1. Search vs. decision problems.

As discussed in class, NP is a class of decision problems, i.e., the answer is either “yes” or “no.” This choice is convenient and often also captures the difficulty of searching for a solution, as you will show in this problem for two examples.

(a) 3-SAT. A 3-SAT formula consists of $m$ clauses, each of which is a disjunction of three terms (where a term is one of $n$ variables or its negation). We say a 3-SAT formula $\varphi$ is satisfiable if there is an assignment of the $n$ variables such that $\varphi$ evaluates to true. The 3-SAT problem asks us to decide whether a given $\varphi$ is satisfiable.

Suppose you are given a black box for a function 3SAT that determines whether a given 3-SAT formula is satisfiable. Describe an algorithm that finds a satisfying assignment for $\varphi$ (assuming it is satisfiable) using polynomially many calls to 3SAT and polynomially many other steps. Prove that your algorithm is correct and analyze its running time. For full credit, your algorithm should use $O(n)$ calls to the black box.

(b) 3-Coloring. We say an undirected graph is 3-colorable if there is an assignment of the colors $\{r, g, b\}$ to the vertices (a coloring) such that no two adjacent vertices have the same color. The 3-Coloring problem asks us to decide whether a given graph is 3-colorable.

Suppose you are given a black box for a function 3Color that determines whether a given graph is 3-colorable. Describe an algorithm that finds a coloring of a given graph using polynomially many calls to 3Color and polynomially many other steps. Prove that your algorithm is correct and analyze its running time. For full credit, your algorithm should use $O(n)$ calls to the black box, where $n$ is the number of vertices in the input graph.

2. Nonoverlapping Paths is NP-complete.

Suppose you are given an undirected graph $G$ and a set $P = \{P_1, P_2, \ldots, P_k\}$, where each $P_i$ is a simple path in $G$. We say a set of paths $Q = \{P_{i_1}, P_{i_2}, \ldots, P_{i_t}\} \subseteq P$ is nonoverlapping if no two paths in the set share a vertex. In the Nonoverlapping Paths problem, you are given a graph $G$, a set of paths $P$ in $G$, and a positive integer $t$, and you are asked whether there is a nonoverlapping set of paths $Q \subseteq P$ with $|Q| \geq t$.

(a) Show that Nonoverlapping Paths is in NP.

(b) Show that Nonoverlapping Paths is NP-hard.

3. Restricted Monotone Satisfiability is NP-complete.

We say an instance of Satisfiability is monotone if every term is a non-negated variable (i.e., $x_i$ can appear as a term but $\bar{x}_i$ cannot). Monotone instances are always satisfiable since we can simply set every variable to true, but the problem becomes more difficult if we restrict the number of variables that can be true. In the Restricted Monotone Satisfiability problem, we are given a positive integer $k$ and a monotone satisfiability instance (a set of clauses, each of which is a disjunction of non-negated variables), and we are asked whether there is an assignment of the variables with at most $k$ variables set equal to true such that all clauses evaluate to true.

(a) Show that Restricted Monotone Satisfiability is in NP.

(b) Show that Restricted Monotone Satisfiability is NP-hard.

A *cut* in an undirected graph is a bipartition of the vertices. The *size* of a cut is the number of edges crossing the cut (i.e., the number of edges whose endpoints are on opposite sides of the cut). We say a cut is *maximum* if its size is as large as possible. The problem of finding a maximum cut is NP-hard. (You are not asked to prove this.)

Design a polynomial-time algorithm that finds a cut whose size is at least half as large as the maximum cut. Prove that your algorithm is correct and analyze its running time.