HW 4

Due on **Wednesday, September 20, 2023 at 9pm**

CS 473: Algorithms, Fall 2023 Version: **1.0**

Submissions instructions: As in previous homework.

10 (100 PTS.) Intervals, intervals everywhere.

Two intervals $[x, y], [z, w]$ are ***compatible***, if $[x, y] \subseteq [z, w]$ or $[z, w] \subseteq [x, y]$. Given a set $\mathcal{I}$ of n intervals (say, with all distinct endpoints), describe an algorithm, *as fast as possible*, to compute the largest compatible subset $\mathcal{S} \subseteq \mathcal{I}$. A set of intervals $\mathcal{S}$ is compatible, if any two intervals in it are compatible.

Here, running time matters – an algorithm with running time $O(n^2)$ is worth at most 25 points.

(Hint: Reduce this to a problem seen in class.)

11 (100 PTS.) Boxes, boxes everywhere.

A ***box*** in three dimensions is an axis aligned product of three closed intervals. Formally, such a box $B = [x_1, x_2] \times [y_1, y_2] \times [z_1, z_2]$. Two boxes B, B' are ***compatible***, if $B \subseteq B'$ or $B' \subseteq B$. Given a set $\mathcal{B}$ of n boxes (say, with all the associated values being distinct), describe an algorithm, *as fast as possible*, to compute the largest compatible subset $\mathcal{S} \subseteq \mathcal{B}$. A set of boxes $\mathcal{S}$ is compatible, if any two boxes in it are compatible.

Here, the expected running time is quadratic. (A faster algorithm is possible, but requires data-structures and tools not seen in class.)

(Hint: Construct the appropriate DAG.)

12 (100 PTS.) Synchronized motion planning I.

You are given k robots, and their respective paths $\pi_1, \ldots, \pi_k$. The ith robot can be in any of the n locations specified by its path π_i (i.e., the jth location of robot i is a point $\pi_i[j]$ in $\mathbb{R}^d$ where d is a constant).

Given the locations of the k robots, say $q_i = \pi_i[R_i]$, for $i = 1, \ldots, k$, where $R_i \in [\![n]\!]$. The ***energy*** of this configuration (i.e., $[R_1, \ldots, R_k]$), which is some real number, denoted by $\mathcal{E}(q_1, \ldots, q_k)$, and it can be computed in constant time by a function provided to you.

A configuration of the robots is in ***equilibrium*** if no single robot can move to an adjacent location and decrease the energy (importantly, we allow only a single robot to move at a time). Show an algorithm, as fast as possible, that computes a configuration that is an equilibrium.

Hint: Solve first for $k = 2$, then $k = 3$, and then derive the general algorithm. The running time of your algorithm must be strictly better than $O(n^k)$.