CS41 Lab 8: Polynomial-Time Verifiers

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.

1. **Reducing Vertex-Cover to Independent-Set**. Recall from class the following problems

   - INDEPENDENT-SET takes an undirected graph $G = (V, E)$ and integer k and returns YES iff G contains an independent set of size at least k.
   - 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.

   In class we saw that INDEPENDENT-SET $\leq_{\mathrm{P}}$ VERTEX-COVER. Show that

   $$\text{VERTEX-COVER} \leq_{\mathrm{P}} \text{INDEPENDENT-SET} .$$

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

   (a) V is a polynomial-time algorithm that takes two inputs x and w.

   (b) There is a polynomial function p such that for all strings x, $x \in L$ if and only if there exists w such that $|w| \leq p(|x|)$ and $V(x, w) =$ YES.

   Give polynomial-time verifiers for the following problems, none of which are known to have polynomial-time algorithms.

   (a) INDEPENDENT-SET.

   (b) VERTEX-COVER.

   (c) 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.

   (d) WEDDING-PLANNER. Recall in the Wedding planner problem, the input consists of a list of n people to possibly invite to a wedding, along with m *clauses*, where each clause specifies some criteria for whom to invite or not invite. Output YES iff there exists an invitation list that satisfies all clauses.

   Note: Assume that each clause is of the form e.g. $x_1 \vee \bar{x}_2 \vee \cdots \vee \bar{x}_k$, where x_i means to invite person i, and $\bar{x}_j$ means to not invite person x_j.

   (e) FACTORING. Given numbers n, d written in binary, output YES iff n is divisible by k for some $1 < k \leq d$.

   (f) NOT-FACTORING. Given numbers n, d written in binary, output YES iff n is **NOT** divisible by k for any $1 < k \leq d$.

   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.