

COMP 550 – Algorithms and Analysis

Spring 2023-Assignment04

Issue Date: March 7, 2023

Due Date: Monday, March 21, 11:55 pm

Marks: 5

Student Name: _____

Student PID: _____

Submission:

- You must upload your submission(s) before the deadline in Gradescope.
- Please ensure that your answers are within the given space allocated after the question.

Rules for ALL HWs:

You are encouraged to discuss the homework assignments and study together in groups, but when it comes to formulating/writing solutions **you must work alone and independently**. If required, you should be able to explain your answer clearly to TAs/LAs. Copying homework solutions from another student, from the Internet, solution sets of friends, or other sources will be considered cheating and treated accordingly.

1 Randomized Analysis

The algorithm given below is a method for choosing an item uniformly at random from an arbitrarily long stream of data. (You may consider it as an array)

Algorithm 1 GetSample(Stream S)

Require: A stream of data S

$l \leftarrow 1$

while S is not done **do**

$x \leftarrow$ next Item in S

$l \leftarrow l + 1$

if Random(l) = 1 **then**


$sample \leftarrow x$

end if

end while

return $sample$

▷ See Random function below

▷ **Line** 

Algorithm 2 Random(int l)

Require: $l \in \mathbb{N}$

Uniformly return an integer k with equal probability where $1 \leq k < l$

At the end of the algorithm, the variable l stores the length of the input stream S; this number is not known to the algorithm in advance. If S is empty, the output of the algorithm is (correctly!) undefined.

In the following, consider an arbitrary non-empty input stream S , and let n denote the (unknown) length of S.

(You may want to watch the recitation 2 recording as we covered some hints for this question)

1. Prove that the item returned by `GetSample(S)` is chosen uniformly at random from S . (Hint: Construct the model first, then try to prove by induction)

2. What is the *exact* expected number of times that `GetSample(S)` executes line $(\odot \circ \odot)$?

3. What is the exact expected value of l when `GetSample(S)` executes line $(\odot \circ \odot)$ for the last time?(Hint: Recall question 1)

4. What is the exact expected value of l when either $\text{GetSample}(S)$ executes line $(\odot \circ \odot)$ for the second time (or the algorithm ends, whichever happens first)? (Hint: Recall question2)

2 Programming Question

This programming assignment is a bit more challenging than Assignment #2, and is intended to utilize the divide and conquer paradigm. This document includes some leading questions that will guide you to the final solution. For the submission of the programming part, please go to the GitHub URL for the assignment template.

2.1 Description of the problem

Alice has two boxes of pencils, each containing n pencils, and each pencil has a height. The pencils in each box are arranged in a single row, and the pencils are of different heights. The distance between the two rows of pencils is defined as

$$\sum (a_i - b_i)^2$$

where a_i denotes the height of the i th pencil in the first row and b_i denotes the height of the i th pencil in the second row.

The positions of two pencils within a row can be swapped, and you can minimize the distance between the two rows by swapping them. What is the minimum number of swaps required to obtain this minimum distance?

For example, Alice has two boxes of pencils with height $\{2,3,1,4\}$ and $\{3,2,1,4\}$. It is clear to minimize the distance, we could swap the first two pencils in the first box or the first two pencils in the second box to make the distance 0. Therefore, the minimum number of swaps would be 1.

2.2 Analysis of the problem

2.2.1 The minimum distance

As the problem above defines, the minimum distance is $\sum (a_i - b_i)^2$. Let's expand the equation which we have:

$$\sum (a_i - b_i)^2 = \sum (a_i)^2 + \sum (b_i)^2 - \sum (2a_i b_i)$$

It is easy to conclude that regardless of how we swap the pencil within each box, the $\sum (a_i)^2 + \sum (b_i)^2$ will always be a constant. Therefore, minimizing the distance is equivalent to:

$$\text{maximize } 2 \sum (a_i b_i)$$

2.2.2 Transformation of array

We could easily prove that , $\sum_{i=1}^n a_i b_i \geq \sum_{i=1}^n a_i b_{j_i}$, Where j_1, j_2, \dots, j_n is a random permutation of $1, 2, \dots, n$, is always true. Therefore, to solve the problem, we could design a map $f : \mathbb{R}^n \rightarrow \mathbb{R}^n$ such that it will let the index of i th largest element to i . For example, consider $\vec{A} = [1, 199, 3, 4, 7]^T$. Then the mapping of vector A would be $f(\vec{A}) = [0, 2, 3, 4, 1]^T$.

Implementing this method will be the first task of the programming assignment.

2.2.3 Simplification of the Original Problem

Now let's consider the \vec{A} as the mapping of the first input array, and \vec{B} as the mapping of the second input array. It is easy to see that if we just sort \vec{A} and \vec{B} we will have the minimum distance, and the number of

swaps during the sort could be considered as the upper bound for total # of swapping we will have for arbitrary arrays A and B.

- (a) What would be the upper bound for the total number of swaps for array \bar{A} and \bar{B} ? (Hint: Recall the definition of inversion from Assignment 2)

The above question asks us to output the least number of swaps. The hard part would be that in order to swap the minimum # times, we might need to swap both arrays \bar{A} and \bar{B} . Therefore, if we could try to use arrays \bar{A} and \bar{B} to build an array C, and we just swap array C, the problem would be made much easier.

2.2.4 Building Array C

Let's think again about the goal we would like to achieve for \bar{A} and \bar{B} . We want to find the minimum number of swaps to make $\bar{A}=\bar{B}$. Or in other words, we are trying to sort array \bar{A} by using the order of array \bar{B} (Or we want to sort array \bar{B} by using the order of array \bar{A}). Let's define array C

as follows:

$$C[\bar{A}[i]] = \bar{B}[i]$$

- (a) Based on the above definition, if $\bar{A} = \bar{B}$ What would $C[\bar{A}[i]]$ be?
What would $C[i]$ be? (Hint: Try to come up with some examples)

Think again. How could array C relate to the solution of the problem? And this would be the key to the whole question!

2.3 Hints for programming

- (a) The expected complexity of your whole program is $\Theta(n \lg n)$
- (b) Review the definition of inversions and think about how you could have better implement the algorithm in assignment 2. You should be able to calculate this in $\Theta(n \lg n)$