Wrapping Up Our NP Hardness Reductions
Grouping Problems

• The textbook does a good job grouping problems

  • Packing problems (Independent set)
  • Covering problems (Vertex cover, set cover)
  • 8.5: Sequencing Problems (Hamiltonian cycle/path, traveling salesmen)
  • 8.6 Partitioning Problems (3-dimensional matching)
  • 8.7 Graph Coloring (3-coloring)
  • 8.8 Numerical Problems (Subset sum, knapsack)

• If your problem seems to fit into one of these categories, it is a reasonable strategy to pick a problem from the same category as a reduction candidate
Today’s Plan

- We are finishing our in-class exploration of reductions
  - We’ll show that subset sum is NP-complete
- *Didn’t we say that subset sum had an $O(nW)$ dynamic programming algorithm, just like knapsack?*
  - Yes, but $O(nW)$ is not polynomial with respect to the input size (representation); $W$ is a value!
- Steps to show subset sum is NP-complete?
  - Show subset sum is in NP
  - Reduce a known NP-complete problem to it (in poly-time)
  - Prove yes instances map to yes instances (both ways)
SUBSET-SUM is NP Complete:

Vertex-Cover $\leq_p$ SUBSET-SUM
Subset Sum Problem

• **SUBSET-SUM.**
  Given $n$ positive integers $a_1, \ldots, a_n$ and a target integer $T$, is there a subset of numbers that adds up to exactly $T$

• Step 1: show that **SUBSET-SUM** $\in$ NP
  - Certificate: a set of numbers
  - Poly-time verifier: checks if the set is a subset of the input integers, and if so, that the set sums exactly to $T$

• Step 2: prove that **SUBSET-SUM** is **NP hard** by reducing a known NP-complete problem to it
  - We’ll reduce from vertex cover
Map the Problems

**Vertex Cover**

What is a possible solution?

A selection of vertices to be in vertex cover $C$

What is the requirement?

$C$ must contain at most $k$ vertices

What are the restrictions?

If $(u, v) \in E$, then either $u$ or $v$ must be in $C$

**Subset Sum**

A subset of integers $S \subseteq \{a_1, \ldots, a_n\}$

What is the requirement?

Numbers in $S$ must sum to $T$

What are the restrictions?

$S$ must be a subset of input integers
Vertex Cover to Subset Sum

- **Theorem.** \( \text{VERTEX-COVER} \leq_p \text{SUBSET-SUM} \)

- **Proof.** Given a graph \( G \) with \( n \) vertices and \( m \) edges and a number \( k \), we construct a set of numbers \( a_1, \ldots, a_t \) and a target sum \( T \) such that \( G \) has a vertex cover of size \( k \) iff there is a subset of numbers that sum to \( T \)

\[
\langle G, k \rangle \xrightarrow{\text{Poly time Reduction}} \langle a_1, \ldots, a_n, T \rangle \xrightarrow{\text{Subset-Sum Algorithm}} \exists \text{ subset that sums to } T \xrightarrow{\text{Algorithm for Vertex Cover}} \exists \text{ vertex cover of size } k
\]
Vertex Cover to Subset Sum

- **Theorem.** \( \text{VERTEX-COVER} \leq_p \text{SUBSET-SUM} \)

- **Proof.** Given a graph \( G \) with \( n \) vertices and \( m \) edges and a number \( k \), we construct a set of numbers \( a_1, \ldots, a_t \) and a target sum \( T \) such that \( G \) has a vertex cover of size \( k \) iff there is a subset of numbers that sum to \( T \)

  - In our reduction, we’ll use two types of gadgets

    - when reducing a problem \( X \) to a problem \( Y \), a gadget is a small (partial) instance of problem \( Y \) that is used to "simulate" a feature of problem \( X \)

    - Gadget 1: integers that represent vertices of \( G \), and
    - Gadget 2: integers that represent edges of \( G \)
Vertex Cover to Subset Sum

• **Theorem.** VERTEX-COVER $\leq_p$ SUBSET-SUM

• **Reduction (idea)**

  • We'll create a set of $n + m$ integers, where we have one integer for every vertex, and one integer for every edge ($A = \{a_1, \ldots, a_n, a_{n+1}, \ldots, a_{n+m}\}$)

  • **Goals** when creating our vertex/edge gadgets:

    • Must force the selection of $k$ vertex integers: need to ensure that no other set of input integers can sum to $T$

    • Must force an edge covering: for every edge $(u, v)$, we need to ensure that our subset can't sum to $T$ unless either $u$ or $v$ is picked
Vertex Cover to Subset Sum

**Theorem.** VERTEX-COVER $\leq_p$ SUBSET-SUM

**Reduction.**
First, arbitrarily number the edges from 0 to $m - 1$. Then, create set of $n + m$ integers and a target value $T$ as follows:

- **Vertex integer** $a_v : m^{th}$ (most significant) bit is 1 and for $i < m$, the $i^{th}$ bit is 1 if $i^{th}$ edge is incident to vertex $v$

- **Edge integer** $b_{uv} : m^{th}$ digit is 0 and for $i < m$, the $i^{th}$ bit is 1 if this integer represents an edge $i = (u, v)$

- **Target value** $T = k \cdot 4^m + \sum_{i=0}^{m-1} 2 \cdot 4^i$

Note: Each integer is a $m + 1$-bit number in base four*
Vertex Cover to Subset Sum

- Example: consider the graph $G = (V, E)$ where $V = \{u, v, w, x\}$ and $E = \{(u, v), (u, w), (v, w), (v, x), (w, x)\}$

<table>
<thead>
<tr>
<th></th>
<th>5th</th>
<th>4th: (wx)</th>
<th>3rd: (vx)</th>
<th>2nd: (vw)</th>
<th>1st: (uw)</th>
<th>0th: (uv)</th>
</tr>
</thead>
<tbody>
<tr>
<td>$a_u$</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>$a_v$</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>$a_w$</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>$a_x$</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>$b_{uv}$</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>$b_{uw}$</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>$b_{vw}$</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>$b_{vx}$</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>$b_{wx}$</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

- If $k = 2$ then $T = 222222_4 = 2730$

$a_u := 111000_4 = 1344$
$a_v := 110110_4 = 1300$
$a_w := 101101_4 = 1105$
$a_x := 100011_4 = 1029$

$b_{uv} := 010000_4 = 256$
$b_{uw} := 001000_4 = 64$
$b_{vw} := 000100_4 = 16$
$b_{vx} := 000010_4 = 4$
$b_{wx} := 000001_4 = 1$
Correctness

- **Claim.** $G$ has a vertex cover of size $k$ if and only if there is a subset $X$ of corresponding integers that sums to value $T$

- $(\Rightarrow)$ Let $C$ be a vertex cover of size $k$ in $G$, define $X$ as
  \[ X := \{a_v \mid v \in C\} \cup \{b_i \mid \text{edge } i \text{ has exactly one endpoint in } C\} \]

<table>
<thead>
<tr>
<th></th>
<th>5th</th>
<th>4th: (wx)</th>
<th>3rd: (vx)</th>
<th>2nd: (vw)</th>
<th>1st: (uw)</th>
<th>0th: (uv)</th>
</tr>
</thead>
<tbody>
<tr>
<td>$a_u$</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>$a_v$</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>$a_w$</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>$a_x$</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>$b_{uv}$</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>$b_{uw}$</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>$b_{vw}$</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>$b_{vx}$</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>$b_{wx}$</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

$C = \{v, w\}$

\[
T = 222222_{10} = 2730
\]

\[
T = k \cdot 4^m + \sum_{i=0}^{m-1} 2 \cdot 4^i
\]
Correctness

- **Claim.** \( G \) has a vertex cover of size \( k \) if and only there is a subset \( X \) of corresponding integers that sums to value \( T \)

- \( (\Rightarrow) \) Let \( C \) be a vertex cover of size \( k \) in \( G \), define \( X \) as
  \[
  X := \{a_v \mid v \in C\} \cup \{b_i \mid \text{edge } i \text{ has exactly one endpoint in } C\}
  \]

\[
\begin{array}{cccccc}
\text{5th} & \text{4th: (wx)} & \text{3rd: (vx)} & \text{2nd: (vw)} & \text{1st: (uw)} & \text{0th: (uv)} \\
\hline
a_v & 1 & 0 & 1 & 1 & 0 & 1 \\
a_w & 1 & 1 & 0 & 1 & 1 & 0 \\
b_{uv} & 0 & 0 & 0 & 0 & 0 & 1 \\
b_{uw} & 0 & 0 & 0 & 0 & 1 & 0 \\
b_{vx} & 0 & 0 & 1 & 0 & 0 & 0 \\
b_{wx} & 0 & 1 & 0 & 0 & 0 & 0 \\
\end{array}
\]

\[
C = \{v, w\}
\]

\[
T = 222222_4 = 2730
\]

\[
T = k \cdot 4^m + \sum_{i=0}^{m-1} 2 \cdot 4^i
\]
Correctness

- **Claim.** $G$ has a vertex cover of size $k$ if and only if there is a subset $X$ of corresponding integers that sums to value $T$

- $(\Rightarrow)$ Let $C$ be a vertex cover of size $k$ in $G$, define $X$ as
  \[ X := \{a_v \mid v \in C\} \cup \{b_i \mid \text{edge } i \text{ has exactly one endpoint in } C\} \]

- Sum of the most significant bits of $X$ is $k$
  - Only vertex gadgets have MSB set, select $k$ vertex integers

- All other bits must sum to 2, why?

- Consider column for edge $(u, v)$:
  - Either both endpoints are in $C$, then we get two 1's from $a_u$ and $a_v$ and none from $b_{uv}$
  - Exactly one endpoint is in $C$: get 1 bit from $b_{uv}$ and 1 bit from $a_u$ or $a_v$

- Thus the elements of $X$ sum to exactly $T$
Vertex Cover to Subset Sum

• **Claim.** \( G \) has a vertex cover of size \( k \) if and only there is a subset \( X \) of corresponding integers that sums to value \( T \)

• ( \( \Leftarrow \)) Let \( X \) be the subset of numbers that sum to \( T \)

• That is, there is \( V' \subseteq V, E' \subseteq E \) s.t.

\[
X := \sum_{v \in V'} a_v + \sum_{i \in E'} b_i = T = k \cdot 4^m + \sum_{i=0}^{m-1} 2 \cdot 4^i
\]

• These numbers are base 4 and there are no carries

• Each \( b_i \) only contributes 1 to the \( i^{th} \) digit, which is 2 in our target \( T \)

• Thus, for each edge \( i \), at least one of its endpoints must be in \( V' \)
  • \( V' \) is a vertex cover

• Size of \( V' \) is \( k \): only vertex-numbers have a 1 in the \( m^{th} \) position
Subset Sum: Final Thoughts

- Polynomial time reduction?
  - $O(nm)$ since we check vertex/edge incidence for each vertex/edge when creating $n + m$ numbers

- Does a $O(nT)$ subset-sum algorithm mean vertex cover can be solved in polynomial time?
  - No! $T \approx 4^m$

- NP hard problems that have pseudo-polynomial algorithms are called weakly NP hard
Steps to Prove $X$ is NP Complete

- Step 1. Show $X$ is in NP
- Step 2. Pick a known NP hard problem $Y$ from class
- Step 3. Show that $Y \leq_p X$
  - Show both sides of reduction are correct: if and only if directions
  - State that reduction runs in polynomial time in input size of problem $Y$
Acknowledgments

- Some of the material in these slides are taken from
  - Jeff Erickson’s Algorithms Book (http://jeffe.cs.illinois.edu/teaching/algorithms/book/Algorithms-JeffE.pdf)