# CS170 Fall 2013 Solutions to Homework 11

Zackery Field, section Di, 103, `cs170-fe`

November 29, 2013

## 1 [10 pts.] Mindbending Logic

Show that if Horn SAT is not NP-complete, then P≠NP. (Hint: Consider the contrapositive.)

Following the hint, let the contrapositive be; If P=NP, then the Horn SAT is NP-complete. It is proven in the text that Horn SAT is in P. Since Horn SAT is in P, if P=NP, then Horn SAT is also NP-complete.

## 2 [20 pts.] Proving NP-completeness by Generalization

For each of the problems below, prove that it is NP-complete by showing that it is a *generalization* of an NP-complete problem.

(a) Subgraph Isopmorphism: Given as input two undirected graphs $G$ and $H$, determine whether $G$ is a subgraph of $H$ (that is, whether by deleting certain vertices and edges of $H$, we obtain a graph that is, up to renaming vertices, identical to $G$), and if so, return the corresponding mapping of $V(G)$ into $V(H)$.

This is a generalization of the clique problem. The clique problem seeks a clique of size $k$. To have the subgraph isomorphism output the clique$(G, k)$ of some we can construct a graph $T$ of size $k$ that is complete. Then $T$ will be a subgraph of $G$ if there is a clique of size $k$ in $G$. Since subgraph isomorphism is a generalization of the clique problem, subgraph isomorphism is NP-complete.

(b) Longest Path: Given a graph $G$ and an integer $g$, find in $G$ a simple path of length $g$.

This is a generalization of the Rudrata Path problem. Take some graph $G = (V, E)$. The Rudrata Path problem takes this graph $G$ and finds some simple path of length $g = |V|$, xor that none exists. (If $G$ is weighted, then you can easily make it unweighted.) So for any Rudrata path problem Rudrata$(G)$, we can form a Longest Path problem, Longest$(G, g = |V|)$. Since the Rudrata path problem is NP-complete, the longest path problem is NP-complete.

(c) Max SAT: Given a CNF formula and an integer $g$, find a truth assignment that satisfies at least $g$ clauses.

This problem is a generalization of the SAT problem. Take some instance of a SAT problem $I = (CNF)$. Since SAT is attempting to discover any satisfying set or return that none exists, we can set $g = 0$ and run Max_SAT$(I = (CNF), g = 0)$. Since SAT is NP-complete, Max SAT is also NP-complete.

(d) Sparse Subgraph: Given a graph and two integers $a$ and $b$, find a set of $a$ vertices of $G$ such that there are at most $b$ edges between them.

This is a generalization of the vertex cover problem. Let $V = (G, a)$ be a vertex cover problem instance, where $G = (V, E)$ is some graph, and $a$ is the maximum number of vertices in your cover. This vertex cover problem is equivalent to a Sparse Subgraph problem where $b = |E|$: Sparse_SubGraph$(V = (G, a), b = |E|)$. Since vertex cover is NP-complete, the sparse subgraph problem is also NP-complete.

## 3 [20 pts.] Subset Product

Similar to Subset Sum, in the Subset Product problem you are given a set of integers $A = \{a_1, a_2, \ldots, a_n\}$ and a goal $g$, and your job is to determine whether there is a subset $S \subseteq A$ such that the product of the numbers in $S$ is exactly $g$.

(a) Prove that Subset Product is in NP.

Given some solution subset $S$ of a Subset_Problem$(A, g)$. Find the product of the subset of integers in $S$ and determine if $prod(S) = g$. Since the multiplication of a finite set of terms is polynomial, verification of a solution to a Subset Problem is polynomial in $S$, and the Subset Product is in NP.

(b) Point out the flaw in the given proof of NP-completeness.

In the proof, they create a set of numbers $A' = \{2^{a_1}, 2^{a_2}, \cdots, 2^{a_n}\}$. The process of creating this set is exponential in each term of $A = a_1, a_2, \ldots, a_n$. This is an invalidation of the proof because a reduction argument requires that the function $f(I)$ (for some problem instance I) be polynomial in $I$.

(c) One of the first problems to be proved NP-complete is *Exact Cover*, which is like Set Cover, except that the chosen subsets have to be *disjoint*. Prove that Subset Product is NP-complete by a reduction from Exact Cover. Make sure you prove that you do not make the same mistake as the proof in (b). (Hint: You may assume without proof that the first $n$ prime numbers can be calculated in time polynomial in $n$ , and the $n$th prime number is $O(n \log n)$.)

We are given an instance $I$ of a set cover problem with subsets $\{S_1, S_2, \ldots, S_n\}$. We can assign each of the elements in these sets a prime number. Such that two elements that are equal (but in different sets) are given the same prime number. The smallest number in the sets is assigned 2, the next element 3, all the way up to the number of distinct inputs, $n$. The goal in the reduced subset product problem is the product of the first $n$ prime numbers. If a subset product is found to equal this goal value, then there exists a disjoint collection of subsets that covers the initial exact cover instance $I$.