Announcements

• Exam 3 on Friday
• Homework 5 Solutions online
Last Time

• NP Decision/Optimization Problems
  – Decision: Does object with easily checkable property exist?
  – Optimization: What object maximizes/minimizes easily computation function?

• SAT

• Hamiltonian Cycle

• It is believed that some NP problems are hard, but proving anything is very difficult.
Reduction $A \rightarrow B$

- Instance of problem $A$
- Polynomial time reduction algorithm
- Instance of problem $B$
- Hypothetical algorithm for $B$

Solution to $A$
- Shows $B$ is at least as hard as $A$
- Polynomial time interpretation algorithm

Solution to problem $A$ instance

Solution to problem $B$ instance
Today

- NP-Completeness
- 3-SAT
Circuit SAT

**Problem:** Given a circuit C with several Boolean inputs and one Boolean output, determine if there is a set of inputs that give output 1.

![Circuit Diagram]

**Important Reduction:**
Any NP decision problem $\rightarrow$ Circuit SAT
Any NP Decision Problem \( \rightarrow \) Circuit SAT

• Any NP decision problem asks if there is some X that satisfies a polynomial-time checkable property.

• In other words, for some polynomial-time computable function F, it asks if there is an X so that \( F(X) = 1 \).

• Create a circuit C that computes F. The problem is equivalent to asking if there is an input for which C outputs 1.
NP-Complete

Circuit-SAT is our first example of an **NP-Complete** problem. That is a problem in NP that is at least as hard as any other problem in NP.

• **Good news:** If we find a polynomial time algorithm for Circuit-SAT, we have a polynomial time algorithm for all NP problems!

• **Bad news:** If any problem in NP is hard, Circuit-SAT is hard.

Note: Decision problems can be NP-Complete. For optimization problems, it is called **NP-Hard**.
Other NP-Complete/Hard Problems

The following are all NP-Complete/Hard:

• Formula SAT
• Maximum Independent Set
• TSP
• Hamiltonian Cycle
• Knapsack

How do we show this? By finding reductions from other NP-Hard/Complete Problems.
3-SAT

3-SAT is a special case of formula-SAT where the formula is an AND of clauses and each clause is an OR of at most 3 variables or their negations.

**Example:**

\[(x \lor y \lor z) \land (\bar{x} \lor u) \land (w \lor \bar{z} \lor u) \land (\bar{u} \lor w \lor \bar{z}) \land (\bar{y})\]
Circuit-SAT $\rightarrow$ 3-SAT

• Start with circuit

• Create variable for each wire

• Create formula with clause for each gate and output

\[(v \iff y \lor z) \land (u \iff x \land y) \land (w \iff u \land v) \land (t \iff \overline{w}) \land (t)\]
These Aren’t 3-SAT Clauses

We have 3-variable clauses that aren’t 3-SAT clauses. Write it in terms of them.

- Write truth table
- Each 3-SAT clause sets one output to false.

\[(x \lor y \lor \bar{z}) \land (x \lor \bar{y} \lor z)\]
\[\land (\bar{x} \lor y \lor z) \land (\bar{x} \lor \bar{y} \lor z)\]

\[= (z \iff x \lor y)\]
Note

This means that 3-SAT is also NP-Complete since we have:

Any problem in NP $\rightarrow$ Circuit SAT $\rightarrow$ 3-SAT

What other problems can we show to be NP-Complete/NP-Hard this way?
Another Look at 3-SAT

**Lemma:** A 3-SAT instance is satisfiable if and only if it is possible to select one term from each clause without selecting both a variable and its negation.

**Example:**

\[(x \lor \underline{y} \lor z) \land (\underline{x} \lor y) \land (\underline{y} \lor \underline{z}) \land (\underline{x} \lor z)\]
Proof

If satisfiable:

- Satisfying assignment causes at least one term in each clause to be true.
- Select one such term from each clause.
- Cannot contradict each other.

**Example:**

\[(x \lor y \lor z) \land (\bar{x} \lor y) \land (\bar{y} \lor \bar{z}) \land (\bar{x} \lor z)\]

\[x = \text{False}, \ y = \text{True}, \ z = \text{False}\]
Proof

If there is a way to select terms:

• Set those variables to be true
  – Can do this without contradiction
• Set other variables arbitrarily
• Causes whole statement to be true

Example: \((x \lor y \lor z) \land (\overline{x} \lor \overline{y})\)

\[x = \text{True}, \ y = \text{True}, \ z = \text{False}\]
3-SAT $\rightarrow$ Maximum Independent Set

Want to encode this select one term from each clause as a graph.

- Create one vertex for each term in each clause.
- Edges between terms in same clause.
- Edges between contradictory terms.

**Example:**

$$(x \lor y \lor z) \land (\bar{x} \lor y) \land (\bar{y} \lor \bar{x})$$
Analysis

An independent set in this graph has:

• At most one vertex from each clause.
• No vertices representing contradictory terms.

It has an independent set of size \#Clauses if and only if, you can select one term form each clause without a contradiction.

Therefore, \( |\text{MIS}| = \#\text{Clauses} \) if and only if the 3-SAT has a solution.
Example

\[(x \lor y \lor z) \land (\overline{x} \lor y) \land (\overline{y} \lor \overline{x})\]

\[x = \text{False}, \ z = \text{True}, \ y = \text{Whatever}\]
Intermediate Problems

To prove our more complicated reductions, it will help to have the correct problem to prove reductions from.

A convenient problem is the one the book calls Zero-One Equations.
Zero-One Equations

**Problem:** Given a matrix $A$ with only 0 and 1 as entries and $b$ a vector of 1s, determine whether or not there is an $x$ with 0 and 1 entries so that

$$Ax = b.$$ 

This problem is clearly in NP. We will show that it is NP-Complete.
Example

\[
\begin{bmatrix}
1 & 0 & 1 \\
1 & 1 & 0
\end{bmatrix}
\begin{bmatrix}
x_1 \\
x_2 \\
x_3
\end{bmatrix}
= 
\begin{bmatrix}
1 \\
1
\end{bmatrix}
\]

Equivalently, do there exist \(x_1, x_2, x_3 \in \{0,1\}\) so that

\[x_1 + x_3 = 1\]
\[x_1 + x_2 = 1\]

Generally, this is what a ZoE looks like. A bunch of sets of \(x_i\)s that need to add to 1.