CS 473 ✧ Fall 2024

Homework 7

Due **Wednesday, October 30**, 2024 at 9pm Central Time

This is the last homework before Midterm 2.

---

1. Suppose we want to build a sensor that counts passing cars at a freeway interchange, and reports the total at the end of each day. The obvious algorithm initializes a counter to zero at the start of each day, increments the counter each time a car passes the sensor, and finally reports the current value of the counter at the end of the day. Internally, the sensor stores the counter n as a vector of $O(\log n)$ bits.

   In 1978, Robert Morris discovered a simple randomized method to maintain an *approximation* of the counter value n using only $O(\log\log n)$ expected bits. Morris's counter maintains an integer L, but with different update and query algorithms:

   - **Initialize:** At the beginning of the day, set $L \leftarrow 0$.
   - **Update:** Each time a car passes the sensor, increment L with probability 2^{-L}.
   - **Query:** At the end of the day, report $\tilde{n} = 2^L - 1$.

   Intuitively, L is attempting to store the value $\log_2 n$.

   In the following questions, let $L(n)$ denote the value of L after n updates.

   (a) Prove that $\mathrm{E}[2^{L(n)}] = n + 1$.

   (b) Prove that $\mathrm{E}[4^{L(n)}] = O(n^2)$. *[Hint: Your proof should yield an exact expression, not just a big-O bound.]*

   (c) Compute $\mathrm{E}[(\tilde{n} - n)^2]$, where $\tilde{n} = 2^{L(n)} - 1$.

   (d) Prove that $\Pr[|\tilde{n} - n| > 4n/5] < 4/5$.

   Obviously, this analysis only implies that the Morris counter is a crude estimate, but the accuracy and confidence can be improved using the same median-of-means analysis as HW5.3. If we let N be the median of $O(\log(1/\delta))$ independent estimates, each of which is the mean of $O(1/\varepsilon^2)$ independent Morris counters, then $\Pr[|N - n| > \varepsilon n] < \delta$.

2. Suppose you are given a directed graph $G = (V, E)$, two vertices s and t, a positive capacity function $c \colon E \to \mathbb{R}^+$, and a second function $f : E \to \mathbb{R}$.

   (a) Describe and analyze an efficient algorithm to determine whether f is a maximum (s, t)-flow in G.

   (b) Describe and analyze an efficient algorithm to determine whether f is the *unique* maximum (s, t)-flow in G.

   Do not assume ***anything*** about the function f.

3. Suppose you are given an $n \times n$ checkerboard with some of the squares deleted. You have a large set of dominos, just the right size to cover two squares of the checkerboard. Describe and analyze an algorithm to determine whether one tile the board with dominos—each domino must cover exactly two undeleted squares, and each undeleted square must be covered by exactly one domino.

   Your input is a boolean array $Deleted[1..n, 1..n]$, where $Deleted[i, j] = \text{True}$ if and only if the square in row i and column j has been deleted. Your output is a single boolean; you do ***not*** have to compute the actual placement of dominos. For example, for the board shown below, your algorithm should return True.