NP Completeness and More Reductions
List of NP Complete Problems So Far

- Circuit-SAT
- SAT
- 3-SAT
- INDEPENDENT SET
- VERTEX COVER
- SET COVER
- CLIQUE
- More to come:
  - 3-COLOR
  - Subset Sum/Knapsack
  - Hamiltonian cycle / path
Graph 3-Color Problem

- **3-COLOR.** Given an undirected graph $G = (V, E)$, is it possible to color the vertices with 3 colors s.t. no adjacent nodes have the same color.

- We argued last class that $3\text{-COLOR} \in \text{NP}$. 

![Graph with 3-coloring]

*yes instance*
3-SAT to 3-Color Problem

- **Theorem.** $3$-SAT $\leq_p 3$-COLOR

- **Proof.** Given a 3-SAT instance $\Phi$, we define $G$ by as follows
  - **Truth gadget:** a triangle with three nodes $T, F, X$ (for true, false and other) — they must get different colors (say true, false, other)
  - **Variable gadget:** triangle made up of variable $a$, its negation $\bar{a}$ and the $X$ node of the truth gadget — enforces $a, \bar{a}$ are colored true/false
3-SAT to 3-Color Problem

• Theorem. $3$-SAT $\leq_p$ $3$-COLOR

• Proof. Given a 3-SAT instance $\Phi$, we define $G$ by as follows
  
  • Truth gadget: a triangle with three nodes $T, F, \text{ and } X$ (for true, false and other) — they must get different colors (say true, false, other)
  
  • Variable gadget: triangle made up of variable $a$, its negation $\overline{a}$ and the $X$ node of the truth gadget — enforces $a, \overline{a}$ are colored true/false
  
  • Clause gadget: joins three literal nodes (from the variable gadget) to node $T$ in the truth gadget using a subgraph as shown below

$$\Phi$$

$$(a \lor b \lor \overline{c})$$
3-SAT to 3-Color Problem

- **Theorem.** 3-SAT $\leq_p$ 3-COLOR

- **Proof.**
  - Clause gadget enforces that in a valid 3-coloring, not all three literals can be colored FALSE
  - Notice that if $a, b$ get the same color (FALSE) then the right-end-point of the triangle must be colored the same (show in blue)
    - $\overline{c}$ can only be colored True in this case (why?)

\[(a \lor b \lor \overline{c})\]
Theorem. 3-SAT $\leq_p$ 3-COLOR

Proof.

- Clause gadget enforces that in a valid 3-coloring, not all three literals can be colored FALSE

- Notice that if $a, b$ get the same color (FALSE) then the right-end-point of the triangle must be colored the same (show in blue)
  - $\overline{c}$ can only be colored True in this case (why?)

\[(a \lor b \lor \overline{c})\]
3-SAT to 3-Color Problem

- Theorem. $\text{3-SAT} \leq_p \text{3-COLOR}$
- All valid 3-colorings of the “half-gadget of the clause) on the left
- Overall $G$ for example instance on the right
3-SAT to 3-Color Problem

- **Theorem.** $3$-SAT $\leq_p 3$-COLOR

- **Proof.**
  
  - $(\Rightarrow)$ If $\Phi$ is satisfiable, color the variables based on the satisfying assignment (and because each clause is satisfied) extend the coloring to the clause gadgets.

  - $(\Leftarrow)$ If $G$ is 3-colorable, then we can assign truth values based on the colors (at least one of the literals in each clause must be colored true) and thus the resulting assignment must satisfy $\Phi$.

- Note this problem extends to $k$-coloring of graphs for $k \geq 3$ and the generalized problem is also hard.
MY HOBBY:
EMBEDDING NP-COMPLEX PROBLEMS IN RESTAURANT ORDERS

WE'D LIKE EXACTLY $15.05
WORTH OF APPETIZERS, PLEASE.

...EXACTLY? UHH...

HERE, THESE PAPERS ON THE KNAPSACK
PROBLEM MIGHT HELP YOU OUT.

LISTEN, I HAVE SIX OTHER
TABLES TO GET TO—

—AS FAST AS POSSIBLE, OF COURSE. WANT
SOMETHING ON TRAVELING SALESMAN?

From: https://xkcd.com/287/
Subset Sum Problem

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

- **SUBSET-SUM $\in$ NP**
  - Certificate: a subset of numbers
  - Poly-time verifier: checks if subset is from the given set and sums exactly to $T$

- Problem has a pseudo-polynomial $O(nT)$-time dynamic programming algorithm similar to Knapsack

- We will show that it is NP hard by reducing from vertex cover
  - NP hard problems that have pseudo-polynomial algorithms are called *weakly NP hard*
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 $T$. 

![Diagram showing the relationship between VERTEX-COVER and SUBSET-SUM algorithms.](image)
Vertex Cover to Subset Sum

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

- **Proof.** Label the edges of \( G \) as \( 0, 1, \ldots, m - 1 \)

- **Reduction.** Create \( n + m \) integers and a target value \( T \) as follows

  - Each integer is a \( m + 1 \)-digit number written in base four
  - Integers representing vertices and edges:
    - \( a_v : m \)th (most significant) digit is 1 and for \( i < m \), the \( i \)th digit is 1 if \( i \)th edge is incident to vertex \( v \)
    - \( b_{uv} : m \)th digit is 0 and for \( i < m \), the \( i \)th digit is 1 if this integer represents edge \( i = (u, v) \)
  - Target value \( T = k \cdot 4^m + \sum_{i=0}^{m-1} 2 \cdot 4^i \)
Vertex Cover to Subset Sum

• **Theorem.** VERTEX-COVER \( \leq_p \) SUBSET-SUM

• Proof. Label the edges of \( G \) as \( 0, 1, \ldots, m - 1 \)

• **Reduction.** Create \( n + m \) integers and a target value \( T \) as follows

• Each integer is a \( m + 1 \)-digit number written in base four

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

\[
\begin{align*}
  a_u &= 111000_4 = 1344 \\
  a_v &= 110110_4 = 1300 \\
  a_w &= 101101_4 = 1105 \\
  a_x &= 100011_4 = 1029 \\
  b_{uv} &= 010000_4 = 256 \\
  b_{uw} &= 001000_4 = 64 \\
  b_{vw} &= 000100_4 = 16 \\
  b_{vx} &= 000010_4 = 4 \\
  b_{wx} &= 000001_4 = 1
\end{align*}
\]

• If \( k = 2 \) then \( T = 222222_4 = 2730 \)
Vertex Cover to Subset Sum

- **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 the subset \(X\) 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 digits of \(X\) is \(k\) and all other digits sum to 2.

- Thus the elements of \(X\) sum to exactly \(T\)

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

\[
\begin{align*}
a_u &:= 111000_4 = 1344 & b_{uv} &:= 010000_4 = 256 \\
a_v &:= 110110_4 = 1300 & b_{uw} &:= 001000_4 = 64 \\
a_w &:= 101101_4 = 1105 & b_{vw} &:= 000100_4 = 16 \\
a_x &:= 100011_4 = 1029 & b_{vx} &:= 000010_4 = 4 \\
                   &                   & b_{wx} &:= 000001_4 = 1
\end{align*}
\]

\[E = \{(u,v), (u,w), (v,w), (v,x), (w,x)\} \quad C = \{v, w\} \quad T = 222222_4 = 2730\]
Vertex Cover to Subset Sum

- **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 $T$

- $(\iff)$ Let $X$ be the subset of numbers that sum to $T$ then 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
- Thus, for each edge $i$, at least one of its endpoints must be in $V'$
  - $V'$ is a vertex cover
- Size of $V'$ must be $k$: only vertex-numbers have a 1 in the $m$th position
Subset Sum to Knapsack

- **Knapsack.** Given \( n \) elements \( a_1, \ldots, a_n \) where each element has a weight \( w_i \geq 0 \) and a value \( v_i \geq 0 \) and target weight \( W \) and value \( K \). Does there exist a subset \( X \) of numbers such that

\[
\sum_{a_i \in X} w_i \leq W
\]

\[
\sum_{a_i \in X} v_i \geq K
\]

- **Knapsack \( \in \) NP**
  - Can check if given subset satisfies the above conditions

- **Subset-Sum \( \leq_p \) Knapsack.** \( K = W = T \) and \( w_i = v_i = a_i \) for all \( i \)
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)

  • Hamiltonian cycle reduction images from Michael Sipser’s Theory of Computation Book