Collaboration Policy: You are encouraged to collaborate with up to 4 other students, but all work submitted must be your own *independently* written solution. List the computing ids of all of your collaborators in the `collabs` command at the top of the tex file. Do not share written notes, documents (including Google docs, Overleaf docs, discussion notes, PDFs), or code. Do not seek published or online solutions for any assignments. If you use any published or online resources (which may not include solutions) when completing this assignment, be sure to cite them. Do not submit a solution that you are unable to explain orally to a member of the course staff. Any solutions that share similar text/code will be considered in breach of this policy. Please refer to the syllabus for a complete description of the collaboration policy.

Collaborators: list your collaborators

Sources: list your sources

PROBLEM 1 *Divide and Conquer with MSTs*

Professors Pettit and Hott have been discussing minimum spanning trees and attempting to create new algorithms to compute them. Professor Pettit claims to have created a divide and conquer algorithm as follows:

> Given a graph $G = (V, E)$, partition the set of vertices V into two sets V_L and V_R such that $|V_L|$ and $|V_R|$ differ by at most 1. Let E_L be the set of edges that are incident only on vertices in V_L and E_R be the set of edges incident only on vertices in V_R. Recursively compute the minimum spanning tree on each of the two subgraphs $G_L = (V_L, E_L)$ and $G_R = (V_R, E_R)$, then select the minimum weight edge $e \in E$ that crosses the cut (V_L, V_R). Use e to combine the two minimum spanning trees into a single minimum spanning tree for G.

Help us to evaluate his algorithm. Either prove that it correctly computes a minimum spanning tree of graph G or provide a counterexample for which the algorithm fails.

Solution:

PROBLEM 2 *As You Wish*

Buttercup has given Westley a set of n tasks $t_1, \ldots, t_n$ to complete on the farm. Each task $t_i = (d_i, w_i)$ is associated with a deadline d_i and an estimated amount of time w_i needed to complete the task. To express his undying love to Buttercup, Westley strives to complete all the assigned tasks as early as possible. However, some deadlines might be a bit too demanding, so it may not be possible for him to finish a task by its deadline; some tasks may need extra time and therefore will be completed late. Your goal (inconceivable!) is to help Westley minimize the deadline overruns of any task; if he starts task t_i at time s_i, he will finish at time $f_i = s_i + w_i$. The deadline overrun (or lateness) of tasks—denoted L_i—for t_i is the value

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

Give a polynomial-time algorithm that computes the optimal order T for Westley to complete Buttercup's tasks so as to minimize the maximum L_i across all tasks. That is, your algorithm should compute T that minimizes

$$\min_T \max_{i=1,\ldots,n} L_i$$

In other words, you do not want Westley to complete *any* task *too* late, so you minimize the deadline overrun of the task completed that is most past its deadline. Describe how you know your algorithm produces an optimal schedule. *Hint: you may wish to construct an exchange argument here.* Additionally, analyze your algorithm's running time.

Solution:

PROBLEM 3 *Unit Intervals*

You are given a set of points $P = \{p_1, p_2, ..., p_n\}$ on the real line (you may assume these are given to you in sorted order). Describe an algorithm that determines the smallest set of unit-length closed intervals that contains all of the given points. For example, the points $\{0.9, 1.2, 1.3, 2.1, 3.0\}$ can be covered by $[0.7, 1.7]$ and $[2.0, 3.0]$. State the runtime of your algorithm.

Solution: