Your name
Your student number

CS 466/666: ALGORITHM DESIGN AND ANALYSIS
PROBLEM SET 2

DUE ON: OCTOBER 4, 2017

ACKNOWLEDGEMENTS.

Acknowledge all collaborators with whom you discussed any of the problems in this problem set and all external sources that you may have used in the completion of the assignment. See the website for the full homework policy.

1. PROBLEM 1: MIDDLE OF THE ROAD

For this question, we will examine a question that comes up in the study of quicksort and a number of other applications as well. For some integers \(x_1, \ldots, x_n \in \mathbb{Z}\), we define the rank of an integer \(y \in \mathbb{Z}\) to be the number of elements in the list \(x_1, \ldots, x_n\) that are no larger than \(y\). Formally,

\[
\text{rank}(y) = |\{i \in [n] : x_i \leq y\}|
\]

Let \(X\) be a random variable obtained by drawing \(i \in [n]\) uniformly at random and setting \(X = x_i\).

(a) Establish the expected value of the rank of \(X\).

**Lemma 1.1.** The rank of \(X\) has expected value \(E[\text{rank}(X)] = (\text{enter the correct value here})\).

**Proof.** Enter your answer here. \(\square\)

(b) Establish the variance of the rank of \(X\).

**Lemma 1.2.** The rank of \(X\) has variance \(\text{Var}[\text{rank}(X)] = (\text{enter the correct value here})\).

**Proof.** Enter your answer here. \(\square\)

(c) Show that the rank of \(X\) is neither very small nor very large except with small probability.

**Lemma 1.3.** Let \(E\) denote the event that the rank of \(X\) satisfies \(\text{rank}(X) \leq \frac{n}{10}\) or \(\text{rank}(X) \geq \frac{9n}{10}\). Then \(\Pr[E] \leq (\text{enter the correct value here})\).

**Proof.** Enter your answer here. \(\square\)

(d) Let \(X_1, \ldots, X_t\) be \(t\) copies of \(X\) drawn independently at random for some odd \(t \geq 1\). Show that the probability that the median of these values has very small or very large rank is exponentially small in \(t\).

**Lemma 1.4.** Define \(p \geq \Pr[E]\) where \(E\) is the event defined in Lemma 1.3. Then for any odd \(t \geq 1\), the rank of the variable \(Y = \text{median}(X_1, \ldots, X_t)\) satisfies

\[
\Pr \left[ \text{rank}(Y) \leq \frac{n}{10} \text{ or } \text{rank}(Y) \geq \frac{9n}{10} \right] \leq \left( \frac{t}{\lceil t/2 \rceil} \right) p^{t/2} \leq 2^t p^{t/2} = (4p)^{t/2}.
\]

**Proof.** Enter your answer here. \(\square\)

**Hint.** An extremely useful tool for this and many other problems is the union bound, which states that for any events \(E_1, E_2, \ldots, E_m\) we have \(\Pr[E_1 \lor E_2 \lor \cdots \lor E_m] \leq \sum_{i \leq m} \Pr[E_i]\).
2. Problem 2: Counting with accuracy

(a) Prove the following fundamental fact about averages of samples.

**Lemma 2.1.** Let $X_1, \ldots, X_t$ be independent random variables each with expected value $E[X_i] = \mu$ and variance $\text{Var}[X_i] = \sigma^2$. Then the variable $X = \frac{X_1 + X_2 + \cdots + X_t}{t}$ satisfies

\[
E[X] = \mu \quad \text{and} \quad \text{Var}[X] = \frac{\sigma^2}{t}.
\]

*Proof.* Enter your answer here. □

(b) Determine the variance of the Morris counter seen in class.

**Lemma 2.2.** The random variable $Z = 2^C_k - 1$ corresponding to the value returned by the Morris counter when the stream contains $k$ ones has variance $\text{Var}[Z] = \binom{k}{2}$.

*Proof.* Enter your answer here. □

*Hint.* You may want to use a proof by induction to show that $E[(2^C_k)^2] = \frac{3}{2}k^2 + \frac{3}{2}k + 1$.

(c) Use the above two results to show that we can design approximate counters with arbitrary accuracy by running some number $t \geq 1$ of copies of the Morris counter in parallel and averaging their estimates.

**Theorem 2.3.** For any $\epsilon > 0$ and $n \geq 1$, there is a value $t \geq 1$ for which the algorithm $A$ that runs $t$ copies of the Morris counter is an $\epsilon$-approximate counter for inputs $x_1, \ldots, x_n \in \{0, 1\}$ and has memory complexity $O(t \log \log n)$.

*Proof.* Enter your answer here. □

*Remark.* Your proof should specify exactly what the value of $t$ should be (as a function of $n$ and/or $\epsilon$). As an aside, is the algorithm $A$ you obtain this way a very efficient approximate counter?
3. PROBLEM 3: COUNTING WITH CONFIDENCE

(a) Show that when $E_1, \ldots, E_t$ are independent events that each occur with small probability, the probability that more than half of the events occur is exponentially small in $t$.

**Lemma 3.1.** Let $E_1, E_2, \ldots, E_t$ be independent events that each occur with probability $\Pr[E_i] = \mu \leq \frac{1}{3}$. Then

$$\Pr \left[ \left| \left\{ i \leq t : E_i \text{ occurs} \right\} \right| > \frac{t}{2} \right] \leq \left( \frac{t}{\lfloor t/2 \rfloor} \right) \mu^{t/2} \leq (4\mu)^{t/2}.$$ 

**Proof.** Enter your answer here. □

(b) Using the lemma above, show that the probability of error of approximate counters can be reduced to any $\delta > 0$ with only a small multiplicative factor increase in the memory cost.

**Theorem 3.2.** Let $A$ be an $\epsilon$-approximate counter with memory complexity $m$. Then for any $\delta > 0$, there is a randomized streaming algorithm $A'$ with memory complexity $O(m \log \frac{1}{\delta})$ which outputs an approximate count $\tilde{c}$ that satisfies $(1 - \epsilon)k \leq \tilde{c} \leq (1 + \epsilon)k$ with probability at least $1 - \delta$ when $k = \sum_i x_i$ is the number of ones in the stream.

**Proof.** Enter your answer here. □
4. Problem 4: Impossibility

I have argued in class that the only way to beat the naïve counter’s memory requirements in streaming algorithms is to consider randomized algorithms that are allowed to err with some probability. In this question, you will prove this assertion.

(a) Prove that every deterministic (exact) counter has memory complexity $\Omega(n)$ for streams of length $n$.

**Theorem 4.1.** Let $A$ be a deterministic streaming algorithm that outputs the number of 1s observed in the stream $x_1, \ldots, x_n \in \{0, 1\}$. Then $A$ must have at least $\lceil \log n \rceil$ bits of memory.

**Proof.** Enter your answer here. □

*Hint.* The pigeonhole principle, which says that if you put $m$ pigeons into $\ell < m$ boxes then you must end up with a box that contains at least 2 pigeons, may be useful in your proof.

(b) Define the worst-case memory complexity of a randomized algorithm to be the maximum number of bits of memory that the algorithm requires over all inputs and all possible values of the random bits that it uses. Use the result from part (a) to show that randomized algorithms which output the correct count all the time also have large memory complexity.

**Theorem 4.2.** Let $A$ be a randomized streaming algorithm that always correctly outputs the number of 1s observed in the stream $x_1, \ldots, x_n \in \{0, 1\}$. Then $A$ has worst-case memory complexity at least $\lceil \log n \rceil$.

**Proof.** Enter your answer here. □

*Hint.* Can you show that if $A$ has memory complexity $m$, then there is also a deterministic streaming algorithm for counting that also requires only $m$ bits of memory?

(c) **Bonus.** Extend the results above to show that for any $\epsilon < 1$, every $\epsilon$-approximate counter which always outputs an estimate in the range $(1 - \epsilon)k \leq \hat{c} \leq (1 + \epsilon)k$ also has memory complexity $\Omega(\log n)$. 