Introduction to Algorithms	Problem Set 9
CS 482, Spring 2008	Due Friday, April 25

(1) Solve Chapter 11, Exercise 1 in Kleinberg & Tardos.

(2) For each of the following optimization problems, present an integer program whose optimum value matches the optimum value of the given problem. The combined number of variables and constraints in your integer program should be polynomial in the size of the given instance of the optimization problem.

(a). Set Cover. Given a universal set $\mathcal{U}$ and a collection of subsets $S_1, S_2, \ldots, S_m \subseteq \mathcal{U}$, what is the minimum size of a subcollection $\{S_{i_1}, S_{i_2}, \ldots, S_{i_k}\}$ whose union is $\mathcal{U}$?

(b). Independent Set. Given a graph $G = (V, E)$, find an independent set of maximum cardinality.

(c). Max-3SAT. Given a set of Boolean variables $x_1, x_2 \ldots, x_n$ and a set of clauses $C_1, C_2, \ldots, C_m$ each consisting of a disjunction of 3 literals from the set $\{x_1, \bar{x}_1, \ldots, x_n, \bar{x}_n\}$, what is the maximum number of clauses that can be satisfied by a truth assignment?

(d). Max-Cut. Given an undirected graph $G = (V, E)$, what is the maximum size of a cut? (A cut (A, B) is any partition of the vertex set V into two nonempty subsets. The size of a cut is equal to the number of edges with one endpoint on each side of the partition.)

It is not necessary to prove that your answer is valid. However, you should explain your notation well enough that we completely understand the structure of your integer program.

Example: In the weighted vertex cover problem, one is given a graph $G = (V, E)$ and a non-negative weight w_v for every vertex $v \in V$. One is asked to find the minimum total weight of a vertex cover. The equivalent integer program is:

$$\begin{array}{ll} \min & \sum_{v \in V} w_v x_v \\ \text{s.t.} & x_u + x_v \geq 1 \ \text{ for all edges } e = (u, v) \\ & x_v \in \{0, 1\} \ \text{ for all vertices } v \end{array}$$

(3) After graduating from Cornell, you decide to use your algorithms training for the benefit of humanity by joining the recently created Federal Bureau of Domestic Spying (FBDS). To avoid an election-year fiasco, Congress passed a piece of "compromise legislation" giving the FBDS the authority to secretly wiretap some people's phones without a warrant, but the number of such people (called "targets of investigation") is limited to be no greater than k on any given day. The FBDS can monitor any phone call as long as at least one of its participants is a target of investigation, otherwise they can't monitor the call.

Your first day on the job, the FBDS asks you to write a program that takes a list of phone calls (each specified by a pair of citizens) and computes a list of up to k targets of investigation, to maximize the combined number of phone calls that can be monitored.

"That's impossible," you say. "The problem of deciding if we can legally monitor *every* phone call is equivalent to Vertex Cover, which is NP-Complete."

"Then design an approximation algorithm," they say. (This is assuming they don't say, "We've known for years that P=NP, but it's a national secret.") Give a polynomial-time 2-approximation algorithm for this problem. In other words, your polynomial-time algorithm should output a list of up to k targets of investigation, such that the combined number of phone calls that can be monitored using these targets of investigation is at least half the maximum number of phone calls that could be legally monitored.