Parth Ray

Assignment 4 – Greedy Proofs

1. Triathlon Competitor Scheduling
    a. Sort contestants in descending order of combined run and bike time. And this is your schedule for the smallest time to completion for the competition.
    b. Proof of correctness: Longest combined run and bike time algorithm finds the minimum completion time of the triathlon.

        Given n contestants P_1 to P_n, with combined run and bike times RB_1 to RB_n and swim times s_1 to s_n.

        Assume that there is an ordering $\sum^*$ that is the optimal solution but is different from $\sum$ the ordering given by the greedy algorithm.

    If $\sum^*$ is not $\sum$, it must have two consecutive contestants: i, j where j has the longer combined run and bike time but i is before j in $\sum^*$. Swapping the two contestants does not affect the maximum completion time of the contestants before and after the two contestants i and j. Since we know that j has the longer combined run and bike time, then $RB_i < RB_j$. Let C be the completion time of the contestants before i. Let S be the total swim time to completion before i. Then, before the swap our max completion time would look like:

    $\max\{\max(C_{\text{before i}}) , (S + s_i) + RB_i , (S + s_i + s_j) + RB_j , \max(C_{\text{after j}})\}$

    and the max time to completion after the swap would look like:

    $\max\{ \max(C_{\text{before j}}) , (S + s_j) + RB_j , (S + s_j + s_i) + RB_i , \max(C_{\text{after i}})\}$.

    Let's compare the max before the swap and after the swap assuming that max TTC of the triathlon before i and after j is less than or equal to the TTC of i and j. Thus, we have $\max_{\text{before}} \{(S + s_i) + RB_i , (S + s_i + s_j) + RB_j\}$ which would result in the max TTC of the triathlon being $(S + s_i + s_j) + RB_j$ since $RB_i < RB_j$. Now, we compare this max TTC value to that of $\max_{\text{after}} \{(S + s_j) + RB_j , (S + s_j + s_i) + RB_i\}$. First, $(S + s_i + s_j) + RB_j$ must be greater than $(S + s_j) + RB_j$ since their difference results in s_i which is greater than zero. Then, $(S + s_i + s_j) + RB_j$ is also greater than $(S + s_j + s_i) + RB_i$, since their difference results in $RB_j - RB_i$ which is greater than zero. Thus, since the $\max_{\text{before}}$ is greater than the possible maximums for $\max_{\text{after}}$, the new swapped contestant order cannot be worse than the optimal order. Now we know that every time we swap two consecutive contestants out of order, the max TTC of the triathlon cannot get any larger. These steps are repeatable for all two consecutive contestants that are out of order ($RB_i < RB_j$). Therefore, we can continue swapping in $\sum^*$, until no more pairs are out of order. Then, we will get that $\sum^*$ is equal to $\sum$ since both will be ordered from longest combined run and bike time and therefore have the same max TTC of the triathlon that is minimized.

    c. The complexity will be O(n log n) since we must sort the contestants in descending order of combined run and bike time.

2. Shipping Efficiently
    a. Proof of correctness: Packing the boxes in the order in which they arrive and once a box does not fit then send the truck on its way always ships the packages in the fewest trucks.

        Setup: Assume $\sum^* = \{t_1^*, \ldots, t_n^*\}$ is the optimal solution that ships all the packages in the fewest trucks. Let $\sum = \{t_1, \ldots, t_m\}$ be the set of trucks determined by the greedy algorithm specified above.

        Assume for contradiction that $\sum$ is not optimal, so $\sum^* \neq \sum$ and $m > n$. Then, there must be some box w_i in some truck t_k that is held off and is in a later truck t_{k+1}^*. Since w_i is in t_k and because the boxes must be shipped in the order of arrival, the greedy algorithm puts w_{i+1} in the same kth truck or $k+1^{st}$ truck. Then, because w_i is in t_{k+1}^* and since boxes are shipped in order of arrival, the optimal solution must put the w_{i+1} box in the same $k + 1^{st}$ truck or $k + 2^{nd}$ truck. Thus, for every package after w_i, say w_j, if the greedy algorithm puts w_j in some truck t_k then the optimal solution has w_j in the later truck t_{k+1}. Therefore, if the optimal holds at least one box for the next truck, it will always produce a larger set of trucks, than if we packed the truck to its maximum possible capacity every time. Thus, $n \geq m$. This contradicts that $\sum$ is not optimal. Therefore, $\sum$ is optimal.

    b. The algorithm complexity is O(n) since we are packing the boxes as they come, so we only have to go through the list of boxes of size n once.