Daniel Campos

December 3rd, 2012

Algorithms 2300 Section 5

HW 7

1. 8.1

In order to solve we make a polynomial amount of calls to TSP and vary an input, b by using a binary search. In this case T is the sum of all the edge weights present in graph G.

The tour is at most T and a minimum of zero and we use these bounds to use binary search. The first step to get this is to check the TSP with the b value being T/2. If it returns false then it is know that the tour must be greater than or equal to T/2. If not we know that it must be less than or equal to T/2. Using the principles of binary search we continue to do this until we find the min edge length in the graph. This will result in a total amount of calls being log T. It is known that log T will be polynomial since when F is the edge weight in G and we replace all the edge weights with F then at most |E|Log F bits in order to give a representation of T. Which is also polynomial in input size. Therefore we make only polynomial calls to TSP using binary search and that solves TSP-OPT which shows it is. NP-Hard

1. 8.4
   1. Clique-3 has at each time a graph G and an integer k. Each one of the answers has a collection in the form of a set S of k amount of vertices. By using the checking algorithm C(G,k,S) we are able to see if there are the correct amount of vertices in S and each one has an edge that goes to the other k-1 vertices present in the set. This is done in O(|G|+k) time.
   2. This goes the incorrect way. It should form a understood and know NP-complete problem and it does not. We want to show something like a clique-3.
   3. Using the following example. The set of numbers 1🡨🡪2🡨🡪3. {2} is a vertex with size 1, 1🡨🡪3 is not a clique. Therefore the statement based on the complement of a vertex cover creating a clique is wrong.
   4. When you are given a set (G; F)

If(f>3){return “no clique, vertices are only possible adjacent to 3 other vertices”;}

Else if(f==1){ take vertex and return it}

Else if{f==2||f==3}{search the f elements in the subset V to find a kcliqque. If one it is found, if not “no k-clique” is returned;}

There are n(n-1)/2 subsets of V with a size of 2 and n(n-1)(n-2)/6 subsets of V with a size of 3

In the cases where f is 2 or 3 the time would be O(G^3)

1. 8.10
   1. This can be seen as a clique problem. An input (G,,k) for the Clique and H is a graph that has k vertices and every pair is connected by an edge so that a clique of size k. If H is a sub graph of G then G will contain a clique of size k
   2. This is based of the Rudrata-path idea. When a graph G with n vertices, g=n-1. Based on that (G,n-1) will give the instance of the longest path. But a path with a length n-1 must contain n vertices and therefore will be a Rudrata path. Besides that since a rudrata path is length n-1 and hence for a graph with n vertices LONGEST-PATH(G,n-1) is asking for the Rudrata path. I think