CS 331: Algorithms and Complexity
Homework I

Trung Dang Nathan Mardanov Kevin Tian

Due date: September 10, 2025, end of day (11:59 PM), uploaded to Canvas.

Late policy: 15% off if submitted late, and 15% off for every further 24 hours before submission.

Please list all collaborators on the first page of your solutions.

When runtimes are unspecified, slower runtimes than the intended solution receive partial credit.

1 Problem 1

There is a magical land called Intervalia where n creatures live, with a unique social structure. The i^{th} creature has a *designated interval* $[\ell_i, r_i] \in \mathbb{Z}^2$, where $\ell_i < r_i$. Intervalia has a law that all of the ℓ_i and r_i must always be distinct. Whenever the i^{th} creature and j^{th} creature meet, how they interact depends on their designated intervals.

- *Non-overlap.* If $[\ell_i, r_i] \cap [\ell_j, r_j] = \emptyset$, i.e., the two intervals do not overlap, then the creatures must immediately part ways (to not intersect).
- *Containment.* If $[\ell_i, r_i] \subset [\ell_j, r_j]$ or $[\ell_i, r_i] \supset [\ell_j, r_j]$, then whichever creature has their interval contained in the other's, must bow as a sign of respect.
- *Partial overlap.* Neither of the other two cases holds. There is only partial overlap between the intervals, and the creatures shake hands.

You want to understand some basic statistics on how creatures in Intervalia feel about each other: the number of pairs that interact in each of the three possible ways.

Formally, let L be an Array containing n tuples of integers, and denote its i^{th} entry by $L[i] = (\ell_i, r_i)$, where $\ell_i < r_i$. Assume that no two integers in L are equal.

For each of the $\binom{n}{2} = \frac{n(n-1)}{2}$ pairs (i, j) with $1 \le i < j \le n$, exactly one of the following holds for the intervals $[\ell_i, r_i]$ and $[\ell_j, r_j]$: non-overlap, containment, or partial overlap. Give an algorithm that, on input L, counts how many pairs (i, j) satisfy each in $O(n \log(n))$ time.

Note that you only need to solve two of these three counting problems. Correctly answering one is worth **(10 points)**, and correctly answering another is worth the other **(10 points)**.

2 Problem 2

Let L be an Array containing n real numbers.

(i) **(5 points)** Assume $L[i] > 0$ for all $i \in [n]$. Give an algorithm that, on input L, prints indices (i, j) with $1 \leq i \leq j \leq n$, maximizing the product $\prod_{k=i}^{j} L[k]$.

(ii) **(15 points)** Assume nothing other than $L[i] \in \mathbb{R}$ for all $i \in [n]$. Give an algorithm that, on input L, prints indices (i, j) with $1 \leq i \leq j \leq n$, maximizing the product $\prod_{k=i}^{j} L[k]$. Completing this part of the problem gives credit for the other part.

3 Problem 3

Let $n = 2^k$ for $k \in \mathbb{Z}_{\geq 0}$ be a power of two. We recursively define $\mathbf{H}_k$, the $n \times n$ *Walsh-Hadamard transform* (WHT) matrix, as follows, where the base case is a $2^0 \times 2^0$ matrix (i.e., a number):

$$\mathbf{H}_0 = 1, \quad \mathbf{H}_k := \begin{pmatrix} \mathbf{H}_{k-1} & \mathbf{H}_{k-1} \\ \mathbf{H}_{k-1} & -\mathbf{H}_{k-1} \end{pmatrix}.$$

(i) **(5 points)** What are $\mathbf{H}_1 \in \mathbb{R}^{2\times 2}$, $\mathbf{H}_2 \in \mathbb{R}^{4\times 4}$, and $\mathbf{H}_3 \in \mathbb{R}^{8\times 8}$?

(ii) **(5 points)** Prove that for all $k \in \mathbb{N}$, $\mathbf{H}_k^2 = C_k \mathbf{I}_n$ for some $C_k > 0$. What is C_k?

(iii) **(5 points)** Give an algorithm $\mathsf{WHT}_n(\mathbf{v})$ that, on input $\mathbf{v} \in \mathbb{R}^n$, outputs $\mathbf{H}_k\mathbf{v}$.

(iv) **(5 points)** Give an algorithm $\mathsf{WHTInv}_n(\mathbf{v})$ that, on input $\mathbf{v} \in \mathbb{R}^n$, outputs $\mathbf{H}_k^{-1}\mathbf{v}$.

4 Problem 4

(i) (**10 points**) In the country of Numerica, there are n citizens, named $1, 2, \ldots, n$. In Numerica's elections, each citizen votes for one name in $[n]$. Numerica has a law such that n is always odd, so ties are impossible. A leader is chosen when someone receives more than $\frac{n}{2}$ votes. You are the vote-counter, and you want to check if a leader was elected.

Formally, given an Array L of length n with entries in $[n]$ (the votes), where n is odd, output a leader if one exists, or correctly report that none exists. Your algorithm must run in $O(n)$ time and should not rely on hashing (which only gives randomized guarantees).

(ii) (**10 points**) In the country of Linexico, the n citizens live in $\mathbb{R}^2$, with the i^{th} citizen's home at (x_i, y_i). On any day where more than half the homes lie on the same line, every citizen must take part in a line dance at sundown. Linexico has a law such that n is always odd. You are Linexico's choreographer, and want to check if a line dance will happen today.

Formally, let L be a length-n Array instance containing tuples in $\mathbb{R}^2$ (the homes), where n is odd. We say that a line, given as a slope-intercept pair $(m, b) \in \mathbb{R}^2$, is a *dance line* if at least half of the $L[i] = (x_i, y_i)$ lie on the line, i.e., $y_i = mx_i + b$. Give an $O(n \log(n))$-time algorithm that either returns a dance line of L, or correctly concludes that none exists.

5 Problem 5

(**20 points**) Complete the assignment at this link. This link is only accessible on your UT email.