H6.

- The assignment is due at Gradescope on Friday, March 3 at 11:59am. Late assignments will not be accepted. Submit early and often.
- You are permitted to study with friends and discuss the problems; however, *you must write up you own solutions, in your own words*. Do not submit anything you cannot explain. If you do collaborate with any of the other students on any problem, please do list all your collaborators in your submission for each problem.
- Finding solutions to homework problems on the web, or by asking students not enrolled in the class is strictly prohibited.
- We require that all homework submissions are prepared in Latex. If you need to draw any diagrams, however, you may draw them with your hand. Please use *a new page to begin each answer.*

PROBLEM 1 *Stabbing intervals*

Let X be a set of n intervals on the real line. A set of points P *stabs* X if every interval in X contains at least one points in P. Given the arrays $X_L[1,\ldots,n]$ and $X_R[1,\ldots,n]$ specifying the left and right end points of the intervals, we want to design a greedy algorithm that computes the smallest set of points stabbing X.

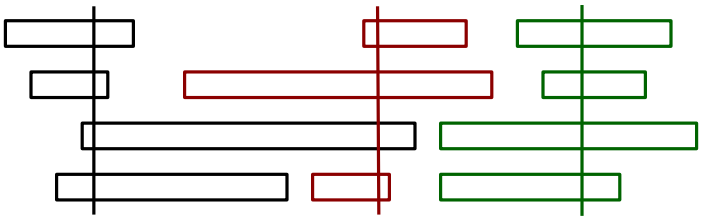

Figure 1: A set of intervals stabbed by 3 points.

(a) Prove that it suffices to consider only points in $X_R[1,\ldots,n]$ as candidates for P. That is, consider any valid solution P. Show that there is a valid solution P' with the same number of points as P and it only uses points in $X_R[1,\ldots,n]$.

Solution:

(b) Give a greedy algorithm for the problem. Show that it is optimal using an exchange argument (consider an optimal solution X and show that one can get a valid solution by exchanging one point in X with the first point your algorithm picks). Hint: review how the proof works for the class scheduling problem.

Solution:

(c) Give pseudocode implementing your algorithm as fast as possible. Analyze the running time. You can make calls to algorithms we learned in the course e.g. Karatsuba, mergesort, etc.

Solution:

PROBLEM 2 *Homework*

College life is hard. You are given n assignments $a_1, \ldots, a_n$ in your courses. Each assignment $a_i = (d_i, t_i)$ has a deadline d_i when it is due and an estimated amount of time it will take to complete, t_i. We assume that all deadlines d_i are distinct. You would like to get the most out of your college education, and so you plan to finish all of your assignments. Let us assume that when you work on one assignment, you give it your full attention and do not work on any other assignment. We also assume that you start to work on your assignments from time 0 and can work without any break until all assignments are done.

In some cases, your outrageous professors demand too much of you. It may not be possible to finish all of your assignments on time given the deadlines $d_1, \ldots, d_n$; indeed, some assignments may have to be turned in late. Your goal as a sincere college student is to minimize the lateness of any assignment. If you start assignment a_i at time s_i, you will finish at time $f_i = s_i + t_i$. The lateness value—denoted ℓ_i—for a_i is the value

$$\ell_i = \begin{cases} f_i - d_i & \text{if } f_i > d_i \\ 0 & \text{otherwise} \end{cases}$$

Devise a polynomial-time algorithm that computes a schedule O that specifies the order in which you complete assignments which minimizes the maximum ℓ_i for all assignments, i.e.

$$\min_O \max_i \ell_i$$

In other words, you do not want to turn in *any* assignment *too* late, so you minimize the lateness of the latest assignment you turn in.

(a) Prove that there is an optimal solution with no break time. That is, consider an arbitrary solution O and show that there is a valid solution O' that is at least as good and has no break time.

Solution:

(b) Give a greedy algorithm and prove that your algorithm is optimal using an exchange argument. That is, find a property that an optimal ordering should satisfy. Consider an ordering O violating the property and show that you can swap two consecutive assignments i followed by j in O that violate the property and produce a new ordering O'. Let s_i be the start time of assignment i in O. Calculate new finish times of i, j and the new lateness based on s_i. Show that the new ordering O' is no worse than O. Hint: review the problem of storing files on a tape.

Solution:

(c) Give pseudocode implementing your algorithm as fast as possible. Analyze the running time. You can make calls to algorithms we learned in the course e.g. Karatsuba, mergesort, etc.

Solution: