Homework Assignment 1

Due: Thursday, May 1, in class.

General remark: whenever you are asked to provide an α-approximation algorithm, you need to prove that your algorithm produces a feasible α-approximate solution.

1. **Greedy algorithms for Job Interval Scheduling.** Recall that in the job interval scheduling problem we have a set J of jobs, and each job $j \in J$ has a release date r_j, a deadline d_j and a processing time p_j. Job j is associated with a set $\mathcal{I}_j$ of time intervals: the set of all length-p_j intervals contained in $[r_j, d_j]$. The goal is to schedule maximum possible number of jobs on one machine. We have shown in class that the following algorithm achieves a factor 2-approximation:

GREEDY
Among all available jobs, choose job j that has an interval $I \in \mathcal{I}_j$ with left-most right endpoint; schedule j on I and discard all job intervals overlapping with I.

   A mirror reflection of GREEDY is an algorithm we call GREEDY':

GREEDY'
Among all available jobs, choose job j that has an interval $I \in \mathcal{I}_j$ with right-most left endpoint; schedule j on I and discard all job intervals overlapping with I.

   Algorithm GREEDYx2 Runs GREEDY and GREEDY' and returns the better of the two solutions.

   (a) Show that GREEDYx2 achieves a factor 2-approximation.

   (b) Show that your analysis is asymptotically tight.

2. **Routing in SONET ring.** The following problem arises in telecommunications networks, and is known as the SONET ring loading problem. The network consists of a cycle on n nodes, numbered 0 through $n-1$ clockwise around the cycle. Some set C of calls is given; each call is a pair (i, j) originating at node i and destined to node j. The call can be routed either clockwise or counterclockwise around the ring. The objective is to route the calls so as to minimize the total load on the network. Let $e_1, \ldots, e_n$ denote the edges of the cycle. The load L_i on edge e_i is the number of calls routed through it, and the total load is $\max\{L_i \mid 1 \leq i \leq n\}$. Design a 2-approximation algorithm for the SONET ring loading problem.

3. **Steiner Tree.** In the directed Steiner Tree problem, we are given a directed edge-weighted graph $G = (V, E)$, a root vertex r and a subset $T \subseteq V$ of vertices called terminals. The goal is to find a minimum-cost subset E' of edges, such that in the graph induced by E', there is a directed path from r to every terminal in T.

   (a) Show an approximation-preserving reduction from Set Cover to directed Steiner Tree.

   Hint: Given an instance of Set Cover, construct a 3-layered instance of Steiner Tree. First layer contains the root r, second layer contains Steiner vertices and third layer contains terminals. Prove that your reduction is approximation-preserving.

(b) Recall that there is an $O(\log n)$-approximation algorithm for Set Cover, and there is no factor-$c \log n$ approximation algorithm (for some constant c) unless $P = NP$. What is the implication of your reduction to the approximability of directed Steiner Tree?

4. **Asymmetric Traveling Salesman Problem.**

   (a) Recall that we have shown in class an $O(\log n)$-approximation algorithm for the problem using Cycle Covers. Prove that our analysis of this algorithm is asymptotically tight.

   (b) Consider a special case of Asymmetric Traveling Salesman Problem, where all edge weights are either 1 or 2 (recall that from the problem definition, the input graph $G = (V, E)$ is a complete graph, so for each pair u, v of vertices of G, (u, v) and (v, u) belong to E). Design a factor-1.5 approximation algorithm for this problem.

5. **Local Ratio.** Given a weighted Set Cover instance $I = (U, \mathcal{S})$, where U is the set of elements and $\mathcal{S}$ is a collection of subsets of U, the *frequency* of element i is the number of sets in $\mathcal{S}$ containing i. Let f_I denote the frequency of the most frequent element. Show an f_I-approximation algorithm for weighted Set Cover using the Local Ratio technique.

Extra Credit. Consider the following algorithm for the Minimum Steiner Tree problem.

- Find a minimum-weight spanning tree τ of the graph. Root τ at any terminal t.
- While there is a leaf vertex v in the current tree τ, such that v is not a terminal, delete v from τ.
- Return the final tree τ. Notice that every leaf of τ is a terminal.

What is the approximation factor that this algorithm achieves? Show asymptotically matching upper and lower bounds.