Problem Set 4

In the following problems involving graphs, m always represents the number of edges, and n always represents the number of nodes. You should also justify the correctness of your algorithms wherever it's not obvious.

1. A group of climbers has n ropes of length $\ell_1, \ldots, \ell_n$. They would like to climb a set of routes all of which require a rope of length h. Unfortunately none of the ropes are long enough, so they've decided to tie pairs of ropes together (assume that lengths add when the ropes are tied together). Give an $O(n \log n)$ algorithm to find the largest number of routes that they can climb simultaneously.

2. As in the interval scheduling problem, we have a collection of n tasks numbered 1 through n with task i starting and finishing at times $s(i)$ and $f(i)$ respectively. Only one task may be scheduled at any point in time. In an effort to be as annoying as possible, Bob would like to insist on scheduling the most inconvenient task. Formally, let $S(j)$ denote the maximum number of tasks that may be scheduled given that task j must be scheduled. Bob would like to choose j to minimize $S(j)$. Give an $O(n \log n)$ algorithm to help Bob find such a task.

3. You have a series of tasks to complete numbered 1 through n. Task i takes d_i days to complete. Once complete, it yields a revenue of r_i dollars per day. It's only possible to work on one task at a time and all tasks can be completed within T days. Give an $O(n \log n)$ time algorithm to find the ordering of the tasks which maximizes the revenue earned by day T. Show that your algorithm yields an optimal solution. (Hint: What's the optimal ordering when there are two tasks?)

4. Three friends, Alice, Bob, and Charlie live at different towns which are connected by a network of roads, modelled as an connected undirected graph. Traveling along edge e takes time $t(e)$. They're hungry, so they'd like to meet up for lunch as soon as possible. Find a town that

they can all reach in the smallest amount of time. Your algorithm should run in $O(m \log n)$ time.

5. An ambitious colony of aliens has set out to colonize a collection of planets with space probes. The planets are represented as nodes of a directed graph, and the edges represent pairs of planets with a direct route between them. Each planet p has some integer capacity $C(p)$. We say that p is at capacity if it contains at least $C(p)$ probes. Initially, planet s is at capacity and all others are empty. Every day, each planet which is already at capacity creates some new probes, and sends one such probe to all planets along outgoing edges which are not already at capacity. Find the smallest number of days after which planet t is at capacity. Your algorithm should run in $O(m \log n)$ time.

1 Good problems that you don't have to turn in.

1. Draw a graph and run Dijkstra by hand.

2. A network of roads is modelled as a graph with vertices representing connections between roads, and edges representing roads. Some roads might be one-way, so the graph is directed. The travel time along roads varies throughout the day. You have access to a function f, where $f(e, t)$ gives the travel time along road e at time t. For each road e, you're guaranteed that $t + f(e, t)$ is an increasing function of t (meaning that cars are guaranteed to get off a road in the order they got on). Given two nodes s and t, give an $O(m \log n)$ algorithm to find the shortest path between s and t.

3. Let G be a graph whose edge weights represent the capacity at which water can flow along that edge. Give nodes s and t give an efficient algorithm to find the path between s and t with maximum capacity. That is, find a path from s to t whose minimum edge weight is as large as possible.