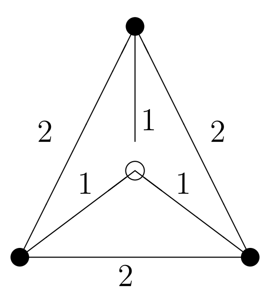
CS 3510 - Honors Algorithms
Homework 8
Assigned April 19
Do not hand this in, but please do it!!!

1. For a graph $V = (G, E)$, we say $D \subseteq V$ is a *dominating set* if every vertex $v \in V$ is either in D or adjacent to at least one member of D. Notice that a triangle has a dominating set of size 1, but the minimum vertex cover has size 2.

   The *dominating set problem* (DS) is: Given $G = (V, E)$ and an integer k, does there exist a dominating set of size at most k. Show that DS is **NP**-complete.

   (Hint: Consider using Vertex Cover for your reduction. Notice that a vertex cover is a set of vertices that covers all of the edges, and a dominating set is a set of vertices that covers all the vertices (by edge adjacencies). Thus, it seems to make sense to try to add "dummy" vertices along the edges of the original graph. But this isn't quite enough yet. Think about it carefully.)

2. We will consider the *Steiner Minimum Tree* (SMT) problem. We are given a graph $G = (V, E)$ with weight function w, and a set of vertices $V' \subseteq V$ of *terminal nodes*. We want to find a minimum-cost tree $T \subseteq G$ that spans the vertices of V'. The tree T may use nodes in $V - V'$. For instance, consider the graph, where the dark vertices are in V' and the middle vertex is not:

   Notice that the minimum tree for connecting the dark vertices in V' without using other vertices has total weight 4, but using the middle vertex we can achieve this with weight 3. The tree that achieves the minimum weight (possibly using vertices in $V - V'$) is called the *Steiner tree* and any vertex in T but not in V' is called a *Steiner node*. The SMT problem takes input $G = (V, E), V' \subseteq V, C$ and asks whether or not there is a Steiner tree of weight at most C.

   Let us suppose our graph satisfies the triangle inequality: $w(x, y) + w(y, z) \geq w(x, z)$ for all $x, y, z \in V$. Let T' be the minimum spanning tree on V'. Show that T' is a 2-approximation for the SMT problem (i.e., if T is a minimum SMT, then $w(T') \leq 2w(T)$). there is a Steiner tree

3. The max-flow problem can be generalized in many ways. Some examples are:

   (a) There are many sourcs and sinks, and we wish to maximize the total flow from all sources to all sinks.

   (b) Each edge has not only a capacity, but also a *lower bound* on the flow it must carry.

   (c) The outgoing flow from each vertex v is not the same as the incoming flow, but it is smaller by a factor of $(1-\epsilon_v)$, where ϵ_v is a loss coefficient associated with the vertex v.

   (d) Each edge has a cost per unit flow associated with it, and we must find, among all flows of maximum value, the one that minimizes the total cost.

   In each case, show how to solve the more general problem by (1) reducing it to the original max-flow problem whenever possible, or (2) reducing it to linear programming in the remaining cases. (Recall that we could solve max-flow by setting up a series of linear inequalities and giving a linear objective function to maximize or minimize. If you cannot see a direct reduction to the original max-flow problem, try to modify the linear program given in class and in the notes.)

   (Careful: if you want actual algorithms, think about which way your reductions need to go!)

4. Suppose someone presents you with a solution to the max-flow problem on some network. Give a *linear* time algorithm to determine whether the solution does indeed give a maximum flow.

   (Here assume that the input is just the graph and the solution is to the optimization problem, not the decision version of the problem.)