Computer Science 180, Homework 3

Michael Wu
UID: 404751542

February 1st, 2018

Chapter 8, Problem 15

We will show that the Nearby Electromagnetic Observation problem is NP-complete by first showing that a solution can be checked in polynomial time, then by showing that the Vertex Cover problem reduces to the Nearby Electromagnetic Observation problem.

To show that a solution can be checked in polynomial time, we show that we can verify whether a set of locations is sufficient in polynomial time. Assume there are n frequencies and m locations. Given a set of locations L' with size $k \leq m$ that form a sufficient set for the Nearby Electromagnetic Observation problem, we can check whether or not L' is correct by going through each of b interference sources (F_i, L_i) for $1 \leq i \leq b$. Begin with a set P of at most size nm containing all pairs (f_j, l_k) of locations in L' to all frequencies. For each interference source, if a location is blocked from a frequency remove the corresponding pair from P. Then check that the number of unique frequencies in P is equal to j. If this is true L' is a solution to the Nearby Electromagnetic Observation problem. This algorithm has the time complexity $O(bnm)$, proving that the Nearby Electromagnetic Observation problem can be verified in polynomial time. Thus it is NP.

To show that Vertex Cover$\leq_P$Nearby Electromagnetic Observation, we begin by noting that the Nearby Electromagnetic Observation problem is similar to finding the Vertex Cover of a bipartite graph, with one set of vertices F and another set of vertices L. Thus we can reduce the Vertex Cover problem to the Nearby Electromagnetic Observation problem by converting a graph $G = (V, E)$ to a bipartite graph. We do this by splitting each edge

$e \in E$ and inserting a vertex. Let this inserted vertex f be in the set F. Each vertex f only connects to vertices in V, thus we have a bipartite graph. V corresponds to the locations in the Nearby Electromagnetic Observation problem and F corresponds to the frequencies. Create a set of interference sources such that each frequency can only be received by two locations in V. These two locations correspond to the edges in the original graph G. Thus for the Vertex Cover problem, we can construct a set of frequencies, locations, and interference sources that represent G. Then solving the Nearby Electromagnetic Observation problem gives a set of locations that are sufficient for the representation of G. This set of locations also form a vertex cover for G, as each frequency will be accessed. The frequencies correspond to the edges in G, thus we obtain a set of vertices that cover each edge. This gives us a solution to the vertex cover problem. Thus

$$\text{Vertex Cover} \leq_P \text{Nearby Electromagnetic Observation}$$

which proves that the Nearby Electromagnetic Observation problem is NP-complete.

Chapter 8, Problem 18

We will show that the Decisive Subset problem is NP-complete by first showing that a solution can be checked in polynomial time, then by showing that the Vertex Cover problem reduces to the Decisive Subset problem.

A solution S to the Decisive Subset problem can be checked in polynomial time by iterating linearly through each of t issues I_i for $1 \leq i \leq t$, counting the n member's votes to come up with the original vote, then checking $|S|$ votes from the subset to see if they match the original. This algorithm has the time complexity $O(tn)$, so a solution to the Decisive Subset problem can be checked in polynomial time. Thus the Decisive Subset problem is NP.

To show that Vertex Cover$\leq_P$Decisive Subset, we attempt to transform an arbitrary undirected graph $G = (V, E)$ into an instance of the Decisive Subset problem, and show that a solution to the Decisive Subset problem yields a vertex cover. Let there be n vertices and t edges in G. We construct an instance of the Decisive Subset problem by making a voting member m_i for $1 \leq i \leq n$ correspond to the vertex v_i in G. We make an issue I_j for $1 \leq j \leq t$ correspond to the edge e_j in G. For each issue I_j, make the member m_i abstain from voting unless e_j connects to v_i. If v_i is connected to e_j, have member m_i vote yes. So for every issue, two members will vote yes and the rest will abstain. This yields a complete instance of a Decisive Subset problem, as we have determined every issue, member, and vote necessary. If we have a solution S to the Decisive Subset problem for a given k, on every issue the members in S must have at least one person vote yes, because the original vote consisted of all yes's on every issue. Thus our decisive subset corresponds to a vertex cover for the graph G, because each member $m_i \in S$ corresponds to a vertex $v_i \in V$ and each issue I_j corresponds to an edge e_j. Because every issue gets at least one yes vote, every edge is covered by one vertex. This results from us making member m_i vote yes on an issue I_j only if vertex v_i is connected to edge e_j. Thus the Vertex Cover problem reduces to the Decisive Subset problem. Since Vertex Cover is NP-complete, Decisive Subset must be NP-complete as well.

Chapter 8, Problem 22

First we need to split up $G = (V, E)$ into connected subgraphs. Do this as follows

```
initialize an empty set S of all subgraphs
for all v in V
    create a subgraph g
    add v to subgraph g
    remove v from V
    for all v_graph in subgraph g
        for all e in E
            if e connects v_graph with v_other
                add e to subgraph g
                add v_other to subgraph g
                remove e from E
                remove v_other from V
            endif
        endfor
    endfor
    add subgraph g to S
endfor
return S
```

Because this algorithm contains three loops over elements of G, it is $O(G^3)$, which is polynomial time.

Then for each subgraph $g \in S$, call A beginning with $k = 1$ on each subgraph and increasing k by one each iteration until A returns "no". Then we know the independent set g has a maximum independent set of size $k-1$. Summing the maximum independent set sizes of all subsets yields a result x, the maximum independent set size of G. This allows us to solve the Independent Set problem in polynomial time, as the number of calls to A is $O(G)$.

Chapter 8, Problem 31

We will show that the Undirected Feedback Set problem is NP-complete by first showing that a solution can be checked in polynomial time, then by showing that the 3SAT problem reduces to the Undirected Feedback Set problem.

A solution X to the Undirected Feedback Set problem can be checked in polynomial time by finding $G - X$, then performing a breadth-first search on $G - X$ to check for cycles. A cycle is found if an already visited vertex is connected to the vertex being searched. In the worst case scenario, no cycle is found and every edge and vertex is traversed. This only requires linear time, as we simply step through the graph in the case that no cycle is detected. Because our breadth-first search is $O(G)$, the Undirected Feedback Set problem is $O(G)$.

We reduce the 3SAT problem to the Undirected Feedback Set problem by constructing cycles for each variable and clause. Let there be n variables x_i for $1 \leq i \leq n$ and m clauses C_j for $1 \leq j \leq m$ in our 3SAT problem. For each variable, we construct three vertices labeled $x_i = 0$, $x_i = 1$, and $x_i =$ any, and connect them all to each other to form a triangle. So we have n cycles, meaning that if we were to produce a feedback set X, it must have at least n vertices. The feedback set of size n must contain a unique value for each variable x_i. For each clause C_j, create three edges connecting each of the three variables in the clause. For example if a clause is $(x_1, \overline{x_2}, x_3)$, draw edges $(x_1 = 1, x_2 = 0)$, $(x_2 = 0, x_3 = 1)$, and $(x_3 = 1, x_1 = 1)$. Then the clause forms a cycle that is broken if and only if one of its element is in the feedback set X, satisfying that clause. This completes our construction of $G = (V, E)$ for our Undirected Feedback Set problem. We then ask if there is a feedback set of size at most $k = n$. If yes, we know that there also exists a satisfying assignment for our 3SAT problem, as this means that there is a unique value for each variable x_i such that each clause is satisfied, because if a clause is not satisfied then it would form a cycle in $G - X$. The vertices in X give a solution to the 3SAT problem, proving that we can solve the 3SAT problem in polynomial time if we are given a polynomial time algorithm for the Feedback Set problem. Thus

$$\text{3SAT} \leq_p \text{Feedback Set}$$

and so Feedback Set is NP-complete.

Chapter 8, Problem 36

We will show that the Daily Special Scheduling problem is NP-complete by first showing that a solution can be checked in polynomial time, then by showing that the Undirected Hamiltonian Path problem reduces to the Daily Special Scheduling problem.

To verify that a solution X of the Daily Special Scheduling problem costs at most x dollars, simply take the ordering X of the k specials, traverse the ordering, and buy the necessary ingredients for each special. Make sure to keep track of each of the leftover ingredients along the way that are still good enough to use. At the end, total the ingredients bought and multiply by the cost of the ingredients bought to find the total cost of the ordering X. If this cost is less than or equal to x, the solution X is valid. Because we have k specials and n ingredients, This algorithm has the time complexity $O(kn)$, and so the Daily Special Scheduling problem is NP.

We will make some simplifying assumptions about the Daily Special Scheduling problem to help us reduce the Undirected Hamiltonian Path problem to the Daily Special Scheduling problem. Let every special have the set of required fresh ingredients be the set $F = \emptyset$, so that no special requires fresh ingredients. Let the unit size $s(j) = 2$ for ingredient I_j, so they all must be bought in multiples of 2 grams. Let each unit cost $c(j) = 1$ dollar. Let each ingredient last for $t(j) = 2$ days, so that they can only be used on their date of purchase and on the following day. Now we wish to convert an arbitrary undirected graph $G = (V, E)$ into an instance of the Daily Special Scheduling problem. We wish to make the k vertices v_i for $1 \leq i \leq k$ correspond to the specials s_i. For every special s_i, we require 1 gram of each of k unique ingredients I_{ij} for $1 \leq j \leq k$. Thus we have created a k^2 long list of ingredients. In this state, the total amount of money required to produce all the specials is $s(j)c(j)k^2 = 2k^2$. For each edge $e = (v_a, v_b)$ in our original graph G, we now change ingredient I_{ab} to be the same as I_{ba}. Effectively, this ties together the specials (s_a, s_b). Instead of only using 1 gram out of the 2 grams of I_{ba} and 1 gram out of the 2 grams of I_{ab} that we purchase, we can arrange s_a and s_b sequentially so that they can both use the same ingredient, saving 2 dollars. If the specials are ordered in such a way that every special after the first saves 2 dollar, our total cost will be $2k^2 - 2k + 2$ dollars. If we can find an ordering of k specials that costs at most $2k^2 - 2k + 2$ dollars using a solution to the Daily Special Scheduling problem, we have then found a Hamiltonian path through the undirected graph G. Because every special

s_i is scheduled with a cost of at most $2k^2 - 2k + 2$ dollars, we know that every vertex v_i can be visited in a corresponding manner such that an edge connects each vertex. Thus

$$\text{Undirected Hamiltonian Path} \leq_p \text{Daily Special Scheduling}$$

This proves that the Daily Special Scheduling problem is NP-complete.