CSCI 570 - Homework 03

- **Author:** Boyang Xiao
- **Due Date:** Sep. 14^{th} 2022
- **USC id**: 3326-7302-74
- **Email**: boyangxi@usc.edu

Problem 1

Answer:

1. We sort all ropes from the shortest to the longest.
2. We pick up the first two **shortest** ropes to connect them together.
3. We pick up the **shortest** one from the **remainings** till we connect all the ropes.

Problem 2

Answer:

- **Algorithms:**
  - We sort all the tasks in the descending order of the time of b_i(the second part).
  - We arrange the the task with largest b_i first and smallest b_i last.

- **Proof:**
  - First of all, since the computer A can only handle one task at a time, it makes no difference how we arrange the order of the first part of these tasks. So I am gonna prove why we arrange the second part of the tasks like these:
  - Suppose there is a way of arrangement where there are two tasks (a_i, b_i) and (a_j, b_j), and $b_i < b_j$. But in this arrangement, task i is arranged ahead of task j.
  - Since task i is arranged ahead of task j, the total time for computer B to finish these two tasks $time_b$ can be calculated as below. To note that b_i must be started ahead of b_j and their duration can have overlaps.

$$max(b_i, b_j) <= time_b < (b_i + b_j)$$

- However, in our optimal arrangement, task i is arranged ahead of task j, and the $time_b$ can be calculated by:

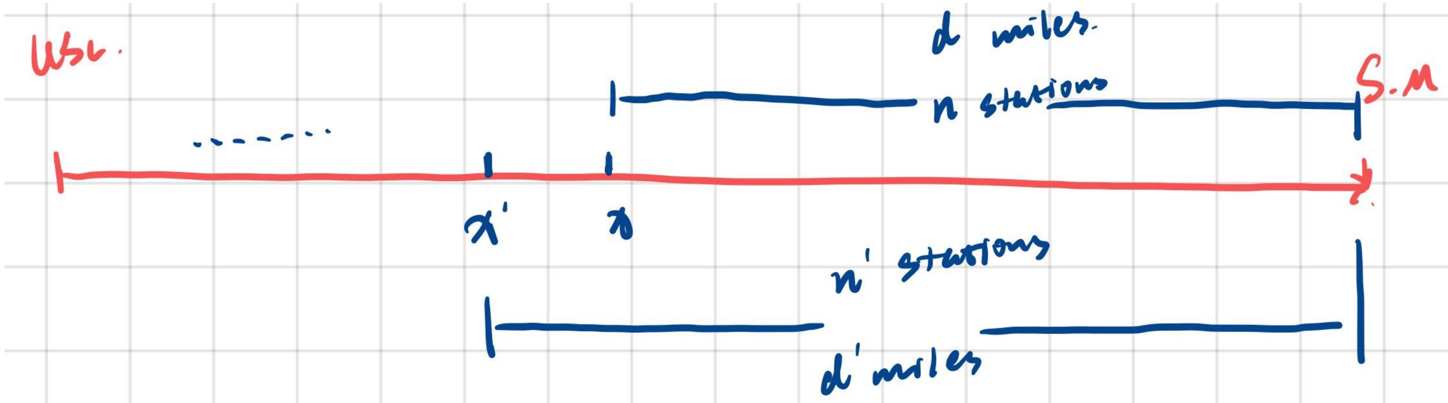
$$time_b = max(b_i, b_j)$$

- As we can see, when the time to finish all the first parts of tasks remain the same, arranging the second part of longest time first can obtain a minimized total time.

Problem 3

Answer:

- **Algorithms:**
  - Everytime we arrive at a gas station, we check the remaining milage that we can cover and compare it with the distance to the next gas station/Santa Monica(if there is no gas station left ahead).
  - If we can't make it to the next station, we stop at this gas station and fill up the tank.
  - Else, we pass by this gas station and repeat the loop at the next station.
- **Proof:**
  - Assume that in the algorithms we come up with above, there is one station along the route that we should stop by and there are still n stations behind this specific station that we should stop by and d miles we have to cover, as shown in the Figure below.
  - Suppose that at the one gas station x' before this station x, the remaining milage p' is greater than the distance to the station x, but we still stop at this station and fill up the tank.
  - Then, after we stop by at the station x', the distance d' we have to cover must be greater than d we mentioned above, which means that the numbers n' of stations we have to stop by must also be grrater than or equal to n.
  - Therefore, if we don't follow the algorithms we come up with above, the total number of stations we have to stop by can only be greater (or the same at least). So our algorithm is the optimal



- **Runtime complexity:**