Below is a set of practice problems on proving problems NP-complete, to help you check your understanding of the approach we’ve discussed in class.

You can find solutions to these problems on the course web site. We’re also willing to listen to you describe your solutions or to look at your writeups during office hours.

These exercises refer to several well-known NP-complete problems, most of which we will talk about in class: SAT, 3-CNF-SAT, SUBSET-SUM, PARTITION, HAM-CYCLE, TSP, VERTEX-COVER, and CLIQUE. If you want to make some progress before we get to these problems in class, they are all defined in your book and in numerous places online.

For homework 3, concentrate on part 1. The remaining problems in part 2 will also be useful for Homework 4, and for studying for the second exam.

Note: If you want to discuss these problems, I expect you to give correctness and complexity proofs for your algorithms here, just as on the actual homework. Efficiency matters!

### Practice Problems, Part 1

1. Consider the problem COMPOSITE: given an integer $y$, does $y$ have any factors other than 1 and itself? For this exercise, you may assume that COMPOSITE is in NP, and you will be comparing it to the well-known NP-complete problem SUBSET-SUM: given a set $S$ of $n$ integers and an integer target $t$, is there a subset of $S$ whose sum is exactly $t$?

Clearly explain whether or not each of the following statements follows from the NP-ness of COMPOSITE and the NP-completeness of SUBSET-SUM:

(a) $\text{SUBSET-SUM} \leq_p \text{COMPOSITE}$.

(b) If there is an $O(n\sqrt{t})$ algorithm for SUBSET-SUM, then P = NP.

(c) If there is an $O(n^3 \log t)$ algorithm for SUBSET-SUM, then there is a polynomial time algorithm for COMPOSITE.

(d) If there is an $O(\log y)$ algorithm for COMPOSITE, then P = NP.

(e) If P $\neq$ NP, then no problem in NP can be solved in polynomial time.

2. Two well-known NP-complete problems are 3-CNF-SAT and TSP, the traveling salesman problem. The 2-CNF-SAT problem is a SAT variant in which each clause contains at most 2 literals; it is known to have a polynomial-time algorithm. Is each of the following statements true or false? Justify your answer.

(a) $\text{3-CNF-SAT} \leq_p \text{TSP}$.

(b) If P $\neq$ NP, then 3-CNF-SAT $\leq_p$ 2-CNF-SAT.

(c) If P $\neq$ NP, then no NP-complete problem can be solved in polynomial time.
3. Prove that the following problem, the Non-Bored Jogger Problem (NBJ), is NP-complete.

You are given as input a weighted, undirected multigraph \( G \); a distinguished home node \( v \) in \( G \), and an integer \( \ell \geq 0 \). Each edge in \( G \) has a positive integer weight. Self-loops are permitted in \( G \), as are multiple edges with the same endpoints. Does there exist a path in \( G \) that starts and ends at \( v \), traverses a set of edges with total weight \( \ell \), and traverses each edge at most once? (\textit{Hint}: Reduce from SUBSET-SUM.)

4. The subgraph isomorphism problem takes as input two undirected graphs \( G_1 \) and \( G_2 \) and asks whether \( G_1 \) is a subgraph of \( G_2 \). In other words, the problem asks whether there is a one-to-one function \( f \) mapping the vertices of \( G_1 \) to the vertices of \( G_2 \), such that edge \((u, v)\) exists in \( G_1 \) iff \((f(u), f(v))\) exists in \( G_2 \).

Show that the subgraph isomorphism problem is NP-complete, using a reduction from one of: SAT, 3-CNF-SAT, CLIQUE, or VERTEX-COVER.

Practice Problems, Part 2

1. The set intersection problem (SIP) is defined as follows: Given finite sets \( A_1, A_2, \ldots, A_r \) and \( B_1, B_2, \ldots, B_s \), is there a set \( T \) such that

\[ |T \cap A_i| \geq 1 \text{ for } i = 1, 2, \ldots, r \]

and

\[ |T \cap B_j| \leq 1 \text{ for } j = 1, 2, \ldots, s? \]

Prove that the set intersection problem is NP-complete. (\textit{Hint}: Reduce from 3-CNF-SAT).

2. A feedback vertex set in a directed graph \( G = (V, E) \) is a subset \( V' \) of \( V \), such that \( V' \) contains at least one vertex from each directed cycle in \( G \). The feedback vertex set problem (FVS) is: given a directed graph \( G \) and an integer \( k \), does \( G \) have a feedback vertex set with at most \( k \) vertices?

Prove that VERTEX-COVER \( \leq_p \) FVS. Does this fact alone imply that FVS is NP-complete? Why or why not?

3. In the BOUNDED-DEGREE-SPANNING-TREE (BDST) problem, you are given as input an undirected graph \( G = (V, E) \) and a positive integer \( k \leq |V| - 1 \). The question is whether or not there is a spanning tree of \( G \) in which no vertex has degree more than \( k \) (i.e. each vertex has at most \( k \) edges of the spanning tree incident to it).

Show that BDST is NP-complete.

4. Consider the following problem, called FIRE-STATION-PLACEMENT. You are given a simple, undirected weighted graph \( G = (V, E) \) and an integer \( f \leq |V| \). Your goal is to choose \( f \) vertices of \( G \) as fire stations so as to minimize the distance from any vertex of \( G \) to a fire station. That is, if \( d(v, u) \) is the shortest-path distance in \( G \) between vertices \( v \) and \( u \), the goal is to choose a subset \( F \subseteq V \) of \( f \) firestations so as to minimize

\[ \max_{v \in V} \min_{u \in F} d(v, u). \]

Show that this is an NP optimization problem.