COMP 355
Advanced Algorithms
Cook’s Theorem, 3SAT, and Independent Set
Chapter 8 (KT)
Section 34.4-34.5 (CLRS)
Recap

**P:** is the set of decisions problems solvable in polynomial time, or equivalently, the set of languages for which membership can be determined in polynomial time.

**NP:** is the set of languages that can be verified in polynomial time, or equivalently, that can be solved in polynomial time by a “guessing computer”, whose guesses are guaranteed to produce an output of “yes” if at all possible.

**Polynomial reduction:** \( L_1 \leq_p L_2 \) means that there is a polynomial time computable function \( f \) such that \( x \in L_1 \) if and only if \( f(x) \in L_2 \). A more intuitive way to think about this is that if we had a subroutine to solve \( L_2 \) in polynomial time, then we could use it to solve \( L_1 \) in polynomial time. Polynomial reductions are transitive, that is, \( L_1 \leq_p L_2 \) and \( L_2 \leq_p L_3 \) implies \( L_1 \leq_p L_3 \).

**NP-Hard:** \( L \) is NP-hard if for all \( L' \in NP \), \( L' \leq_p L \). By transitivity of \( \leq_p \), we can say that \( L \) is NP-hard if \( L' \leq_p L \) for some known NP-hard problem \( L' \).

**NP-Complete:** \( L \) is NP-complete if (1) \( L \in NP \) and (2) \( L \) is NP-hard.

It follows from these definitions that:
- If any NP-hard problems is solvable in polynomial time, then every NP-complete problem (in fact, every problem in NP) is also solvable in polynomial time.
- If any NP-complete problem cannot be solved in polynomial time, then every NP-complete problem (in fact, every NP-hard problem) cannot be solved in polynomial time.
Thus all NP-complete problems are equivalent to one another (in that they are either all solvable in polynomial time, or none are).
Satisfiability

**Literal:** A Boolean variable or its negation.

\[ x_i \text{ or } \overline{x_i} \]

**Clause:** A disjunction of literals.

\[ C_j = x_1 \lor \overline{x_2} \lor x_3 \]

**Conjunctive normal form:** A propositional formula \( \Phi \) that is the conjunction of clauses.

\[ \Phi = C_1 \land C_2 \land C_3 \land C_4 \]

**SAT:** Given CNF formula \( \Phi \), does it have a satisfying truth assignment?

**3-SAT:** SAT where each clause contains exactly 3 literals.

\[ \uparrow \]

\[ \text{each corresponds to a different variable} \]

**Ex:**

\[ \left( \overline{x_1} \lor x_2 \lor x_3 \right) \land \left( x_1 \lor \overline{x_2} \lor x_3 \right) \land \left( x_2 \lor x_3 \right) \land \left( \overline{x_1} \lor \overline{x_2} \lor \overline{x_3} \right) \]

**Yes:** \( x_1 = \text{true}, x_2 = \text{true}, x_3 = \text{false} \).
Cook’s Theorem: SAT is NP-complete.

SAT is in NP: We nondeterministically guess truth values to the variables. We can then plug the values into the formula and evaluate it. Clearly, this can be done in polynomial time.

SAT is NP-Hard:
1. Every NP-problem can be encoded as a program that runs in polynomial time on a given input, subject to a number of nondeterministic guesses.
2. We can express its execution on a specific input as straight-line program that contains a polynomial number of lines of code.
3. Compile each line of code into machine code, and convert each machine code instruction into an equivalent boolean circuit
4. Express each of these circuits equivalently as a boolean formula.
INDEPENDENT SET: Given a graph $G = (V, E)$ and an integer $k$, is there a subset of vertices $S \subseteq V$ such that $|S| \geq k$, and for each edge at most one of its endpoints is in $S$?

Ex. Is there an independent set of size $\geq 6$? Yes.
Ex. Is there an independent set of size $\geq 7$? No.
Claim: IS is NP-complete.

(a) Reduction of 3-SAT to IS and (b) Clause clusters for the clauses \((x_1 \lor x_2 \lor x_3)\) and \((\overline{x_1} \lor x_2 \lor x_3)\).
3 Satisfiability Reduces to Independent Set

Claim. $3$-SAT $\leq_p$ INDEPENDENT-SET.

Pf. Given an instance $\Phi$ of $3$-SAT, we construct an instance $(G, k)$ of INDEPENDENT-SET that has an independent set of size $k$ iff $\Phi$ is satisfiable.

Construction.

- $G$ contains 3 vertices for each clause, one for each literal.
- Connect 3 literals in a clause in a triangle.
- Connect literal to each of its negations.

$$
\Phi = \left( \overline{x_1} \lor x_2 \lor x_3 \right) \land \left( x_1 \lor \overline{x_2} \lor x_3 \right) \land \left( \overline{x_1} \lor x_2 \lor x_4 \right)
$$
3 Satisfiability Reduces to Independent Set

Claim. $G$ contains independent set of size $k = \vert \Phi \vert$ iff $\Phi$ is satisfiable.

Pf. $\Rightarrow$ Let $S$ be independent set of size $k$.

$ S$ must contain exactly one vertex in each triangle.

Set these literals to true.

Truth assignment is consistent and all clauses are satisfied.

Pf $\Leftarrow$ Given satisfying assignment, select one true literal from each triangle.

This is an independent set of size $k$. □

$$\Phi = \left( x_1 \lor x_2 \lor x_3 \right) \land \left( x_1 \lor \overline{x_2} \lor x_3 \right) \land \left( \overline{x_1} \lor x_2 \lor x_4 \right)$$
3SAT to IS reduction

\[ \Phi = (x_1 \lor \overline{x_2} \lor x_3) \land (\overline{x_1} \lor x_2 \lor x_3) \land (\overline{x_1} \lor x_2 \lor \overline{x_3}) \land (x_1 \lor x_2 \lor \overline{x_3}) \]

Reduce this 3SAT to IS

What does k need to be?
What does the graph look like?
A few things about this reduction

- Every NP-complete problem has three similar elements:
  a) something is being selected
  b) something is forcing us to select a sufficient number of such things (requirements)
  c) something is limiting our ability to select these things (restrictions).
  A reduction’s job is to determine how to map these similar elements to each other.

- Our reduction did not attempt to solve the 3SAT problem.

- Remember this rule! If your reduction treats some entities different than others, based on what you think the final answer may be, you are very likely making a mistake.

- Remember, these problems are NP-complete!
By Cook’s Theorem, we know that every problem in NP is reducible to 3SAT.

When we showed that IS ∈ NP, it followed immediately that IS ≤ₚ 3SAT.

When we showed that 3SAT ≤ₚ IS, we established their equivalence.

By transitivity, it follows that all problems in NP are now reducible to IS.
(CLRS 34.5-8) In the **half 3-SAT** problem, we are given a 3-SAT formula $f$ with $n$ variables and $m$ clauses, where $m$ is even. We wish to determine whether there exists a truth assignment to the variables of $f$ such that exactly half the clauses evaluate to False (0) and exactly half the clauses evaluate to True (1).

Prove that the half 3-SAT problem is NP-complete. (You may assume that the 3-SAT formula has at most 3 literals per clause, not necessarily exactly 3.)
Next Time

Clique, Vertex Cover, and Dominating Set