HW 10

Due on **Wednesday, December 6, 2017 at 10am**

CS/ECE 374: Algorithms & Models of Computation, Fall 2017 Version: **1.01**

Submission instructions as in previous homeworks.

1 (100 pts.) Things are hard.

1.A. (40 pts.) Let G be an undirected graph with n vertices and m edges. Given a parameter $k > 0$, consider a problem of finding a subset $S \subseteq \mathsf{V}(\mathsf{G})$, of size at most k, such that every edge uv in the graph is adjacent to a vertex of S, or alternatively, there is an edge vz, such that $z \in S$. We refer to this decision problem as 2VertexCover. Prove that 2VertexCover is NP-Complete.

1.B. (30 pts.) Let G be an undirected graph with n vertices and m edges. A subset $S \subseteq \mathsf{V}(\mathsf{G})$ is ***(t, k)-heavy***, if $|S| \leq t$, and every vertex in S is connected (directly by an edge) to at least k other vertices in S. Prove that deciding if there is a (t, k)-heavy subset in G is NP-Complete.

1.C. (30 pts.) In the zippy car problem, you are given n cars, and there are a set P of $2n$ distinct spots where the cars might be parked. For every car c_i there are two spots $s_i, s_i' \in P$ where it might be parked (you can assume no two cars are assigned the same spot). There are also m customers that might rent these cars. The jth customer can access only a subset of the spots $P_j \subseteq P$. The question is whether there is way to park the cars in the spots, such that every customer has access to a parked car (naturally, the same car, might be available to many customers – not everybody is going to drive in the same time).

Prove that this problem is NP-Complete.

2 (100 pts.) Reserving paths.

Consider the following problem. You are managing a communication network, modeled by a directed graph $G = (V, E)$. There are c *users* who are interested in making use of this network. User i (for each $i = 1, 2, \ldots, c$) issues a *request* to reserve a specific path P_i in G on which to transmit data.

You are interested in accepting as many of these path requests as possible, subject to the following restriction: if you accept both P_i and P_j, then P_i and P_j can not share any modes.

Thus the Path Selection decision problem asks: Given a directed graph $G = (V, E)$, a set of requests $P_1, \ldots, P_c$-each of which must be a path in G- and a number k, is it possible to select at least k of the paths so that no two of the selected paths share any nodes?

This problem is NP-Complete. But assume that you were given an access to the oracle of Delphi, denoted by **orac**, who is willing to tell you, given an instance of Path Selection whether or not it is feasible. Furthermore, the oracle is in a good mood, and she is willing to answer as many such questions as you might have.

In the optimization version of the problem, you have to compute the maximum number of paths that can be allocated (i.e., you are not given k).

Describe a polynomial time algorithm, that uses **orac**, and computes and outputs the optimal solution (i.e., the maximum number of paths that can be allocated without a conflict, and the paths themselves).

3 (100 pts.) Almost independent.

A subset S of vertices in an undirected graph G is ***almost independent*** if for any vertex in S, there is at most one other vertex of S that it is adjacent to it in G. Prove that finding the size of the largest almost independent set of vertices in a given undirected graph is NP-Hard.