Homework 7

Student: YOUR NAME HERE

Purpose:

- Design and create your own reductions.
- Learn to use the solutions to other problems to solve a new problem.
- Become familiar with classic NP-Complete problems.

General Homework Policies:

- This is an optional assignment. It will NOT be turned in or graded. However, I encourage you to complete it for additional practice.

NOTE: For this homework, unless a specific problem indicates otherwise, you may only assume the following problems are NP-Complete for your reductions. Do not assume that any other problems are NP-complete! For many of these problems, you have to find some structure, if it exists. If it doesn't exists, you can assume the algorithm returns "FALSE", "UNSATISFIABLE", or anything else convenient.

- SAT: Given a boolean formula F in CNF form with n variables and m clauses, find an assignment of the n variables that satisfies F, if one exists.
- $3-$SAT : A special case of SATwhere every clause has at most 3 literals.
- Vertex Cover : Given graph G and number k, find a vertex cover of size $\leq k$, if one exists. For a graph $G = (V, E)$, $S \subset V$ is a vertex cover if it covers every edge: for every edge $(x, y) \in E$ either $x \in S$ or $y \in S$ (or both). (So one or both endpoints of every edge are in the vertex cover.)
- Independent Set : Given graph G and number k, find an independent set of size $\geq k$, if one exists. For a graph $G = (V, E)$, $S \subset V$ is an independent set if no edges are contain in S - for all $x, y \in S$, $(x, y) \notin E$.
- Clique : Given graph G and number k, find a clique of size $\geq k$, if one exists. For a graph $G = (V, E)$, $S \subset V$ is a clique if for all $x, y \in S$, $(x, y) \in E$. (S is a fully connected subgraph - all pairs of vertices in S have an edge between them).
- Hamiltonian Cycle (Rudrata): Given graph G, find a cycle that visits every vertex exactly once, if one exists.

Remember that to show a problem is NP-Complete, you need to do three things.

1. Show that the problem is in NP.
2. Reduce this problem from a previously established NP-complete problem. (Describe an algorithm for the known NP-complete problem that can use a solution for YOUR problem as a subroutine.)
3. Prove that the reduction works. (Show that you return a valid solution IF AND ONLY IF the input to the original NP-complete problem has a valid solution.)

Homework 7

1. Consider the following problem: given a set of clauses (each a disjunction of literals) and an integer k, find a satisfying assignment in which *at most* k variables are TRUE, if such an assignment exists. Prove that this problem is NP-complete.

2. A *tadpole* is a graph on an even number of vertices, say $2k$, in which k of the vertices form a clique and the remaining k vertices are connected in a "tail" that consists of a path joined to one of the vertices of the clique. Given a graph G and integer k, the TadpoleProblem, asks for a subgraph which is a $2k$ sized tadpole (labeled so that we know which part is the clique and which is the tail). Prove that the TadpoleProblem is NP-complete.

3. In the Exact4SAT problem, the input is a set of clauses, each of which is a disjunction of exactly 4 literals, and such that each variable occurs at most once in each clause. The goal is to find a satisfying assignment, if one exists. Instead of proving that Exact4SAT is NP-complete you will implement a piece of the proof. Specifically, you will write a method that solves a known NP-complete problem (3SAT) using a solution to Exact4SAT. You have been provided a solution to Exact4SAT in the file `ExactFourSAT.java`. You will write a solution to the 3SAT problem by adding code to the `isSatisfiable` method in the `ThreeSAT.java` file. Your solution should run in *polynomial time* if the Exact4SAT solution ran in polynomial time. Of course, the Exact4SAT solution given does not in fact run in polynomial time.

   Note that you are welcome to add additional private methods or data fields but you may not modify the method signature of the `isSatisfiable` method. You also may not modify the `ExactFourSAT.java` file. Your solution should not print anything.

4. **(PRACTICE)**Consider the Clique problem restricted to graphs in which every vertex has degree at most 3. Call this problem Clique-3.
   (a) Prove that Clique-3 is in NP.
   (b) What is wrong with the following proof of NP-completeness for Clique-3?
   We just showed that Clique-3 is in NP.
   We know that the Clique problem in general graphs is NP-complete, so it is enough to present a reduction from CLIQUE-3 to CLIQUE. In fact, we use the fact that CLIQUE is a generalization of CLIQUE-3. Given an input graph G with vertices of degree ≤ 3 and parameter k, run an algorithm for CLIQUE unchanged on the same graph and same input parameter. If a clique of size $\geq k$ is present, we return it, otherwise return FALSE.
   In CLIQUE-3, we are looking for a clique of size $\geq k$ in G. This is exactly what CLIIQUE will return, if one exists. This proves the correctness of the reduction and, therefore the NP-completeness of Clique-3.
   (c) It is true that the Vertex Cover remains NP-complete even when restricted to graphs in which every vertex has degree at most 3. Call this problem VC-3. What is wrong with the following proof of NP-completeness for Clique-3?
   We present a reduction from VC-3 to Clique-3. Clique-3 is in NP as proved in part a.
   Given an input graph $G = (V, E)$ with max-degree 3 to VC-3, and a parameter b, we create an instance of Clique-3 by leaving the graph unchanged and switching the parameter to $k = |V| - b$.
   Now, a subset $C \subseteq V$ is a vertex cover in G if and only if the complementary set $V - C$ is a clique in G. Therefore G has a vertex cover of size $\leq b$ if and only if it has a clique of size $\geq |V| - b$. This proves the correctness of the reduction and, consequently, the NP-completeness of Clique-3.

(d) Describe a poly-time algorithm for Clique-3.

5. **(PRACTICE)** Integer Inequality Satisfaction: In this problem, you are given a set of n variables $x_1, x_2, \ldots x_n$, and a set of m inequalities, where the jth inequality looks like $\sum_i a_{i,j} x_i \geq c_j$. The constants $a_{i,j}$ are also integers. Our goal is to find *an integer* set of x_i such that every equation is satisfied. Prove that this problem is NP-complete via a direct reduction from either SATor $3-$SAT . If your clause has n variables and m clauses, how many variables and inequalities does your reduction require?

   Hint: Add inequalities to force your variables to behave like "True" or "False". What do you need after that?

6. **(PRACTICE)** Consider the following *longest* $(s-t)$-path problem. Given a graph G an integer k and two vertices s and t the goal is to find a simple path using k or more edges from s to t, if one exists. Prove that, unlike the problem of finding the shortest path in a graph, the *longest* $(s-t)$-path problem is NP-complete by directly reducing from Hamiltonian (Rudrata) Cycle.