October 7, 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 $\boldsymbol{m}$ minutes and we can take $\boldsymbol{k}$ kayakers in one trip. The kayakers should be served with FCFS discipline but we can decide for each round trip 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 the 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. Suppose there are k_1, k_2, k_3, ... k_n kayakers arriving in the parking lot at times t_1, t_2, t_3, ... t_n
2. Let t be the minimum number of trips required to drop $\sum_{i=i}^{n} k_i$ kayakers. Now we will start looking at the number of kayakers arriving at each time from the end of our sorted list. When $\sum_{i=n}^{n-j_p} k_i = \mathbf{k}$ we will group the kayakers to go in the same trip.(j_p can represent partial number of total kayakers arriving at t_j so that the sum is exactly $\mathbf{k}$). We will go on doing this grouping till the number of kayakers remaining are $\sum_{i=i}^{n} k_i \ mod \ \boldsymbol{k}$.

   We will start our first trip as soon as we have at least $\sum_{i=i}^{n} k_i \ \boldsymbol{mod\ k}$ (or $\mathbf{k}$ if $\sum_{i=i}^{n} k_i \ \boldsymbol{mod\ k} = 0$) kayakers at the parking lot. The second trip will begin when the number of kayakers waiting at the parking lot are $\boldsymbol{k}$. This way we are trying to minimize the start time of the first trip keeping the minimum number of trips constant.

   Note: Intuitively, if we start the first trip when kayakers are less than $(\sum_{i=i}^{n} k_i \ \mathbf{mod\ k})$ then we will need more than $\mathbf{t}$ trips and hence we *may* need more time, but if we start our first trip later than when $(\sum_{i=i}^{n} k_i \ \mathbf{mod\ k})$ kayakers arrive, our last trip *may* end a little later.

1.2 Greedy Stays Ahead Proof

Our greedy strategy is to accommodate as many kayakers on a trip from then last arrival time so that the trips taken to transport the kayakers are minimized and the first trip will start as early as possible. So the notion of 'time' in our greedy-stays-ahead approach will be the amount of time that is required for the trips to transport the remaining kayakers which are going to come in the future and are currently in the parking lot when the bus is at the lot. And our choice of ahead will be to simply say that the Greedy solution G at the current time has atmost same number of trips remaining as as solution S.

Let $t_G(i)$ be the number of trips required by the greedy strategy to carry all the kayakers that are at the parking lot and will come after time i. Let $t_S(i)$ be a similar quantity for S.

We claim that, $t_G(i) \leq t_S(i)$. So, at $i = 0$ the minimum number of trips required to transport all the $\sum_{i=i}^{n} k_i$ kayakers will be: $t = \lceil \sum_{i=i}^{n} k_i / \boldsymbol{k} \rceil$. Now, G starts the first trip only when $\sum_{i=i}^{n} k_i \ \boldsymbol{mod\ k}$ kayakers are present, if S starts a trip before that than after the trip is done it will still need to make t more trips. So after the first trip we say that our claim always holds.

We start the subsequent trips when we have alteast k kayakers at the parking lot so that the number of trips remain minimal. If S tries to start a trip before reaching the maximal capacity of the bus then it will have to make more trips and so it *may* take more time for S to transport all the kayakers.