HW - Week 15

Name: Sahand Sabour **Student ID:** 2020280401

1. Initially, we would sort the points $\{x_1, x_2, ..., x_n\}$ based on their respective positions on the real line, which gives a new set of sorted points $\{x'_1, x'_2, ..., x'_n\}$. Accordingly, we would have

$$x'_1 \leq x'_2 \leq ... \leq x'_n$$

Greedy-choice: the left-most point in the sorted set of points.

Greedy-choice property: There is an optimal solution to this sub-problem that includes the left-most point. Suppose S is an optimal solution to the sub-problem and x_i is the left-most interval in the sub-problem. If x_i is in S, then we are done with the proof. Else, we consider that xi is not in S and x_j is the left-most interval in S. Accordingly, since both xi and xj are the left-most interval in the sub-problem, we can have an optimal solution S' = S - x_j + x_i. Therefore, the greedy choice is proven by the greedy choice property.

The proposed algorithm would first sort the points. Then, we would select the left-most point xi for the sub-problem and create the interval [x_i, x_i+1]. Accordingly, if there's a point at x_i+1, we would delete this point and record the interval [x_i, x_i+1]. Lastly, we would move to the next left-most point and repeat the process until the whole line has been scanned.

2. Initially, we would sort the tasks $\{a_1, a_2, ..., a_n\}$ based on their required times $\{p_1, p_2, ..., p_n\}$, which gives the new set of sorted tasks $\{a'_1, a'_2, ..., a'_n\}$ whose processing times meet the following condition

$$p'_1 \leq p'_2 \leq ... \leq p'_n$$

Greedy-choice: choose the activity with the smallest processing time (the left-most one in the sorted set of tasks).

Greedy-choice property: There is an optimal solution to this sub-problem that includes the task with the smallest processing time. Let a_m be the activity with the smallest processing time and let S be an optimal solution. If S includes a_m, then we are done. If a_m is not in S, there would be a task a_k in S with the smallest processing time. Since both a_m and a_k are in the same sub-problem and it was assumed initially that a_m is the task with the smallest processing time, we would have an optimal solution $S' = S - a_k + a_m$. Therefore, the greedy choice is proven by the greedy choice property.

The proposed algorithm would first sort the tasks based on their running time and continuously choose the left-most task, which is the task with the smallest processing time. Since, at each sub-problem, we are choosing the minimum processing time to be added to the total, the overall average completion time would be minimized. This process can be done in constant time while the sorting process has a runtime of O(nlgn). Hence the running time of the algorithm would be O(nlgn).