CS 537: *Randomness in Computing*
Boston University

Prof. Sofya Raskhodnikova
TF: Gavin Brown
February 28, 2020

Solutions to Homework 6

1. **(Improving guarantees of randomized algorithms)**

(a) *You have developed a randomized algorithm $\mathcal{A}$ that, on every input of length n, runs in time $O(n^2)$ and outputs either a correct answer for the problem you are trying to solve or "fail". You proved that, on every input, it returns "fail" with probability at most 0.99. Show how to modify algorithm $\mathcal{A}$ to get a new algorithm that computes a correct answer in expected time $O(n^2)$.*

In order to solve the problem with certainty, one can repeatedly run algorithm $\mathcal{A}$ until it outputs the correct answer. Let X be the random variable corresponding to the number of times one needs to run algorithm $\mathcal{A}$ in order for it to succeed. Then $X \sim \mathit{Geom}(p)$ where $p \geq 0.01$. In particular,

$$\mathbb{E}[X] = \frac{1}{p} \leq 100.$$

The running time of the new procedure is the running time of $\mathcal{A}$ multiplied by X. Its expectation is at most $100 \cdot O(n^2)$, which is $O(n^2)$.

(b) *You have developed a randomized algorithm $\mathcal{B}$ that always solves your problem and, on every input of length n, runs in expected time $T(n)$. Show how to modify algorithm $\mathcal{B}$ to get a new algorithm that solves your problem with probability at least 0.95 and always runs in time $a \cdot T(n)$, for as small constant a as you can.*

The algorithm is as follows: on inputs on length n run $\mathcal{B}$ for time $20 \cdot T(n)$. If $\mathcal{B}$ stops within that time, output the correct answer, otherwise output `FAIL`. To bound the probability of failure, let random variable X_n be the running time of $\mathcal{B}$, so $\mathbb{E}[X_n] = T(n)$. Since X_n is nonnegative, we can apply Markov's inequality:

$$\Pr[\texttt{FAIL}] = \Pr[X_n \geq 20 \cdot T(n)] \leq \frac{T(n)}{20 \cdot T(n)} = \frac{1}{20}.$$

Your friend manages to prove that the variance of the running time of $\mathcal{B}$ on input of length n is at most $\sqrt{n}$. How should you modify your solution above to obtain the best running time while still solving the problem with probability at least 0.95?

Similarly to above, run $\mathcal{B}$ for $T(n) + \sqrt{20}n^{1/4}$ steps and, if it doesn't halt before then, output `FAIL`. By Chebyshev's inequality,

$$\Pr[\texttt{FAIL}] = \Pr[X_n \geq T(n) + \sqrt{20}n^{1/4}] \leq \Pr[|X_n - T(n)| \geq \sqrt{20}n^{1/4}] \leq \frac{\mathrm{Var}(X_n)}{20n^{1/2}} \leq \frac{1}{20}.$$

(c) *You have developed a randomized algorithm $\mathcal{B}$ for computing a function f that, on every input x, returns the correct answer $f(x)$ with probability at least 0.7 and an incorrect answer with the remaining probability. To amplify the success probability, you do the following: you run your algorithm k times and output the answer that appears most frequently in the k runs (breaking ties arbitrarily). Let $t > 0$ be a parameter. Use a Chernoff-Hoeffding bound to find a value of k that ensures that the new algorithm makes a mistake in computing $f(x)$ with probability at most 2^{-t}.*

Let X_i be the indicator random variable corresponding to the event that, in run $i \in [k]$, algorithm $\mathcal{B}$ computes the right answer. Then $\mathbb{E}[X_i] \geq 0.7$. Also, let $X = \sum_{i \in [k]} X_i$.

If $X > \frac{k}{2}$, the new procedure outputs the right answer. Let W be the even that the new procedure give a wrong answer. We apply the Hoeffding Bound for the lower tail from the slides (if you applied Theorem 4.12 from Mitzenmacher-Upfal for both tails, your answer would be a factor of 2 larger):

$$\Pr[W] \leq \Pr\left[X \leq \frac{k}{2}\right] = \Pr\left[\sum_{i\in[k]} X_i \leq 0.5k\right] = \Pr\left[0.5k - \sum_{i\in[k]} X_i \geq 0.2k\right]$$
$$\leq e^{-2k(0.2)^2} = 2^{-k\cdot 0.08 \ln 2}.$$

We want to ensure that this probability is at most 2^{-t}. That is, $k \cdot 0.08 \ln 2 \geq t$, and hence $k \geq t/(0.08 \ln 2) \approx 18.04t$. It is sufficient to take $k \geq 19t$.

2. **(Exercise 4.10 from MU)** *A casino is testing a new class of simple slot machines. Each game, the player puts in \$1, and the slot machine is supposed to return either \$3 to the player with probability 4/25, \$100 with probability 1/200, or nothing with all the remaining probability. Each game is supposed to be independent of other games.*

   The casino has been surprised to find in testing that the machines have lost \$10,000 over the first million games. Your task is to come up with an upper bound on the probability of this event, assuming that their machines are working as specified.

   (a) *Use Theorem 4.12 (Hoeffding Bound) from the MU book to give an upper bound.* Let X be random variable that denotes the loss of the casino in one million games. Our goal is to find an upper bound on $\Pr[X = 10000]$.

   Let X_i be the loss of the casino in the i-th game. If the player gets nothing at the end of the round, the loss of the casino is $\$-1$. If the player gets \$100 at the end of the round, the loss of the casino is 99. For every $i \in \mathbb{N}$, the values taken by the random variable X_i are bounded by -1 and 99. In other words, $\Pr[X_i \in [-1, 99]] = 1$ for all $i \in \mathbb{N}$.

   The probability that the casino gains \$1 in a game is equal to $1 - \frac{4}{25} - \frac{1}{200} = \frac{167}{200}$. Thus, the expected loss of the casino in the ith game is

$$\mu = \mathbb{E}[X_i] = -1 \cdot \frac{167}{200} + 2 \cdot \frac{4}{25} + 99 \cdot \frac{1}{200} = \frac{-4}{200} = \frac{-1}{50} = -0.02.$$

   Now we apply the Hoeffding bound for the upper tail from the slides. (If you apply the one from the textbook, which takes into account both tails, you get a bound that is twice as large and is greater than 1).

$$\Pr[X = 10000] \leq \Pr[X \geq 10000] = \Pr\left[\sum_{i=1}^{10^6} X_i \geq 10000\right]$$
$$= \Pr\left[\frac{1}{10^6}\sum_{i=1}^{10^6} X_i \geq 0.01\right] = \Pr\left[\frac{1}{10^6}\sum_{i=1}^{10^6} X_i - \mu \geq 0.03\right]$$
$$\leq \exp(\frac{-2 \cdot 10^6 \cdot 0.03^2}{100^2}) = \cdot \exp(-0.18) \leq 0.84.$$

In the rest of the problem, you will derive a specialized Chernoff bound for this problem to get a better upper bound.

(b) *Let the random variable X denote the* net loss *to the casino over the first million games. Derive an expression for $E[e^{tX}]$, where t is an arbitrary real number.*

Net loss $X = \sum_{i=1}^{10^6} X_i$, where X_i is as defined in the solution to part (a). Then

$$\mathbb{E}[e^{tX}] = \mathbb{E}[e^{t\sum_{i=1}^{10^6} X_i}] = \mathbb{E}[\prod_{i=1}^{10^6} e^{tX_i}] = \prod_{i=1}^{10^6} \mathbb{E}[e^{tX_i}].$$

The last equality above holds because random variables X_i, and hence random variables e^{tX_i} are independent. For every $i \in [10^6]$, by definition of expectation,

$$\mathbb{E}[e^{tX_i}] = e^{-t} \cdot \frac{167}{200} + e^{2t} \cdot \frac{4}{25} + e^{99t} \cdot \frac{1}{200} = \frac{167e^{-t} + 32e^{2t} + e^{99t}}{200}.$$

We substitute this expression in the product expression we obtained for $\mathbb{E}[e^{tX}]$:

$$\mathbb{E}[e^{tX}] = \left(\frac{167e^{-t} + 32e^{2t} + e^{99t}}{200}\right)^{10^6}.$$

(c) *Derive from first principles a Chernoff bound for the probability* $\Pr[X \geq 10,000]$. (Hint: *Follow the proof of the Chernoff bound in class, by applying Markov's inequality to the random variable e^{tX}. Use the value $t = 0.0006$ in your bound.)*

$$\begin{aligned}\Pr[X \geq 10000] &= \Pr[e^{tX} \geq e^{10000t}] \\ &\leq \frac{\mathbb{E}[e^{tX}]}{e^{10000t}} \\ &= \frac{\left(\frac{167e^{-t}+32e^{2t}+e^{99t}}{200}\right)^{10^6}}{e^{10000t}}.\end{aligned}$$

Substituting $t = 0.0006$ in this expression (and performing the computations), we get 1.6×10^{-4}. This is four orders of magnitude smaller than the probability bound we obtained by naively applying the Hoeffding bound.

3. **(Randomized Quicksort)** *Exercise 4.21 from Mitzenmacher-Upfal. (Recall that we analyzed the expected running time of Quicksort in class. In this problem, you will work out a high probability statement about the running time.)*

(a) We call the size of the set to be sorted at a node of the binary tree (described in the problem) *the size* of that node. Consider an arbitrary root-to-leaf path P in the tree. Let d denote the number of good nodes on that path.

Claim. *For all $k \in [d]$, a child of the kth (from the root) good node of P has size at most $\left(\frac{2}{3}\right)^k n$.*

Proof. The size of a child of a good node Γ is at most $2/3$ times the size of Γ. We prove the claim by induction.

Base Case: The root has size n. Hence, the size of a child of the first good node on the path is at most $\frac{2n}{3}$.

Induction Hypothesis: Assume that the claim is true for all $k \in [\ell]$ for some $\ell \in [d-1]$.

Inductive Step: Consider $k = \ell + 1$. A child of the $(k-1)$st good node has size at most $\left(\frac{2}{3}\right)^{k-1} n$ by the induction hypothesis. Hence, the kth good node, which is a descendant of the $(k-1)$st good node, has size at most $\left(\frac{2}{3}\right)^{k-1} n$. Hence, by the definition of good nodes, a child of the kth good node has size at most $\left(\frac{2}{3}\right)^{k} n$. This proves the claim. □

The size of a child of the last good node on P has size at least 1, and hence $\left(\frac{2}{3}\right)^{d} n \geq 1$. Therefore, $d \leq c \ln n$ for $c = \frac{1}{\ln 1.5}$.

(b) Consider a leaf (a node that contains just one number from the input array) in the binary tree and the path P from the root to that leaf. Let $t = 12c \ln n$. A node is good if the pivot at that node is from among the middle one-third fraction of the values in the set at that node. In other words, each node is good independently with probability at least 1/3. When forming a path P from the root to our leaf, by part (a), we stop after getting $c \ln n$ good nodes (or sooner). So, we can get more than t nodes in our path only if in the first t trials, we get less than $c \ln n$ good nodes. We bound the probability of this event using a Chernoff bound.

Let X_i for $i \in [t]$ be the indicator random variable for the event that the ith node on P is good and $X = \sum_{i\in[t]} X_i$. Then $\mathbb{E}[X_i] \geq 1/3$ for all $i \in [t]$ and, by the linearity of expectation, $\mathbb{E}[X] \geq \frac{t}{3} = 4c \ln n$. Since X_i are independent, we can apply a Chernoff bound (Theorem 4.5, part 2) with the caveat explained in Exercise 4.7. We use it with $\delta = 3/4$ and $\mu_L = 4c \ln n$, where μ_L notes a lower bound on $\mathbb{E}[X]$:

$$\Pr[X \leq c \ln n] = \Pr[X \leq (1 - 3/4) \cdot 4c \log n] \leq e^{-\mu_L \cdot \delta^2/2} = e^{-4c \ln n \cdot 9/32} \leq n^{-2}.$$

Hence, with probability at least $1 - \frac{1}{n^2}$, an arbitrary root-to-leaf path has at most $c' \log n$ nodes, where $c' = 12c/\log_2 e$.

(c) By part (b), the probability that an arbitrary root-to-leaf path contains more than $c' \log n$ bad nodes is at most $\frac{1}{n^2}$. Since the tree has at most n leaves, the total number of root-to-leaf paths in the tree is at most n. Hence, the probability that there exists a root-to-leaf path containing more than $c' \log n$ bad nodes is, by a union bound, at most $\frac{1}{n}$. Thus, with probability at least $1 - \frac{1}{n}$, the longest root-to-leaf path contains no more than $c' \log n$ bad nodes.

(d) By part (c), the number of levels of recursion in the randomized Quicksort is at most $c' \log n$ with probability at least $1 - \frac{1}{n}$. The total running time at each level of recursion is $O(n)$, since the algorithm is simply partitioning the sets of elements around the corresponding pivots. Hence, the running time of randomized Quicksort is $O(n \log n)$ with probability at least $1 - \frac{1}{n}$.

4*. **(Optional, no collaboration)** *In this problem, you will analyze an algorithm for estimating the number of connected components in an undirected graph $G = (V, E)$ on n nodes within $\pm\epsilon n$, where $\epsilon \in (0, 1)$ is a parameter.*

(a) *Let C be the number of connected components in G. For every node v, let n_v denote the number of nodes in the connected component of v. Prove that $C = \sum_{v\in V} \frac{1}{n_v}$.*

For $i \in [C]$, let V_i be the set of vertices in connected component i. So if $v \in V_i$ then $|V_i| = n_v$. Then

$$\sum_{v\in V} \frac{1}{n_v} = \sum_{i\in[C]} \sum_{v\in V_i} \frac{1}{n_v} = \sum_{i\in[C]} 1 = C.$$

(b) *For every node v, let $\hat{n}_v = \min(n_v, 2/\epsilon)$. Let $\hat{C} = \sum_{v\in V} \frac{1}{\hat{n}_v}$. Can $\hat{C}$ be smaller than C? Larger than C? By how much? (Give the best upper bound you can.)*

$\hat{C}$ cannot be smaller than C since, for every term in the sum, we have

$$C = \sum_{v \in V} \frac{1}{n_v} \leq \sum_{v \in V} \max\left\{\frac{1}{n_v}, \frac{\epsilon}{2}\right\} = \hat{C}.$$

We always have $C \geq 1$ and $\hat{C} \leq n\epsilon/2$, with equality when the graph is connected. So $\hat{C} - C \leq n\epsilon/2$.

(c) *Let s be a parameter. Define $\tilde{C}$ to be an estimate obtained as follows:* We sample s uniformly random nodes from G independently with replacement. For each sampled node v, we compute $\hat{n}_v$ by doing a BFS from v for at most $2/\epsilon$ steps. We compute the average of values $\frac{1}{\hat{n}_v}$ over all sampled nodes and set $\tilde{C}$ to be n times the average.

Use a Chernoff-Hoeffding bound to find the (asymptotically) smallest value of s for which

$$\Pr[|\tilde{C} - \hat{C}| \geq \epsilon n/2] \leq 1/3.$$

For each $i \in [s]$, define a random variable X_i as $\frac{1}{\hat{n}_v}$ for the i-th vertex v in the sample, and let $X = \sum_{i \in [s]} X_i$. We will apply the Hoeffding bound from Lecture 10, slide 7. Note that $\frac{\epsilon}{2} \leq X_i \leq 1$, although we'll only use a lower bound of 0 in the Hoeffding bound. By definition, $\tilde{C} = \frac{n}{s} \cdot X$. Since X_i are identically distributed and by linearity of expectation, $\mathbb{E}[X] = \mathbb{E}[\sum_{i \in [s]} X_i] = s \cdot \mathbb{E}[X_1] = s \cdot \sum_{v \in V} \frac{1}{n} \frac{1}{\hat{n}_v} = \frac{s}{n} \cdot \hat{C}$.
By Hoeffding bound applied with with $b = 1$ and $a = 0$,

$$\begin{aligned} \Pr\left[\left|\tilde{C} - \hat{C}\right| \geq \frac{\epsilon n}{2}\right] &= \Pr\left[\left|\frac{n}{s} \cdot X - \frac{n}{s} \cdot \mathbb{E}[X]\right| \geq \frac{\epsilon n}{2}\right] = \Pr\left[|X - \mathbb{E}[X]| \geq \frac{\epsilon s}{2}\right] \\ &\leq 2e^{\frac{-2s}{(1-0)^2}\left(\frac{\epsilon}{2}\right)^2} = 2e^{-s\epsilon^2/2}. \end{aligned}$$

Setting this equal to $\frac{1}{3}$ yields $s = \Theta(1/\epsilon^2)$.

(d) *Argue that, with probability at least 2/3, the estimate $\tilde{C}$ approximates the number of connected components in G within $\pm \epsilon n$.*

Part (b) gives us $|\hat{C} - C| \leq \frac{\epsilon n}{2}$ and part (c) gives us, with probability at least 2/3, that $|\tilde{C} - \hat{C}| \leq \frac{\epsilon n}{2}$. By the triangle inequality, with probability at least 2/3,

$$|\tilde{C} - C| = |\tilde{C} - \hat{C} + \hat{C} - C| \leq |\tilde{C} - \hat{C}| + |\hat{C} - C| \leq \frac{\epsilon n}{2} + \frac{\epsilon n}{2} = \epsilon n.$$

(e) *If G has degree at most d, how many nodes does the procedure above visit, with the setting of s that you found?*

For a single sample, we run BFS at most $2/\epsilon$ steps. At each of these steps we visit all of the current node's at most d neighbors. With $\Theta(1/\epsilon^2)$ samples, this yields $O(d/\epsilon^3)$ nodes visited.