http://www.cs.unh.edu/~ruml/cs758
To prove some problem $A$ is NP-complete:

1. Prove $A \in NP$
2. Prove $A$ is NP-hard.

   (a) Pick a known NP-complete problem $B$
   (b) Design a reduction that translates instances of $B$ into equivalent instances of $A$

      i. Show that translated $A$ version is accepted if and only if the original $B$ version should be accepted.
      ii. Prove that the reduction runs in polynomial time.
Reductions to Graph Problems
CIRCUIT-SAT

SAT

3-CNF SAT

CLIQUE

SUBSET-SUM

VERTEX-COVER

HAM-CYCLE

TSP
Given graph $G$ and integer $k > 1$, does $G$ have clique of size $k$?

CLIQUE $\in$ NP: given clique, test connectivity ($k^2$ time).

CLIQUE is NP-Hard: Reduction from 3-CNF SAT! Formula $\phi$ with $k$ clauses will be SAT iff graph $G$ has a $k$ clique.

For clause $r$ like $(l^r_1 \lor l^r_2 \lor l^r_3)$, add vertices $v^r_1$, $v^r_2$, and $v^r_3$ to $G$. Add edge from $v^r_i$ to $v^s_j$ iff $r \neq s$ and $l^r_i \neq \neg l^s_j$.

SAT $\Rightarrow$ clique: If $\phi$ SAT, at least one literal in each clause is true. These form a clique in $G$ because they cannot conflict.

Clique $\Rightarrow$ SAT: If $k$ clique, make corresponding literals true. Will satisfy all $k$ clauses without conflicts.

Example: $(x_1 \lor \neg x_2 \lor \neg x_3) \land (\neg x_1 \lor x_2 \lor x_3) \land (x_1 \lor x_2 \lor x_3)$
Given graph $G$ and integer $k > 0$, does $G$ have a vertex cover of size $k$?

**VERTEX-COVER $\in$ NP:** given cover, check size and that each edge is covered.

**VERTEX-COVER is NP-Hard:** Reduction from CLIQUE. Form graph complement $G'$, which has edge $(u, v)$ for $v \neq u$ iff original does not. Claim: $G$ has $k$ clique iff $G'$ has $|V| - k$ cover.

**Cover $\Rightarrow$ clique:** All edges in $E$ have at least one endpoint in $Cover$. All pairs $(u, v)$ with both $u$ and $v \notin Cover$ therefore have edge $\in E$. So $V - Cover$ is a clique of size $k$.

**Clique $\Rightarrow$ cover:** Any edge $(u, v) \in E$ implies $\not\in E$ implies $u$ or $v$ not in Clique. This implies $u$ or $v$ remains in $V - Clique$ and hence it covers that edge. Size of $V - Clique$ is $|V| - k$. 
asst 12

Wildcard vote!
Reduction to a Numeric Problem
Reducions

NPC Proofs
Graph Problems
Number Problem

- Reductions
  - Subset Sum
  - Example Formula
  - Subset Sum
  - Resulting Set
  - EOLQs

CIRCUIT-SAT
  ↓
  SAT
  ↓
  3-CNF SAT
  ↓
  CLIQUE
  ↓
  SUBSET-SUM
  ↓
  VERTEX-COVER
  ↓
  HAM-CYCLE
  ↓
  TSP
Given finite set of positive integers, is there a subset that sums to $t$?

**SUBSET-SUM** $\in$ NP: given subset, compute sum.

**SUBSET-SUM** is NP-Hard: Reduction from 3-CNF SAT. Make numbers and the target sum from the formula. For $n$ variables and $k$ clauses, each number will have $n + k$ digits. We ensure no carrying by using base 10 and at most a sum of 6 in each column.

[ see upcoming slide for how to make numbers and target ]

Polynomial time to construct and equivalent to satisfiability.
Example Formula

\[ C_1 : (x_1 \lor \neg x_2 \lor \neg x_3) \land \]
\[ C_2 : (\neg x_1 \lor \neg x_2 \lor \neg x_3) \land \]
\[ C_3 : (\neg x_1 \lor \neg x_2 \lor x_3) \land \]
\[ C_4 : (x_1 \lor x_2 \lor x_3 \) \]
Two kinds of numbers:

- Two numbers for each variable, representing positive/negative literals. (These are the ‘important’ ones!) 1 in the variable’s column, and 1 for clauses where that literal appears.
- Clause numbers just allow slop for 1, 2 or 3 true literals per clause.

Target is 1 for each variable and 4 for each clause. Therefore, it requires exactly one form of each variable and at least one true literal in each clause (plus one or both ‘slop numbers’).

Sum $\Rightarrow$ SAT: read off assignment. Target ensures consistency and variable numbers ensure satisfiability.

SAT $\Rightarrow$ sum: construct sum, choosing slop variables last.
### Resulting Set

<table>
<thead>
<tr>
<th></th>
<th>$x_1$</th>
<th>$x_2$</th>
<th>$x_3$</th>
<th>$C_1$</th>
<th>$C_2$</th>
<th>$C_3$</th>
<th>$C_4$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$v_1$</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>$v_1'$</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>$v_2$</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>$v_2'$</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>$v_3$</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>$v_3'$</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>$s_1$</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>$s_1'$</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>$s_2$</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>$s_2'$</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>$s_3$</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>$s_3'$</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>$s_4$</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>$s_4'$</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>$t$</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
</tr>
</tbody>
</table>
### Resulting Set

<table>
<thead>
<tr>
<th></th>
<th>$x_1$</th>
<th>$x_2$</th>
<th>$x_3$</th>
<th>$C_1$</th>
<th>$C_2$</th>
<th>$C_3$</th>
<th>$C_4$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$v_1$</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>$v_1'$</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>$v_2$</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>$v_2'$</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>$v_3$</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>$v_3'$</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>$s_1$</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>$s_1'$</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>$s_2$</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>$s_2'$</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>$s_3$</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>$s_3'$</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>$s_4$</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>$s_4'$</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>$t$</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
</tr>
</tbody>
</table>
For example:

- What’s still confusing?
- What question didn’t you get to ask today?
- What would you like to hear more about?

Please write down your most pressing question about algorithms and put it in the box on your way out.

*Thanks!*