CS 473 ✧ Fall 2024

Homework 5

Due **Wednesday, October 16**, 2024 at 9pm Central Time

Unless a problem specifically states otherwise, you may assume a function RANDOM that takes a positive integer k as input and returns an integer chosen uniformly and independently at random from $\{1, 2, \ldots, k\}$ in $O(1)$ time. For example, to model a fair coin flip, you could call RANDOM(2).

---

1. Recall that a *priority search tree* is a binary tree in which every node has both a *search key* and a *priority*, arranged so that the tree is simultaneously a binary search tree for the keys and a min-heap for the priorities. A ***heater*** is a priority search tree in which the *priorities* are given by the user, and the *search keys* are distributed uniformly and independently at random in the real interval $[0, 1]$. Intuitively, a heater is a sort of anti-treap.

   The following problems consider an n-node heater T whose priorities are the integers from 1 to n. We identify nodes in T by their *priorities*; thus, "node 5" means the node in T with *priority* 5. For example, the min-heap property implies that node 1 is the root of T. Finally, let i and j be integers with $1 \leq i < j \leq n$.

   (a) What is the *exact* expected depth of node j in an n-node heater? Answering the following subproblems will help you:

      i. Prove that in a random permutation of the $(i+1)$-element set $\{1, 2, \ldots, i, j\}$, elements i and j are adjacent with probability $2/(i+1)$.

      ii. Prove that node i is an ancestor of node j with probability $2/(i+1)$. *[Hint: Use the previous question!]*

      iii. What is the probability that node i is a *descendant* of node j? *[Hint: Do **not** use the previous question!]*

   (b) Describe an algorithm to insert a new item into a heater. Analyze the expected running time as a function of the number of nodes.

   (c) Describe an algorithm to delete the minimum-priority item (the root) from an n-node heater. What is the expected running time of your algorithm?

2. Suppose we generate a bit-string w by flipping a fair coin n times. Thus, each bit in w is equal to `0` or `1` with equal probability, and the bits in w are fully independent. A ***run of length ℓ*** in w is a substring of length ℓ in which all bits are equal. For example, the string `01000011101` contains ***three*** runs of length 3, starting at the third, fourth, and seventh bits

   (a) Suppose n is a power of 2. Show that the expected number of runs of length $\lg n + 1$ is $1 - o(1)$. (Here "lg" is standard shorthand for log-base-2.)

   (b) Show that, for sufficiently large n. the probability that *every* run in w has length less than $\lfloor \lg n - 2 \lg \lg n \rfloor$ is less than $1/n$. *[Hint: Break w into disjoint substrings of length $\lfloor \lg n - 2 \lg \lg n \rfloor$ and use the following fact: The event that all bits in one substring are equal is independent of the event that all bits in any other substring are equal.]*

3. Suppose we are given a coin that may or may not be biased, and we would like to compute an accurate *estimate* of the probability of heads. Specifically, if the actual unknown probability of heads is p, we would like to compute an estimate $\tilde{p}$ such that

$$\Pr[|\tilde{p} - p| > \varepsilon] < \delta$$

where ε is a given ***accuracy*** or ***error*** parameter, and δ is a given ***confidence*** parameter.

The following algorithm is a natural first attempt; here FLIP( ) returns the result of an independent flip of the unknown coin.

```
MEANESTIMATE(ε):
  count ← 0
  for i ← 1 to N
      if FLIP( ) = HEADS
          count ← count + 1
  return count/N
```

(a) Let $\tilde{p}$ denote the estimate returned by MEANESTIMATE(ε). Prove that $\mathrm{E}[\tilde{p}] = p$.

(b) Prove that if we set $N = \lceil \alpha/\varepsilon^2 \rceil$ for some appropriate constant α, then we have $\Pr[|\tilde{p} - p| > \varepsilon] < 1/4$. *[Hint: Use Chebyshev's inequality.]*

(c) We can increase the previous estimator's confidence (for the same accuracy) by running it multiple times, independently, and returning the *median* of the resulting estimates.

```
MEDIANOFMEANSESTIMATE(δ, ε):
  for j ← 1 to K
      estimate[j] ← MEANESTIMATE(ε)
  return MEDIAN(estimate[1..K])
```

Let p^* denote the estimate returned by MEDIANOFMEANSESTIMATE(δ, ε). Prove that if we set $N = \lceil \alpha/\varepsilon^2 \rceil$ (inside MEANESTIMATE) and $K = \lceil \beta \ln(1/\delta) \rceil$, for some appropriate constants α and β, then $\Pr[|p^* - p| > \varepsilon] < \delta$. *[Hint: Use Chernoff bounds.]*