CS41 Lab 9: Polynomial-Time Verifiers

In typical labs this semester, you'll be working on a number of problems in groups of 3-4 students. You will not be handing in solutions; the primary purpose of these labs is to have a low-pressure space to discuss algorithm design. However, it will be common to have some overlap between lab exercises and homework sets.

This week, we've started to understand what makes some problems seemingly hard to compute. In this lab, we'll consider an easier problem of *verifying* that an algorithm's answer is correct. Recall that a *decision problem* is a problem that requires a YES or NO answer. Alternatively, we can describe decision problem as a set $L \subseteq \{0,1\}^*$; think of L as the set of all YES inputs i.e., the set of inputs x such that one should output YES on input x. Let $|x|$ denote the length of x, in bits.

Polynomial-time Verifiers. Call V an efficient *verifier* for a decision problem L if

1. V is a polynomial-time algorithm that takes two inputs: x, and w.
2. There is a polynomial function p such that for all strings x, $x \in L$ if and only if there exists w such that $|w| \leqslant p(|x|)$ and $V(x, w) =$ YES.

w is usually called the *witness* or *certificate*. Think of w as some *proof* that $x \in L$. For V to be a polynomial-time verifier, w must have size some polynomial of the input x. For example, if x represents a graph with n vertices and m edges, the length of w could be n^2 or m^3 or $(n+m)^{100}$ but not 2^n.

The following problems are not known to have polynomial-time algorithms. For each problem describe a polynomial-time verifier (and the corresponding certificates).

1. THREE-COLORING. Given $G = (V, E)$ return YES iff the vertices in G can be colored using at most three colors such that for every edge $e = (u, v)$ in E, u and v have different colors.
2. SAT. Given a List of n boolean variables $x_1, \ldots, x_n$ and m clauses $c_1, \ldots, c_m$, return YES iff there is a *satisfying assignment* i.e., if there exists a truth assignment ot variables $x_1, \ldots, x_n$ such that every clause c_j evaluates to TRUE.
3. INDEPENDENT-SET Given an undirected graph $G = (V, E)$ and integer k and returns YES iff G contains an independent set of size at least k. An independent set is a set of vertices with no edges between them: $W \subseteq V$, and $u, v \in W \Rightarrow (u, v) \notin E$.
4. VERTEX-COVER takes an undirected graph $G = (V, E)$ and integer k and returns YES iff G contains a vertex cover of size at most k. A vertex cover is a set of vertices such that every edge has at least one end in the set: $W \subseteq V$, and $\forall\ (u, v) \in E$, $u \in W$ or $v \in W$.
5. FACTORING. Given numbers n, k written in binary, output YES iff n is divisible by d for some $1 < d \leqslant k$.
6. NOT-FACTORING. Given numbers n, k written in binary, output YES iff n is `NOT` divisible by d for any $1 < d \leqslant k$.

   Hint: The following problem **is solvable** in polynomial time:[1]

   PRIMES: Given a number n written in binary, output YES iff n is a prime number.

[1] This actually wasn't known until 2002, when Agrawal, Kayal, and Saxena created the AKS primality test. Kayal and Saxena were undergraduates at IIT Kanpur at the time; Agrawal was their advisor.