CS170 — Fall 2017— Homework 9 Solutions

Jonathan Sun, SID 25020651

0. Who Did You Work With?

Collaborators: Kevin Vo, Aleem Zaki, Jeremy Ou

1. Basic Complexity Concepts

(a) Suppose we reduce a problem A to another problem B. This means that if we have an algorithm that solves $\underline{\mathbf{B}}$, we immediately have an algorithm that solves $\underline{\mathbf{A}}$.

(b) Again, suppose we show a (polynomial-time) reduction from A to B. This implies that up to polynomial factors, $\underline{\mathbf{B}}$ cannot be any easier than $\underline{\mathbf{A}}$.

(c) The class **NP** is the set of all problems that can be verified in polynomial time (the class of all search problems).

(d) The class **NP**-hard are problems to which all other problems in **NP** reduce because if we could efficiently solve any **NP**-hard problem, we could efficiently solve any problem in **NP**.

(e) The class **NP**-complete are problems which are both **NP**-hard and in **NP**.

(f) For any problem $\prod$ in **NP**, there is an algorithm which solves $\prod$ in time $O(2^{p(n)})$. Becuase **NP** is the class of all search problems, then every problem can check a proposed solution and input instance in polynomial time ($p(n)$) given an algorithm c. So, when we use c on every instance, if there is a solution then c finds it in about $p(n) * 2^n$ time which is $O(2^{p(n)})$.

2. Proving NP-completeness by generalization.

(a) This Subgraph Isomorphism problem is a generalization of the Clique problem since the Clique problem tries to find a clique of size k. So, if we construct a graph of size k that is complete, which I will refer to as H, the Clique problem can be solved by the Subgraph Isomorphism, which takes in graph G and H, and will output something equivalent to the $Clique(G, k)$.

(b) This Longest Path problem is a generalization of the Rudrata problem since the Rudrata problem tries to find a simple path connecting every vertex. So, if we construct a graph $G = (V, E)$, you can solve the Rudrata problem using the Longest Path problem with $g = |V - 1|$.

(c) This Max SAT problem is a generalization of the SAT problem since the SAT problem attempts to discover a set that can satisfies all clauses. So, if we set $g =$ count of all clauses, then we can solve the SAT problem using this Max SAT problem.

(d) This Dense Subgraph problem is a generalization of the Clique problem since the Clique problem tries to find a clique of size k. Since a Clique has all vertices connected to each other with an edge, there will be $\sum_{i=0}^{k-1} i = \frac{k(k-1)}{2}$ edges given there are k vertices. Therefore, the Clique problem can be solved by the Dense Subgraph, which takes in graph G and $\frac{k(k-1)}{2}$.

(e) This Sparse Subgraph problem is a generalization of the Independent Set problem since the Independent Set is a set of vertices in a graph where no two of which are adjacent. So, if we set $b = 0$ so that we are finding a set of vertices of G such that there are no edges between them, then we get the Independent Set problem.

3. Reduction, Reduction, and Reduction

1. This problem can be reduced from the Subset Sum problem since the Subset Sum problem is to see if there a non-empty subset whose sum is k given a set (or multiset) of integers. So, if we were to work with just one person, like Alice, and ignore Bob, then we can have Alice owe D dollars with a wallet containing $d_1, ..., d_n$ dollar bills. Because we are ignoring Bob, Alice must pay for her food with the exact amount by herself. Thus, we have reduced the Subset Sum problem to this problem.

2. This Public Funds problem can also be reduced from the Subset Sum problem since the Subset Sum problem is to see if there a non-empty subset whose sum is k given a set (or multiset) of integers. Since you must use the whole balance of a particular bank account if you choose to use that bank account, one can consider the balance as a bill worth that much money. For example, a bank account with balance of \$1000 will be the same thing as a single \$1000 bill. So with this, to pay for the fences, you must pay the exact amount using the bills (bank accounts) that you have. Furthermore, since there is at most k bank accounts that can be used, we can limit the number of ”bills” that we use with k. Therefore, this Public Funds problem can be used to solve the Subset Sum problem, which is known to be **NP**-complete. Since Subset Sum is **NP**-complete, this means that Public Funds is also **NP**-complete.

3. This Max-Acyclic-Induced-Subgraph problem can be reduced from the Independent Set problem because the Independent Set problem attempts to find a set of vertices in a graph where no two vertices are adjacent. For this problem, I will work with undirected graphs, which basically means that for every directed edge going from one vertex to another, there is another edge going the opposite direction. Because this creates a cycle, then this would not be considered for a DAG since a DAG has no cycles. Therefore, we can use the Max-Acyclic-Induced Subgraph problem to solve the Subset Sum problem with directed edges. Since the Subset Sum problem is **NP**-complete, this means that Max-Acyclic-Induced-Subgraph is also **NP**-complete.

4. Rad Reduction

Main Idea:
The main idea is to use the Independent Set idea problem that because you want to get a net profit of at least k, we should try to maximize our rewards and avoid penalties. So, to simplify the thought process, we can set all the rewards to be worth 1 and the penalty to be a large negative value like $-\infty$. Since we want to maximize profits, we should aim for not getting any penalties. So, the goal is to get rewarded k times. Because the Independent Set problem's goal is to find a set of vertices in a graph where no two vertices are adjacent to each other, we can avoid "bad" vertices by making them represent instances when there would be a penalty.

Reduction:
This Rad Reduction problem can be reduced from the Independent Set problem because the Independent Set problem attempts to find a set of vertices in a graph where no two vertices are adjacent. Because we are trying to avoid instances where we get penalized, we will represent penalties with adjacent vertices since the Independent Set problem avoids adjacent vertices. This way, when we are trying to get a net profit of at least k, we will be looking for k vertices that make up the Independent Set. Therefore, we can use this Rad Reduction problem to solve the Independent Set problem. Since the Independent Set problem is **NP**-complete, this means that this Rad Reduction problem is also **NP**-complete.

Proof:
The proof for this is Rad Reduction Problem is a generalization of the Independent Set problem. Since we know that the Independent Set problem is **NP**-complete, I will need to show that the Independent Set problem reduces to the Rad Reduction problem to prove that the Rad Reduction problem is **NP**-complete. Since we are aiming for a profit of at least k, we can just aim to maximize profit as much as possible by avoiding animal attacks that lead to us getting fined. Since the Independent Set problem avoids vertices that are adjacent, it is fitting to represent those vertices with the animals that will get attacked so that they will not be picked. So, we can solve the **NP**-complete Independent Set problem with this Rad Reduction problem, which in turn means that this Rad Reduction problem is **NP**-complete. □