Fall 2007 **CMSC 858K: Homework 7** Samir Khuller

Due in class: Nov 20.

(1) Consider the following "prim like" algorithm for the Steiner tree problem. Given a graph $G = (V, E)$ with positive weights on the edges, and a special subset S of the vertices. We wish to obtain a steiner tree that spans all the vertices in S. Let $T_1 = \{s_1\}$. At each step T_{i+1} is computed from T_i as follows: attach the vertex from $S - T_i$ that is the "closest" to T_i by a path to T_i and call the newly added special vertex s_{i+1}. (Thus T_i always contains the vertices $s_1, s_2, \ldots, s_i$.)

Prove that this algorithm finds a steiner tree whose cost is at most twice the minimum weight steiner tree.

(2) Design an approximation algorithm for the Traveling Salesman Path problem with specified end-points. The path must start and end at pre-sepcified vertices s and t. Again, like for Christofides method assume triangle inequality.

(Hint: design two algorithms, one that does well when s and t are "close" together and one that does well when they are "far" apart.)

(3) Consider the *rural Chinese postman* problem: Give an undirected graph $G = (V, E)$ with weights on the edges satisfying the triangle inequality, and a specified subset E' of edges, the goal is to find a "short" tour that visits all the edges in the subset E'. In other words, find a tour in which all edges in E' are traversed. (Edges denote streets and all streets are two-way. Edges may be visited more than once in the solution.)

Design a polynomial time approximation algorithm with a 1.5 approximation factor for this problem. (For partial credit design a constant factor approximation.)

(4) Suppose you are a restaurant critic. There are n restaurants you need to compare. You sample a few individuals and ask for their input on "comparing" pairs of restaurants. If a person says that j is better than i, you add a directed edge from i to j. You now obtain a graph. Clearly, this graph may have cycles since different people have different recommendations. Your goal is to produce a DAG (Directed Acyclic Graph) that provides a consistent set of recommendations (or a partial order among the restaurants). How would you find a DAG that maximizes the number of edges in it? This solution might be one that represents the collective opinions in the best way. Develop a polynomial time algorithm to give a $\frac{1}{2}$ approximation.

(5) In class we studied the vertex cover problem in which our goal was to cover *all* the edges using minimum weighted set of vertices. Consider the problem of covering a *fixed* number of the edges instead of all the edges. Let the parameter p which is part of the input denote the number of edges to be covered. The integer program for this problem is given below. The LP can be obtained by relaxing the integrality constraints.

$$\begin{aligned}
\text{min} \quad & \sum_{j=1}^{n} w_j \cdot x_j \\
\text{subject to} \quad & \\
& x_i + x_j \geq y_{ij}, \forall (v_i, v_j) \in E \\
& \sum_{(v_i, v_j) \in E} y_{ij} \geq p \\
& x_j \in \{0, 1\}, j = 1, 2, \ldots, n
\end{aligned}$$

$$y_{ij} \in \{0,1\}, \forall (v_i, v_j) \in E$$

In this exercise we will analyse the algorithm, P-COVER, that is given below.

P-COVER$(G = (V, E), w, p)$

Phase 0:
if $|\{(u,v) \in E| < p$ **then return** error
Solve the linear program to get $x_1, x_2, \ldots, x_n$ and $\{y_{ij}|(v_i, v_j) \in E)\}$.
Copy these values to $x_1', \ldots, x_n'$ and $\{y_{ij}'|(v_i, v_j) \in E\}$
Phase 1:
$C_1 \leftarrow \{v_i | x_i' \geq \frac{1}{2}\}$
Let (V', E') be the induced subgraph on $V' = V \setminus C_1$
Let $d'(v_i)$ be the degree of vertex v_i in (V', E')
Phase 2:
$C_2 \leftarrow \emptyset$
while $\exists$ distinct v_r and v_g in V' with $x_r' > 0, x_g' > 0$ and $\frac{d'(v_r)}{w_r} \geq \frac{d'(v_g)}{w_g}$ **do**
    $\delta \leftarrow \min\{\frac{1}{2} - x_r', \frac{w_g}{w_r} x_g'\}$
    $x_r' \leftarrow x_r' + \delta$
    $x_g' \leftarrow x_g' - \frac{w_r}{w_g}\delta$
    for each edge $(v_l, v_k) \in E'$ **do**
        $y_{lk}' \leftarrow x_l' + x_k'$
    if $x_r' \geq \frac{1}{2}$ **then**
        $C_2 \leftarrow C_2 \cup \{v_r\}$
        Remove v_r and adjacent edges from (V', E')
        Update $d'(v_k)$ for all v_k in the new subgraph (V', E')
Phase 3:
if $\exists v_i$ with $x_i' > 0$ and $C_1 \cup C_2$ covers $< p$ edges **then**
    $C_3 \leftarrow \{v_i\}$
else $C_3 \leftarrow \emptyset$
return $C_1 \cup C_2 \cup C_3$

Let x_i' and y_{ij}' refer to the *final* values in the algorithm.

(a) Consider the following rounding scheme which we know works for the case when $p = |E|$. Solve the LP to get $x_i \in [0,1]$ for each vertex v_i, and let the cover be given by $C = \{v_i | x_i \geq \frac{1}{2}\}$. This algorithm has a performance ratio of 2 when $p = |E|$. Give a counter example to prove that the set $\{v_i | x_i \geq \frac{1}{2}\}$ may not be feasible when $p < |E|$.

(b) Prove that the total value of objective function remains the same during phase 2 of the algorithm P-COVER.

(c) The final solution, x_i' and y_{ij}', may not satisfy some of the constraints of the LP. List the constraints that may not be satisfied and prove the satisfiablity of the constraints that will be satisfied. Be sure to examine all the constraints in the LP.

(d) Prove that P-COVER gives a solution of cost at most $2 \cdot OPT + H$, where H is the heaviest weight of any vertex in G.

(e) Modify the algorithm slightly to improve the performance ratio to 2. (Hint: "Guess" the weight of the heaviest vertex in the optimal solution).