langle \Phi \rangle \mid \Phi$ is a satisfiable Boolean formula$\}$

• **Literal:** A Boolean variable or its negation.

• **Clause:** OR of literals.

• **Conjunctive normal form (CNF):** AND of clauses.

3SAT = $\{\langle \Phi \rangle \mid \Phi$ is a satisfiable Boolean CNF formula, where each clause contains exactly 3 literals$\}$

Ex:

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

Yes: $x_1 = true, x_2 = true, x_3 = false$. 

---

Ex: $C_j = x_1 \lor \overline{x_2} \lor x_3$

$\Phi = C_1 \land C_2 \land C_3 \land C_4$

each corresponds to a different variable
3SAT reduces to INDEPENDENT SET

Theorem. $3$-SAT $\leq_P$ INDEPENDENT-SET.

Proof. “On input $\langle \Phi \rangle$, where $\Phi$ is a 3CNF formula,

1. Construct graph $G$ from $\Phi$
   - $G$ contains 3 vertices for each clause, one for each literal.
   - Connect 3 literals in a clause in a triangle.
   - Connect literal to each of its negations.

2. Output $\langle G, k \rangle$, where $k$ is the number of clauses in G.”

$$k = 3$$

$$\Phi = (\overline{x_1} \lor x_2 \lor \overline{x_3}) \land (x_1 \lor \overline{x_2} \lor x_3) \land (\overline{x_1} \lor x_2 \lor x_4)$$
Correctness. Let $k = \# \text{ of clauses and } \ell = \# \text{ of literals in } \Phi$.

$\Phi$ is satisfiable iff $G$ contains an independent set of size $k$.

- $\Rightarrow$ Given satisfying assignment, select one true literal from each triangle. This is an independent set of size $k$.

- $\Leftarrow$ Let $S$ be an independent set of size $k$.
  - $S$ must contain exactly one vertex in each triangle.
  - Set these literals to true, and other literals in a consistent way.
  - Truth assignment is consistent and all clauses are satisfied.

Run time. $O(k + \ell^2)$, i.e. polynomial in the input size.
Summary

• Basic reduction strategies.
  – Simple equivalence: \( \text{INDEPENDENT-SET} \equiv_{P} \text{VERTEX-COVER} \).
  – Special case to general case: \( \text{VERTEX-COVER} \leq_{P} \text{SET-COVER} \).
  – Encoding with gadgets: \( 3\text{-SAT} \leq_{P} \text{INDEPENDENT-SET} \).

• Transitivity. If \( X \leq_{P} Y \) and \( Y \leq_{P} Z \), then \( X \leq_{P} Z \).

• Proof idea. Compose the two algorithms.

• Ex: \( 3\text{-SAT} \leq_{P} \text{INDEPENDENT-SET} \leq_{P} \text{VERTEX-COVER} \leq_{P} \text{SET-COVER} \).
A language $B$ is **NP-complete** if

1. $B \in \text{NP}$
2. $B$ is **NP-hard**, i.e.,
   every language in NP is poly-time reducible to $B$. 

---

**Diagram:**
- $P$ is a subset of $NP$
- $B$ is a subset of $NP$
- $B$ is reducible to every language in $NP$
Implication of poly-time reductions

Theorem. If

- B is \textbf{NP}-complete,
- \(C \in \textbf{NP}\) and
- \(B \leq_p C\)

then \(C\) is \textbf{NP}-complete.
Implication of poly-time reductions

**Theorem.** If

- $B$ is $\text{NP}$-complete,
- $C \in \text{NP}$ and
- $B \leq_p C$

then $C$ is $\text{NP}$-complete.

**Theorem.** If $B$ is $\text{NP}$-complete and $B \in \text{P}$ then $\text{P} = \text{NP}$.

(Or, if $B$ is $\text{NP}$-complete and $\text{P} \neq \text{NP}$ then there is no poly-time algorithm for $B$.)