CS261: Exercise Set #7

For the week of February 15–19, 2016

Instructions:

(1) *Do not turn anything in.*

(2) The course staff is happy to discuss the solutions of these exercises with you in office hours or on Piazza.

(3) While these exercises are certainly not trivial, you should be able to complete them on your own (perhaps after consulting with the course staff or a friend for hints).

Exercise 31

Recall Graham's algorithm from Lecture #13: given a parameter m (the number of machines) and n jobs arriving online with processing times $p_1, \ldots, p_n$, always assign the current job to the machine that currently has the smallest load. We proved that the schedule produced by this algorithm always has makespan (i.e., maximum machine load) at most twice the minimum possible in hindsight.

Show that for every constant $c < 2$, there exists an instance for which the schedule produced by Graham's algorithm has makespan more than c times the minimum possible.

[Hint: Your bad instances will need to grow larger as c approaches 2.]

Exercise 32

In Lecture #13 we considered the *online Steiner tree* problem, where the input is a connected undirected graph $G = (V, E)$ with nonnegative edge costs c_e, and a sequence $t_1, \ldots, t_k \in V$ of "terminals" arrive online. The goal is to output a subgraph that spans all the terminals and has total cost as small as possible. In lecture we only considered the *metric* special case, where the graph G is complete and the edge costs satisfy the triangle inequality. (I.e., for every triple $u, v, w \in V$, $c_{uw} \leq c_{uv} + c_{vw}$.) Show how to convert an α-competitive online algorithm for the metric Steiner tree problem into one for the general Steiner tree problem.[1]

[Hint: Define a metric instance where the edges represent paths in the original (non-metric) instance.]

Exercise 33

Give an infinite family of instances (with the number k of terminals tending to infinity) demonstrating that the greedy algorithm for the online Steiner tree problem is $\Omega(\log k)$-competitive (in the worst case).

Exercise 34

Let $G = (V, E)$ be an undirected graph that is connected and Eulerian (i.e., all vertices have even degree). Show that G admits an Euler tour — a (not necessarily simple) cycle that uses every edge exactly once. Can you turn your proof into an $O(m)$-time algorithm, where $m = |E|$?

[Hint: Induction on $|E|$.]

[1] This extends the $2 \ln k$ competitive ratio given in lecture to the general online Steiner tree problem.

Exercise 35

Consider the following online matching problem in general, not necessarily bipartite graphs. No information about the graph $G = (V, E)$ is given up front. Vertices arrive one-by-one. When a vertex $v \in V$ arrives, and $S \subseteq V$ are the vertices that arrived previously, the algorithm learns about all of the edges between v and vertices in S. Equivalently, after i time steps, the algorithm knows the graph $G[S_i]$ induced by the set S_i of the first i vertices.

Give a $\frac{1}{2}$-competitive online algorithm for this problem.