CSE 60111: Complexity and Algorithms
Homework 4

Reminder: For NP-Completeness, you must prove several things:

- First, show the problem is in NP: namely, that there is a polynomial size certificate along with a way to verify the ”yes” answer in polynomial time. (Any polynomial will do here, so no need to make it as fast as possible.)
- Second, a reduction from a known NP-Hard problem from the list in our textbook to an instance of the new problem. This reduction must take polynomial time and space, and have a ”yes” answer if and only if the original problem also has a ”yes” answer.

(The expectations for any algorithms questions are the same as the last few homeworks: give the algorithm/pseudocode, the time and space complexity along with justification, and a proof of correctness.)

---

1. (a) There's something special about the number 3: Give a polynomial time algorithm for 2-coloring: given a graph $G = (V, E)$, decide in polynomial time whether G has a proper coloring using only 2 colors.

   (b) Actually, there's nothing special about the number 3: Show that the problem 11-color is NP-Complete: Given a graph $G = (V, E)$, determine if we can properly color G with 11 colors.

2. You are given a graph $G = (V, E)$ with a weight $w(e)$ on every edge $e \in E$. The weights can be positive or negative. You are now asked to solve the following problem: Decide if there is a simple cycle in G such that the sum of edge weights in the cycle sums to exactly 0. Prove that this problem is NP-Complete.

3. There are many different ways to formalize the intuitive problem of clustering, where the goal is to divide up a collection of objects into groups that are "similar" to one another.

   First, a natural way to express the input to a clustering problem is via a set of objects $1..n$, with a numerical distance $d(i,j)$ defined on each pair. (We require only that $d(i,i) = 0$; that $d(i,j) > 0$ for $i \neq j$; and that distances are symmetric: $d(i,j) = d(j,i)$.)

   Sometimes this can be easy: we already saw an approximation algorithm. Another reasonable formulation of the clustering problem can be solved in polynomial time: Divide the objects into k sets so as to maximize the minimum distance between any pair of objects in distinct clusters. (This version turns out to be solvable by a nice application of the Minimum Spanning Tree Problem.)

   A different but seemingly related way to formalize the clustering problem would be as follows: Divide the objects into k sets so as to minimize the maximum distance between any pair of objects in the same cluster. Note the change: Where the formulation in the previous paragraph sought clusters so that no two were "close together," this new formulation seeks clusters so that none of them is too "wide"–that is, no cluster contains two points at a large distance from each other.

   Given the similarities, it's perhaps surprising that this new formulation is computationally hard to solve optimally. To be able to think about this in terms of NP-completeness, let's write it first as a yes/no decision problem. Given n objects with distances on them stored in a matrix $D[i,j]$ and a fixed value b, and a number k, we define low diameter clustering to be the following: Can the objects be partitioned into k sets so that no two points in the same set are at distance greater than b from each other? Show that this problem is NP-Complete.