Computer Science C73 — September 15, 2021
Scarborough Campus — University of Toronto

Homework Assignment #2

(worth 6% of the course grade)
Due: September 22, 2021, by 11:59 pm

• ***You must submit your assignment through the Crowdmark system.*** *You will receive by email an invitation through which you can submit your work in the form of separate PDF docucments with your answers to each question of the assignment. To work with a partner, you and your partner must form a group on Crowdmark. Crowdmark does not enforce a limit on the size of groups.* ***The course policy that limits the size of each group to at most two remains in effect:*** *submissions by groups of more than two persons will not be graded.*
• *It is your responsibility to ensure that the PDF files you submit are legible. To this end, I encourage you to learn and use the LaTex typesetting system, which is designed to produce high-quality documents that contain mathematical notation. You are not required to produce the PDF files you submit using LaTex; you may produce it any way you wish, as long as the resulting document is legible.*
• *By virtue of submitting this assignment you (and your partner, if you have one) acknowledge that you are aware of the policy on homework collaboration for this course.*[a]
• *For any question, you may use data structures and algorithms previously described in class, or in prerequisites of this course, without describing them. You may also use any result that we covered in class, or is in the assigned sections of the official course textbooks, by referring to it.*
• *Unless we explicitly state otherwise, you may describe algorithms in high-level pseudocode or point-form English, whichever leads to a clearer and simpler description.*
• *Unless we explicitly state otherwise, you should justify your answers. Your paper will be graded based on the correctness and efficiency of your answers, and the clarity, precision, and conciseness of your presentation.*

[a] "In each homework assignment you may collaborate with at most one other student who is currently taking CSCC73. If you collaborate with another student on an assignment, you and your partner must submit only one copy of your solution, with both of your names. The solution will be graded in the usual way and both partners will receive the same mark. Collaboration involving more than two students is not allowed. **For help with your homework you may consult only the instructor, TAs, your homework partner (if you have one), your textbook, and your class notes. You may not consult any other source.**"

Question 1. (20 marks) A catering service receives n orders, each requiring ***exactly one time unit*** to complete. There is only one cook, so these orders must be completed one at a time; the cook starts work at time 0.

Order i, $1 \leq i \leq n$, has a ***deadline*** d_i (in time units) by which it must be completed, and a ***profit*** $p_i > 0$ that the catering service receives if the order is completed by the deadline; if the order is completed after the deadline there is no profit made and therefore it is not done at all. Note that several orders may have the same deadline (e.g., around lunch time). The catering service must decide on a schedule in which to do a subset of the orders, each completed by its deadline.

Describe an $O(n^2)$ greedy algorithm that takes as input the set of orders with their deadlines and profits $J = \{(i, d_i, p_i) : \ 1 \leq i \leq n\}$ and outputs a schedule of maximum profit. A schedule is a function S that maps each order i to a time slot j, where j is a positive integer, meaning that order i will be done during the time interval $(j-1, j)$; or to the time slot $+\infty$, meaning that order i will not be done. The schedule S cannot map two different orders to the same time slot (because there is only one cook who completes the orders one at a time); and if it maps order i to time slot $j \neq 0$, then $j \leq d_i$ (since every order that is done must be finished by its deadline). The profit of schedule S is the sum of the orders that are done, i.e., $\sum_{1 \leq i \leq n \ \& \ S(i) \neq \infty} p_i$.

Prove that your algorithm is correct and analyze its running time. (It can be done in $O(n \log n)$ time, but $\Theta(n^2)$ is good enough for the purposes of this question.)
Hint. No job can be scheduled after the maximum deadline. What job should be scheduled in that slot?

Question 2. (15 marks) A video production facility has n videos to process. The processing of each video i, $1 \leq i \leq n$, is done in two phases. In the first phase, the facility's editor edits the video. After the first phase of video i is finished, the edited video is processed by computer. The first phase of video i takes $f_i \geq 0$ units of time and the second phase takes $s_i \geq 0$ units of time.

The facility has a single video editor to do the first phase but it has n computers to do the second phase. So, the first phase must be done sequentially by the single editor, while the second phase can be done in parallel by the computers: As soon as the editor is finished editing a video, she feeds it into one of the free computers for the second phase, even as other computers may still be processing the videos whose first phase she completed earlier. The completion time of a video is the time by which its second phase is finished. The manager of the facility wants to find a schedule that ***minimizes the maximum completion time over all*** n ***videos*** — that is, the earliest time by which both phases of all videos have been been completed.

Describe an efficient greedy algorithm that takes as input the set of videos and the length of their first and second phases $V = \{(i, f_i, s_i) : \ 1 \leq i \leq n\}$ and outputs an optimal schedule. In this case, a schedule is simply a permutation of the integers from 1 to n, indicating the order in which the editor processes the videos. This determines the time by which each video finishes, and therefore the maximum completion time over all videos, which is the quantity we want to minimize.

Prove that your algorithm is correct and analyze its running time.

Question 3. (10 marks) We are given a directed graph $G = (V, E)$, nodes $s, t \in V$, and an edge weight function $\mathbf{wt}$ such that $\mathbf{wt}(e) \geq 0$ for every $e \in E$. In addition, we are given a set E' of ***potential edges***, i.e., pairs of nodes that are not edges in E; and for each $e' \in E'$ we are given a non-negative weight $\mathbf{wt}(e')$.

We want to choose a ***single*** potential edge in E' that, if added to the graph, reduces as much as possible the weight of a shortest $s \rightarrow t$ path. (If there is no potential edge whose addition to G reduces the weight of the shortest $s \rightarrow t$ path, we can choose any potential edge.)

Describe an algorithm that solves this problem in $O\big(k + (m+n)\log n\big)$ time, where $n = |V|$, $m = |E|$, and $k = |E'|$. The graph is given in adjacency list form, and the set of potential edges (and their associated weights) is given as a list. Your algorithm ***must*** use an algorithm we discussed in this course.

Describe your algorithm in ***clear and concise point-form English***. Do not write pseudocode. Do not justify the correctness of your algorithm. Explain why your algorithm meets the required running time.