October 6, 2015

CS 577 - Homework 4
Sejal Chauhan, Vinothkumar Siddharth, Mihir Shete

1 Graded written problem

Input: An alphabetical list with the names of all n kayakers for a given day, together with the times they will arrive at the parking lot.

Constraints: A single round-trip lasts **m** minutes and we can take **k** kayakers in one trip. The kayakers should be served with FCFS discipline but we can decide for each roundtrip when the bus leaves and how many kayakers we take along.

Output: Organize the schedule so that we are done with our task of droppping the kayakers to the kayak launch platform as early as possible.

1.1 Algorithm

We will solve this problem using a greedy strategy. Our strategy works by minimizing the number of trips and minimizing the starting time of each trip. Our algorithm starts by ordering the list by arrival time (earliest arrival time first). The running time for this initial sort is *O(n logn)*.

For this ordered list we will employ the following strategy:

1. If there are k_i kayakers in the parking lot at time t_i and there are no more scheduled arrivals in the interval $t \geq t_i$ to $t \leq t_i + \boldsymbol{m}$ then we should carry the k_i kayakers immediately. (if $k_i \geq \mathbf{k}$ then look at *Point 3*)
2. If there are k_i, k_{i+1}, k_{i+2} ... k_{i+j}, k_{i+j+1} kayakers arriving at t_i, t_{i+1}, t_{i+2} ... t_{i+j}, t_{i+j+1} and each arrival time is within m minutes of the immediate next arrival time (i.e $t_{i+1} - t_i \leq m$ and so on) and $k_i + k_{i+i} + ... + k_{i+j} \leq \mathbf{k}$ but if we add k_{i+j+1} then $\sum_i^{i+j+1} k_i > \mathbf{k}$. Now we know that to transport these many kayakers we will take atleast 2 trips (it can take more than 2 if k_{i+j+1} is much bigger than **k**). Now, to end our trips as early as possible we will aim to start our first trip as early as possible. So, what we do is find an arrival time t_s between t_i and t_{i+j+1} so that number of kayakers arriving from t_s to t_{i+j+1} is just $\leq \mathbf{k}$ (i.e if we include kayakers arriving at t_{s-1} number of kayakers in the interval will become $>\mathbf{k}$). We will start our first trip at t_{s-1} this way we are making sure that the number of trips are same but the starting time of first trip is minimized.
3. If at any point of time there are $\geq\mathbf{k}$ kayakers at the parking lot then we start the trip immediately with **k** kayakers.