COMP 418/518: Homework 1

[Weight of homework: 15% of the final grade]

Authors (fill in your names and NetIDs here)

Due on February 4, 2026
(released on January 21, 2026)

Instructions: For this homework, you should work in groups of 3–4 people (i.e., the groups that have been declared and accepted). Only one member of the group should submit on Gradescope. All group members must be included in the submission on the Gradescope system (otherwise no credit will be given). Also specify in your submission file all the members of your group (name and NetID) using the `\authors` macro. The submission must be typed in LaTeX using the provided template. The use of AI systems (including, but not limited to, ChatGPT, Gemini, Copilot, and Claude) is entirely prohibited. No discussion or collaboration is allowed outside of the group. Late submissions are not accepted.

Note: We will probably request that you upload the LaTeX source for your submission. More information regarding how to make the source submission will be made available in a few days.

Grading: The homework will be graded based on correctness, completeness, and the quality of the provided solutions.

Problem 1 [20 points]

The code shown below describes the offline computation of the standard deviation over the data values (floating-point numbers) placed in an array $X[0..n-1]$:

```
Function StdDev_Offline($X[0..n-1]$):
    $s \leftarrow 0$
    for $i = 0, \ldots, n-1$ do  $s \leftarrow s + X[i]$
    $mean \leftarrow s/n$
    $s \leftarrow 0$
    for $i = 0, \ldots, n-1$ do  $s \leftarrow s + (X[i] - mean)^2$
    return $\sqrt{s/n}$
```

Consider the streaming problem StdDev for calculating the running standard deviation over a stream of floating-point numbers. Describe an efficient streaming algorithm for StdDev. Provide pseudocode for your streaming algorithm. Explain why it is correct and discuss its space and time-per-item complexity.

Note: You can assume that each floating-point number fits in one memory location and that arithmetic operations on them take $O(1)$ time.

Solution

Fill me in.

Problem 2 [20 points]

The code shown below describes an offline (non-streaming) aggregation over the data values (natural numbers) placed in an array $X[0..n-1]$:

```
Function Unknown_Offline($X[0..n-1]$):
    $y \leftarrow 0$
    for $i = 0, \ldots, n-1$ do
        if $X[i] > y$ then  $y \leftarrow X[i]$
    $z \leftarrow 0$
    for $i = 0, \ldots, n-1$ do
        if $X[i] \neq y$ and $X[i] > z$ then  $z \leftarrow X[i]$
    return $z$
```

The algorithm `Unknown_Offline` computes an aggregation function $f : \mathbb{N}^+ \to \mathbb{N}$, where $\mathbb{N} = \{0, 1, 2, \ldots\}$ is the set of natural numbers.

1. Given an informal explanation in English of what `Unknown_Offline` computes.
2. Give pseudocode for a streaming algorithm that performs the running aggregation f. That is, if the current input history is $x_1 x_2 \ldots x_n$, then the current output item should be $f(x_1 x_2 \ldots x_n)$. What is the space and time-per-item complexity of your algorithm?

Solution

Fill me in.

Problem 3 [30 points]

The code shown below describes an offline (non-streaming) aggregation over the data values (natural numbers) placed in an array $X[0..n-1]$ of size $n \geq 1$:

```
Function Choose_Offline($X[0..n-1]$):
    $(y, z) \leftarrow (X[0], 0)$
    for $j = 0, \ldots, n-1$ do
        if $X[j] = X[0]$ then $z \leftarrow z + 1$
    for $i = 1, \ldots, n-1$ do
        $w \leftarrow 0$
        for $j = 0, \ldots, n-1$ do
            if $X[j] = X[i]$ then $w \leftarrow w + 1$
        if $w > z$ then
            $(y, z) \leftarrow (X[i], w)$
        else if $w = z$ and $X[i] > y$ then
            $y \leftarrow X[i]$
    return $y$
```

The algorithm `Choose_Offline` computes an aggregation function $g : \mathbb{N}^+ \to \mathbb{N}$, where $\mathbb{N} = \{0, 1, 2, \ldots\}$ is the set of natural numbers.

1. Given an informal explanation in English of what `Choose_Offline` computes.

2. ***Assumption***: We assume that the input values are drawn from a finite subset $[R] = \{0, 1, \ldots, R-1\} \subseteq \mathbb{N}$, where $R \geq 2$ is a small ***constant***.

   Give pseudocode for a streaming algorithm that performs the running aggregation g (under the aforementioned assumption). That is, if the current input history is $x_1 x_2 \ldots x_n \in [R]^+$, then the current output item should be $g(x_1 x_2 \ldots x_n) \in [R]$. What is the space and time-per-item complexity of your algorithm?

Solution

Fill me in.

Problem 4 [25 points]

Consider the streaming problem MEDIAN(R) for calculating the running median over a stream of integers from the set $[R] = \{0, 1, \ldots, R-1\}$, where $R \geq 2$ is a constant. Describe a streaming algorithm for MEDIAN(R) that uses $O(\log n)$ space (in bits), where n is the length of the stream. You should provide pseudocode for your streaming algorithm.

Solution

Fill me in.

Problem 5 [40 points]

Let $R \geq 2$ be an integer (constant). Show that the space complexity of every algorithm that solves $\textsc{Median}(R)$ is $\Omega(\log n)$, where space is measured in bits.

Solution

Fill me in.

Problem 6 [15 points]

What is the space complexity of the algorithm you gave for Problem 4, as a function of both n and R? How much time does the algorithm need for every step? What data structure would you use to improve the time-per-item requirement of the algorithm? In order to answer this last question, think of the case where R is very large.

Solution

Fill me in.