Review

- P – class of problems with polytime algorithm.
- NP – class of problems with polytime certifier.

Example

<table>
<thead>
<tr>
<th>Problem ($X$)</th>
<th>INDEPENDENT-SET</th>
</tr>
</thead>
<tbody>
<tr>
<td>Instance ($s$)</td>
<td>Graph $G$ and number $k$</td>
</tr>
<tr>
<td>Algorithm ($A$)</td>
<td>No poly-time algorithm known</td>
</tr>
<tr>
<td>Hint ($t$)</td>
<td>Which nodes are in the answer?</td>
</tr>
<tr>
<td>Certifier ($C'$)</td>
<td>Are those nodes independent and size $k$?</td>
</tr>
</tbody>
</table>

NP-Complete

- NP-complete = a problem $Y \in \text{NP}$ with the property that $X \leq_p Y$ for every problem $X \in \text{NP}!$

Cook-Levin Theorem: In 1971, Cook and Levin independently showed that particular problems were NP-Complete.
- We’ll look at CIRCUIT-SAT as canonical NP-Complete problem.
**Circuit-SAT**

**Problem:** Given a circuit built of **And**, **Or**, and **Not** gates with some hard-coded inputs, is there a way to set remaining inputs so the output is 1?

Satisfiable? Yes. Set inputs: 1, 1, 0.

**Circuit-SAT**

**Cook-Levin Theorem** Circuit-SAT is NP-Complete.

**Proof Idea:** encode arbitrary certifier $C(s, t)$ as a circuit

▶ If $X \in \text{NP}$, then $X$ has a poly-time certifier $C(s, t)$:

- $s$ is **Yes** instance $\iff \exists t$ such that $C(s, t)$ outputs **Yes**
- Construct a circuit where $s$ is hard-coded, and circuit is satisfiable iff $\exists t$ that causes $C(s, t)$ to output **Yes**
- $s$ is **Yes** instance $\iff$ circuit is satisfiable
- Algorithm for **Circuit-Sat** implies an algorithm for $X$

**A Circuit-SAT reduction**

See Independent Set example in other slides

**A Circuit-SAT reduction**

▶ Vertex Cover – Does $G$ have VC of size at most $k$? (Counting gadget is an example for $v_3, v_4$ only)
Suppose \( X \) is in NP.

**Fact:** If \( Y \) is NP-complete and \( Y \leq_p X \), then \( X \) is NP-complete.

Want to prove problem \( X \) is NP-complete
- Check \( X \in \text{NP} \).
- Choose known NP-complete problem \( Y \).  
- Prove \( Y \leq_p X \).

Theorem: 3-SAT is NP-Complete.
- In NP? Yes, check satisfying assignment in poly-time.
- Can show that Circuit-SAT \( \leq_p \) 3-SAT

From Circuit-SAT to 3-SAT

To show that Circuit-SAT \( \leq_p \) 3-SAT, we'll show how to construct a 3-SAT formula to model an arbitrary Circuit-SAT instance.

Example.
Reduction: \textsc{Circuit-Sat} \leq_P \textsc{3-Sat}

- One variable $x_v$, per circuit node $v$, plus clauses to enforce circuit computations
- Express Negation, OR, and AND nodes using several implications of the form $A \Rightarrow B$ (which is equivalent to the clause $\neg A \lor B$)
- Negation node: $x_v = \neg x_u$
  - $x_u \Rightarrow \neg x_v$
  - $\neg x_u \Rightarrow x_v$
- OR node: $x_v = x_u \lor x_w$
  - $x_u \Rightarrow x_v$
  - $x_w \Rightarrow x_v$
  - $x_v \Rightarrow x_u \lor x_w$
- AND node: $x_v = x_u \land x_w$
  - $x_v \Rightarrow x_u$
  - $x_v \Rightarrow x_w$
  - $\neg x_v \Rightarrow \neg x_u \lor \neg x_w$

Reduction: \textsc{Circuit-Sat} \leq_P \textsc{3-Sat}

- Clause $C = x_v$ for input bits $v$ fixed to one
- Clause $C = \neg x_v$ for input bits $v$ fixed to zero
- Clause $C = x_o$ for output bit
- This formula is satisfiable iff circuit is satisfiable.
- Deal with clauses of size 1 and 2 by introducing two new variables and clauses that force them to be equal to zero.

Clicker

Which of the following statements is NOT true?

A. SAT \leq_P 3-SAT  
B. 3-SAT \leq_P SAT  
C. k-SAT \leq_P SAT for all $k \geq 2$  
D. k-SAT is NP-complete for all $k \geq 2$

NP-Complete Problems So Far

Theorem: \textsc{IndependentSet}, \textsc{VertexCover}, \textsc{SetCover}, SAT, 3-SAT are all NP-Complete.
Traveling Salesman Problem

- TSP: Given \( n \) cities and distance function \( d(i, j) \), is there a tour that visits all cities with total distance less than \( D \)?
  - Tour: ordering of cities \( i_1, i_2, \ldots, i_n \) with \( i_1 = 1 \)
  - Distance is \( \sum_{j=1}^{n-1} d(i_j, i_{j+1}) + d(i_n, 1) \)
- Applications: traveling salesman, moving robotic arms
- Let’s prove a simpler problem is NP-complete, and then use it to show TSP is NP-complete.

Hamiltonian Cycle Problem

- **HamCycle** - Hamiltonian Cycle. Given directed graph \( G = (V, E) \), is there a cycle that visits each vertex exactly once?
  - \( v_1, v_3, v_2, v_5, v_4, v_6 \) is a Hamiltonian Cycle

**Theorem.** **HamCycle** is NP-Complete.
- It is in NP.
- Need to reduce from some NP-Complete problem. Which one?
- **Claim.** 3-SAT \( \leq_P \) **HamCycle**.

**Reduction** has two main parts.
- Make a graph with \( 2^n \) Hamiltonian cycles, one per assignment.
- Augment graph with clause gadgets to ensure assignments satisfy all clauses
Correspondence between Hamiltonian cycles and truth assignments

- \( x_i = 1 \) \iff traverse \( P_i \) from \( L \to R \)

Details

- \( n \) rows (bidirected paths) \( P_1, \ldots, P_n \) (one per variable)
- Row has \( 3m + 3 \) vertices, connected to neighbors in forward/backward direction
- First and last vertex of row \( i \) connected to first and last of \( i+1 \)
- Source \( s \) connected to first and last of row 1
- First and last of row \( n \) connected to \( t \)
- Edge \((t, s)\)
- Skeleton has \( 2^n \) possible Hamiltonian Cycles, corresponding to truth assignments to \( x_1, \ldots, x_n \)
  - Traverse \( P_i \) L to R \iff \( x_i = 1 \)
  - Traverse \( P_i \) R to L \iff \( x_i = 0 \)
Reduction: Clause Gadgets

For each clause $C_\ell$ construct gadget to restrict possible truth assignments

- New node $c_\ell$
- If $x_i \in C_\ell$
  - Add edges $(v_i, 3\ell, c_\ell)$ and $(c_\ell, v_i, 3\ell + 1)$
  - $c_\ell$ can be visited during L to R traversal of $P_i$
- If $\neg x_i \in C_\ell$
  - Add edges $(v_i, 3\ell + 1, c_\ell)$ and $(c_\ell, v_i, 3\ell)$
  - $c_\ell$ can be visited during R to L traversal of $P_i$

Proof of Correctness

Given a satisfying assignment, construct Hamiltonian Cycle

- If $x_i = 1$ traverse $P_i$ from $L \rightarrow R$, else $R \rightarrow L$
- Each $C_\ell$ is satisfied, so one path $P_i$ is traversed in the correct direction to "splice" $c_\ell$ into our cycle
- The result is a Hamiltonian Cycle

Given Hamiltonian cycle, construct satisfying assignment:

- If cycle visits $c_\ell$ from row $i$, it will also leave to row $i$ because of "buffer" nodes
- Therefore, ignoring clause nodes, cycle traverses each row completely from $L \rightarrow R$ or $R \rightarrow L$
- Set $x_i = 1$ if $P_i$ traversed $L \rightarrow R$, else $x_i = 0$
- Every node $c_j$ visited $\Rightarrow$ every clause $C_j$ is satisfied

Traveling Salesman

TSP. Given $n$ cities and distance function $d(i, j)$, is there a tour that visits all cities with total distance less than $D$?

Theorem. TSP is NP-Complete

- Clearly in NP.
- Reduction? From Ham-Cycle

Clicker

We want to show that Ham-Cycle $\leq_P$ TSP. How can we do so?

Given a Ham-Cycle instance $G = (V, E)$ make TSP instance with one city per vertex and...

A. $d(v_i, v_j) = 1$ if $(v_i, v_j) \in E$, else 2. Tour distance: $\leq n$?
B. $d(v_i, v_j) = 2$ if $(v_i, v_j) \in E$, else 1. Tour distance: $\leq n$?
C. $d(v_i, v_j) = 1$ if $(v_i, v_j) \in E$, else 2. Tour distance: $\leq m$?
Reduction from Ham-Cycle to TSP

Given HamCycle instance $G = (V, E)$ make TSP instance
- One city per vertex
- $d(v_i, v_j) = 1$ if $(v_i, v_j) \in E$, else 2

Claim: there is a tour of distance $\leq n$ if and only if $G$ has a Hamiltonian cycle
- A Hamiltonian cycle clearly gives a tour of length $n$
- A tour of length $n$ must travel $n$ hops of length 1, which corresponds to a Hamiltonian cycle

Ham-Path

Similar to Hamiltonian Cycle: is there a path that visits every vertex exactly once?

Theorem. Ham-Path is NP-Complete.

Two proofs:
- Modify 3-SAT to Ham-Cycle reduction.
- Show that Ham-Cycle reduces to Ham-Path

NP-Complete Problems