Recap from last class:

For problems A & B

\[ A \rightarrow B \]

means reduce A to B:

given a poly-time algorithm for B

we can construct a poly-time alg.

for A.

\[ \text{SAT:} \]

input: Boolean formula \( f \) in CNF with \( n \) variables & \( m \) clauses

output: Satisfying assignment, if one exists

NO otherwise

Example: \( V = \text{OR}, \ A = \text{AND} \)

\[ f = (x_1 \text{OR} x_3 \text{OR} x_5 \text{OR} x_6) \text{AND} (x_2) \text{AND} (x_1 \text{AND} x_2 \text{AND} x_3 \text{AND} x_6) \text{AND} (x_4 \text{AND} x_5 \text{AND} x_6) \text{AND} (x_3 \text{AND} x_4 \text{AND} x_6) \]

Setting: \( x_1 = T, x_2 = T, x_3 = F, x_4 = T, x_5 = T, x_6 = F \)

Satisfies \( f \)
Theorem: SAT is NP-complete

This means that:

a) $\text{SAT} \in \text{NP}$

b) for all $\text{A} \in \text{NP}$, $\text{A} \implies \text{SAT}$

So unlikely that there is a poly-time alg. for $\text{SAT}$ as that would imply a poly-time alg. for every problem in $\text{NP}$.

**3SAT:**

- **Input:** Boolean $f$ in CNF with $n$ variables, $m$ clauses where each clause has $\leq 3$ literals
- **Output:** Satisfying assignment if one exists & NO otherwise.
We'll show 3SAT is NP-complete.

Need to show:

a) 3SAT ∈ NP

b) SAT → 3SAT

Since we know for every A ∈ NP that A → SAT

then we have

A → SAT → 3SAT

so A → 3SAT.

For (a): Given an assignment, in O(1) time per clause we check that it has at least 1 satisfied literal. And so it takes O(n) time to check that f is satisfied.
To show: SAT $\implies$ 3SAT.

Take input $f$ for SAT.

$f$ may have some big clauses.

We'll create $f'$ which has $\leq 3$ literals per clause.

And $f$ is satisfiable $\iff f'$ is satisfiable.

Example:

Consider clause $C = (\overline{x_2} \lor x_3 \lor \overline{x_1} \lor \overline{x_4})$

Create a new variable $y$.

Look at:

$$C' = (\overline{x_2} \lor x_3 \lor y) \land (y \lor \overline{x_1} \lor x_4)$$

Claim: $C$ is satisfiable $\iff C'$ is satisfiable.

Proof: $(\Rightarrow)$ Take a satisfying assignment to $C$.

This assignment satisfies at least one of the clauses in $C$.

Use $y$ to satisfy the other.

$(\Leftarrow)$ Take sat. assign. to $C'$: If $y = T$, then $x_1 = F$ or $x_4 = F$

If $y = F$ then $x_2 = F$ or $x_3 = T$, either way we've got a satisfied literal in $C$. 

Suppose \( C = (x_2 \lor x_3 \lor \overline{x}_1 \lor x_4 \lor x_5) \)

Then create 2 new variables \( y_1 \) & \( y_2 \).

Let \( C' = (x_2 \lor x_3 \lor y_1) \land (y_1 \lor x_1 \lor y_2) \land (y_2 \lor x_4 \lor x_5) \)

Claim: Fix an assignment to \( x_1, x_2, \ldots, x_5 \)

Then \( C \) is satisfied \( \iff \) There is an assignment to \( y_1, y_2 \) so that \( C' \) is satisfied.

Proof:

(\( \Rightarrow \)) if \( C \) is satisfied then that satisfies one of the 3 clauses in \( C \), at least.

We can use \( y_1, y_2 \) to satisfy the other 2.

(\( \Leftarrow \)) if \( C' \) is satisfied, \( y_1 \) & \( y_2 \) satisfy at most 2 clauses, so either \((\overline{x}_2 \lor x_3)\) or \((\overline{x}_1)\) or \((x_4 \lor x_5)\) are satisfied and this satisfies \( C \) too.
In general, for
\[ C = (a_1, v_2, v_3, \ldots, v_k) \]
for literals \( a_1, \ldots, a_k \)
add \( k-3 \) new variables \( y_1, \ldots, y_{k-3} \)
and replace \( C \) by \( k-2 \) clauses:
\[
C' = (a_1 \lor \neg v_1) \land (v_1 \lor \neg v_2) \land (v_2 \lor \neg v_3) \\
\land \ldots \land (v_{k-4} \lor \neg v_{k-2}) \land (v_{k-3} \lor \neg v_{k-1} \lor \neg v_k)
\]

Claim: for any assignment to \( a_1, \ldots, a_k \)

\( C \) is satisfied \( \iff \) there is an assignment to \( y_1, \ldots, y_{k-2} \)
Satisfying \( C' \).
(⇒) Take assignment to $a_1, \ldots, a_i$ satisfying $C$.
Let $a_i$ be min $i$ where $a_i$ is satisfied.
So $a_i = T \implies$ clause $i-1$ in $C'$ is satisfied.
Set $y_i = y_{i-2} = \ldots = Y_{i-2} = T$
$\implies 1$st $i-2$ clauses in $C'$ are satisfied.
Set $y_{i-1} = \ldots = Y_{k-2} = F$
$\implies$ clauses $i, \ldots, k-2$ in $C'$ are satisfied.

(⇐) Take assignment to $a_1, \ldots, a_k, y_1, \ldots, y_{k-2}$ satisfying $C'$.
At least one of $a_i = T$, thus $C$ is satisfied.
Why?
Suppose $a_1 = a_2 = \ldots = a_k = F$
Then & since $C$ is satisfied,
for clause 1 we must have $y_1 = T$
for clause 2, we have $y_2 = T$

for clause $k-3$, we have $y_{k-3} = T$
And then clause $(\neg y_{k-3} \lor y_{k-2} \lor y_{k-1}) = F$
So we have a contradiction.

Given $f$ for SAT
Create a new formula $f'$ by the following procedure:
For each clause $C$ in $f$,
if $C$ has $\leq 3$ literals, keep it the same
if it has $>3$ literals then add $k-3$ new variables & replace $C$ by $C$ as described before.
Use $f'$ as input for 3SAT.

$f$ is satisfiable $\iff f'$ is satisfiable

Given satisfying assignment for $f'$
ignore the new variables
and the assignment for $x_1, \ldots, x_n$
is also a satisfying assignment
for $f$. 