CS 473G: Graduate Algorithms, Spring 2007
Homework 4

Due March 29, 2007

Please remember to submit **separate, individually stapled** solutions to each problem.

---

1. Given a graph G with edge weights and an integer k, suppose we wish to partition the the vertices of G into k subsets $S_1, S_2, \ldots, S_k$ so that the sum of the weights of the edges that cross the partition (*i.e.*, have endpoints in different subsets) is as large as possible.

   (a) Describe an efficient $(1 - 1/k)$-approximation algorithm for this problem.

   (b) Now suppose we wish to minimize the sum of the weights of edges that do *not* cross the partition. What approximation ratio does your algorithm from part (a) achieve for the new problem? Justify your answer.

2. In class, we saw a $(3/2)$-approximation algorithm for the metric traveling salesman problem. Here, we consider computing minimum cost Hamiltonian *paths*. Our input consists of a graph G whose edges have weights that satisfy the triangle inequality. Depending upon the problem, we are also given zero, one, or two endpoints.

   (a) If our input includes zero endpoints, describe a $(3/2)$-approximation to the problem of computing a minimum cost Hamiltonian path.

   (b) If our input includes one endpoint u, describe a $(3/2)$-approximation to the problem of computing a minimum cost Hamiltonian path that starts at u.

   (c) If our input includes two endpoints u and v, describe a $(5/3)$-approximation to the problem of computing a minimum cost Hamiltonian path that starts at u and ends at v.

3. Consider the greedy algorithm for metric TSP: start at an arbitrary vertex u, and at each step, travel to the closest unvisited vertex.

   (a) Show that the greedy algorithm for metric TSP is an $O(\log n)$-approximation, where n is the number of vertices. *[Hint: Argue that the kth least expensive edge in the tour output by the greedy algorithm has weight at most $\mathrm{OPT}/(n-k+1)$; try $k = 1$ and $k = 2$ first.]*

   *(b) **[Extra Credit]** Show that the greedy algorithm for metric TSP is no better than an $O(\log n)$-approximation.

4. In class, we saw that the greedy algorithm gives an $O(\log n)$-approximation for vertex cover. Show that our analysis of the greedy algorithm is asymptotically tight by describing, for any positive integer n, an n-vertex graph for which the greedy algorithm produces a vertex cover of size $\Omega(\log n) \cdot \mathrm{OPT}$.

5. Recall the minimum makespan scheduling problem: Given an array $T[1..n]$ of processing times for n jobs, we wish to schedule the jobs on m machines to minimize the time at which the last job terminates. In class, we proved that the greedy scheduling algorithm has an approximation ratio of at most 2.

   (a) Prove that for any set of jobs, the makespan of the greedy assignment is at most $(2-1/m)$ times the makespan of the optimal assignment.

   (b) Describe a set of jobs such that the makespan of the greedy assignment is exactly $(2-1/m)$ times the makespan of the optimal assignment.

   (c) Describe an efficient algorithm to solve the minimum makespan scheduling problem *exactly* if every processing time $T[i]$ is a power of two.