COMPSCI 514: Problem Set 1

Due: 9/24 by 11:59pm in Gradescope.

Instructions:

- You are allowed to, and highly encouraged to, work on this problem set in a group of up to three members.
- Each group should **submit a single solution set**: one member should upload a pdf to Gradescope, marking the other members as part of their group in Gradescope.
- You may talk to members of other groups at a high level about the problems but **not work through the solutions in detail together**.
- You must show your work/derive any answers as part of the solutions to receive full credit.

1. Probability and Concentration Bound Practice (8 points)

1. (2 points) Prove the union bound using Markov's inequality and indicator random variables. That is, prove that for any events $\mathbf{A}_1, \ldots, \mathbf{A}_m$, $\Pr[\mathbf{A}_1 \cup \ldots \cup \mathbf{A}_m] \leq \sum_{i=1}^{m} \Pr[\mathbf{A}_i]$.
2. (2 points) The maximum score on an problem set is 100% and the class average is 85%. What is the maximum fraction of students who could have scored at or below 50%?
3. (2 points) I store $1,000$ items in a hash table with $100,000$ buckets, using a fully random hash function. What is the probability that there is **at least 1** collision. What if I use $1,000,000$ buckets? What is the probability that there is at least one collision? (**Hint:** You might want to write down a formula and then write a program or use Wolfram Alpha or a similar program to compute these probabilities.)
4. (2 points) Give an example of a random variable $\mathbf{X}$ and a deviation t where Markov's inequality actually gives a tighter bound than Chebyshev's inequality.

2. Maximum Server Loads (8 points)

Suppose there are n servers and k requests. Each request is equally likely to be assigned to any of the servers and all requests are assigned independently of the others. Let $\mathbf{R}_i$ be the number of requests assigned to the ith server.

1. (2 points) Prove that $\Pr[\mathbf{R}_1 \geq t] \leq \binom{k}{t} \cdot 1/n^t$. **Hint:** Consider the union bound with events of the form B_A where A is a subset of $\{1, \ldots, k\}$.
2. (2 points) Prove that $\Pr[\max(\mathbf{R}_1, \mathbf{R}_2, \ldots, \mathbf{R}_n) \geq t] \leq k^t/n^{t-1}$. **Hint:** Note that $\binom{a}{b} \leq a^b$ for any positives integers a and b where $b \leq a$.

3. (2 points) If $k = n/2$, prove that all servers receive $\leq 2\log n$ requests with probability at least $1 - 1/n$. Here $\log n$ is base 2.

4. (2 points) If $k = n$, prove that all servers receive $\leq 4\log n$ requests with probability at least $1 - 2/n$. Here $\log n$ is base 2.

3. Minimum Server Loads (6 points)

Suppose there are n servers. Each request is equally likely to be assigned to any of the servers and all requests are assigned independently of the others. Let $\mathbf{R}_i$ be the number of requests assigned to the ith server.

1. (2 points) Express $\Pr[\mathbf{R}_i = 0]$ as a function of n and k.

2. (2 points) Prove that if $k = 2n\ln n$, then $\Pr[\mathbf{R}_i = 0] \leq 1/n^2$. **Hint:** Note that for any $x \geq 0$, $(1 + x) \leq e^x$.

3. (2 points) Prove that if $k = 2n\ln n$, then the probability every server needs to process at least one request is at least $1 - 1/n$.

4. Designing Your Own Random Hash Function (8 points + 2 bonus points)

In this question you will implement your own random hash function, and test empirically if it satisfies the properties discussed in class.

1. (2 points) Consider a random hash function that takes as input any integer x and maps it to some index in $\{0, \ldots, 99\}$. E.g., an example described in class picks random a, b and lets $\mathbf{h}(x) = (ax+b \mod p) \mod 100$, for a large prime p. Another example might simply append x to a random integer a to obtain an integer $[x; a]$ and then return $[x; a] \mod 100$.

   Design your own random hash function mapping integer inputs to $\{0, \ldots, 99\}$. Describe it in words/pseudocode and implement it in your favorite programming language.

2. (2 points) Compute $\mathbf{h}(1)$ for $n = 10,000$ random instantiations of your hash function. Plot a chart showing the number of times that $\mathbf{h}(1) = i$ for all $i \in \{0, \ldots, 99\}$. Do you think your hash function satisfies $\Pr[\mathbf{h}(1) = i] = 1/100$ for all i? Or is close to satisfying this? Run the test for a few other input values aside from 1. Do you see similar results? Are there any input values where the distribution of outputs is far from uniform?

3. (2 points) Compute $\mathbf{h}(1), \mathbf{h}(2)$ for $n = 10,000$ random instantiations of your hash function. If $\mathbf{h}$ were 2-universal, give an upper bound on the number of hash collisions you would expect to see (i.e., the number of trials for which you expect to have $\mathbf{h}(1) = \mathbf{h}(2)$). Given this, do you think your hash function is 2-universal or close to it? Try out a few other pairs of inputs. Do you see the same behavior? Can you find any pairs of values where the number of collisions is much higher than expected for a 2-universal hash function?

4. (2 points) Compute $\mathbf{h}(1) - \mathbf{h}(2)$ for $n = 10,000$ random instantiations of your hash function. Plot a chart showing the number of times that $(\mathbf{h}(1) - \mathbf{h}(2)) = i$ for all $i \in \{-99, \ldots, 99\}$. If $\mathbf{h}$ were pairwise independent, compute $\Pr[(\mathbf{h}(1) - \mathbf{h}(2)) = i]$ for any i. Given this result and considering your chart of plotted values, do you think your hash function is indeed pairwise independent or close to it? Try out a few other pairs of inputs. Do you see the same behavior?

5. (2 points) **Bonus:** Give an example of a random hash function $\mathbf{h} : U \to [m]$, such that $\mathbf{h}(x)$ is equally likely to be any element of $[m]$ (i.e., $\mathbf{h}$ is 1-universal) but $\mathbf{h}$ is not 2-universal.

For full credit, include any code used in your pdf submission. We will not run the code, but will sanity check it.

5. Randomized Load Balancing Meets Scalability (6 points)

Consider a large scale distributed database, storing items coming from some universe U. A random hash function $\mathbf{h} : U \to [k]$ is used to assign each item x to one of k servers. When that item is queried in the future, the hash function is used to identify which server it is stored on. In many applications, the number of servers scales dynamically, depending on the storage load, availability, etc. If a new server is added to the current set of k, since $\mathbf{h}$ maps only to $[k]$, we will have to pick a new hash function, rehash and move all the stored items.

Consider the following solution: Let $\mathbf{h} : U \times \mathbb{Z} \to [0, 1]$ be a random hash function which maps a (item, server id) pair to a uniform random value in the range $[0, 1]$. Each item x is stored on the server for which $\mathbf{h}(x, i)$ is the largest (since you are mapping to a continuous range, you may assume that there are never ties). If a server with id i is added to the system, we compute $\mathbf{h}(x, i)$ for all stored items, and move any items to server i for which $\mathbf{h}(x, i)$ is now the maximum hash value for x. Similarly, if server i is removed from the system, we simply reassign any item x on that server to the server j for which $\mathbf{h}(x, j)$ is the next highest.

1. (2 points) Consider the case where we have n items, stored on k servers. Let $\mathbf{R}_i$ be the expected load on server i (i.e., the number of items stored on that server). What is $\mathbb{E}[\mathbf{R}_i]$?

2. (2 points) If we have n items, originally stored on k servers, and we add a new server using this method, what is the expected number of items that will be moved to that new server?

3. (2 points) Consider the case where we have n items, stored on k servers. Show that with probability $\geq 99/100$ the number of items moved when a new server is either added or removed from the system is $\leq \frac{20n \ln k}{k}$.

   You may assume that k and $\frac{n}{k}$ are both large, say > 5 to get your bounds to hold.

6. Exponential Tail Bounds from Scratch (8 points)

Throughout, let $\exp(x)$ denote e^x.

1. (2 points) Let $\mathbf{Y}$ be a random variable that takes values in the interval $[0, 1]$ and not just 0 or 1. Prove that for $t > 0$, $\mathbb{E}[\exp(t\mathbf{Y})] \leq 1 + \mathbb{E}[\mathbf{Y}](e^t - 1)$. **Hint:** First show that $\exp(ty) \leq 1 + y(e^t - 1)$ for all $y \in [0, 1]$.

2. (2 points) Let $\mathbf{X}_1, \ldots, \mathbf{X}_n$ be independent random variables that take values in the interval $[0, 1]$. Let $\mathbf{X} = \sum_{i \in [n]} \mathbf{X}_i$ and $\mu = \mathbb{E}[\mathbf{X}]$. Prove that for $0 < \delta < 1$,

$$\Pr[\mathbf{X} \geq (1 + \delta)\mu] = \Pr[\exp(t\mathbf{X}) \geq \exp(t(1 + \delta)\mu)] \leq \frac{\mathbb{E}[\exp(t\mathbf{X})]}{\exp(t(1 + \delta)\mu)} .$$

   Hint: Use the Markov bound.

3. (4 points) Prove that $\Pr[\mathbf{X} \geq (1 + \delta)\mu] \leq \exp(-\delta^2\mu/3)$.

   Hint: Consider setting $t = \ln(1 + \delta)$ and using part one of the question. You may use the fact that $(1 + \delta)^{1+\delta} \geq e^{\delta + \delta^2/3}$ and that for any $x \geq 0$, $(1 + x) \leq e^x$. You may also want to recall that for independent random variables $\mathbf{Y}, \mathbf{Z}$, $\mathbb{E}[\mathbf{YZ}] = \mathbb{E}[\mathbf{Y}] \cdot \mathbb{E}[\mathbf{Z}]$.