COS 330: Great Ideas in Theoretical Computer Science

Fall 2025

Problem Set 3

Module: Randomization

Below is a reminder of key aspects of the PSet:

- The only goal of this PSet is to help you develop your problem-solving skills in preparation for the exams. Your performance on this PSet will not directly contribute to your grade, but will indirectly improve your ability to do well on the exams.
- Because your performance does not directly impact your grade, you may use any resources you like (collaboration, AI, etc.) to help you complete the PSet.
- We suggest taking a serious stab at the PSet alone, to help self-evaluate where you're at. But, we also suggest collaborating with friends, visiting office hours, asking on Ed, and/or using AI tools to help when stuck. Even when able to complete the entire PSet on your own, you may still find any of these methods useful to discuss the PSet afterwards.
- Throughout the PSet, we've included some general tips to help put these into broader context. Exams will not have these, and future PSets may have fewer.

Aligning Expectations

At the end of this PSet we've included an appendix with some common inequalities and bounds that you may find useful. We don't expect you to memorize these, so we will provide a similar appendix on the exam.

However, note that as you work on the problems, you should try to develop intuition for when and how to apply these inequalities. In particular, you should learn to recognize the main differences between the inequalities (e.g., Chernoff is stronger than Markov, but requires independence and boundedness). When solving a problem, you should be able to identify which inequality is most appropriate to apply, and use the appendix as a reference for the exact statement. See this Ed post for advice on how to think about and use these inequalities.

Problem Solving Tips

Don't assume the problems are sorted by difficulty. If you get stuck on a problem, try a different one and come back later. If you get stuck on a subproblem, try a different subproblem or problem and come back later (on subproblems that say "use the result from part (a)", you

can assume the result from part (a) even if you didn't solve it).

If after spending 30 minutes on a problem you feel like you're not making any progress, consider asking for help. Ask questions on Ed, go to office hours, or feel free to ask AI for hints or explanations. However, avoid giving up too quickly. The more time you spend struggling with a problem, the more you will learn from it. Make sure you give a problem at least 30 minutes of 100% focused effort before asking for help, but treat this as a lower bound, not as an "always ask for help after 30 minutes".

Problem 1 : Is Quicksort Quick?

Recall the Quicksort algorithm you've learned in the past, with the pivot element chosen uniformly at random from the array. Here is a brief description of the algorithm:

- Pick a pivot element uniformly at random from the array.
- Partition the remaining elements into two subarrays: those less than the pivot and those greater than or equal to the pivot.
- Recursively apply Quicksort to both subarrays.
- Concatenate the sorted left subarray, the pivot, and the sorted right subarray to form the final sorted array.
- Repeat until the base case of a single element or empty array is reached.

Let's analyze the expected number of comparisons this randomized Quicksort algorithm makes when sorting an array of n distinct elements. Let $a_1, a_2, \ldots, a_n$ be the elements of the array, and $s_1 < s_2 < \ldots < s_n$ be the sorted order of these elements. Define the indicator random variable $X_{i,j}$ for $1 \leq i < j \leq n$ as 1 if s_i and s_j are compared during the execution of the algorithm, and 0 otherwise.

(a) Determine the expected value $\mathbb{E}[X_{i,j}]$ for $1 \leq i < j \leq n$.

(b) Using the result from part (a), show that the expected total number of comparisons made by the randomized Quicksort algorithm is at most $2n \ln n$ (note that ln is the natural logarithm).

(c) Show that with probability at least $4/5$, the number of comparisons made by the randomized Quicksort algorithm is at most $10n \ln n$.

Problem 2 : A Load Off

Consider n servers that need to handle n incoming requests (e.g., queries to an LLM server). Each request is assigned to a server uniformly at random and independently of other requests. We want to analyze the load on the servers, where the load on a server is defined as the number of requests assigned to it.

(a) What is the expected load on each server?

(b) Show that the expected number of servers with a load of exactly 2 is $\sim n/(2e)$.

(c) Show that the probability that any server has a load of more than $2 \ln n$ is at most $1/n$.

Problem 3 : Learning to Learn

Consider a probability distribution over the set $\{1, 2, \ldots, n\}$ which is described by a vector $p = (p_1, p_2, \ldots, p_n)$, where p_i is the probability of sampling the element i. This probability distribution is unknown to you and your goal is to find an approximation $\hat{p} = (\hat{p}_1, \hat{p}_2, \ldots, \hat{p}_n)$ such that the two distributions are close in the following metric: $\sum_{i=1}^{n} |p_i - \hat{p}_i| \leq \epsilon$[1].

(a) Show that if p and $\hat{p}$ are close according to the metric above, then for any function $f : \{1, 2, \ldots, n\} \to [0, 1]$, the expected values of f under the two distributions are also close, i.e.,

$$|\mathbb{E}_{i \sim p}[f(i)] - \mathbb{E}_{i \sim \hat{p}}[f(i)]| \leq \epsilon.$$

Note that the above is one way of justifying why we care about approximating the distribution p in this metric.

(b) Suppose you independently sample m times from the p distribution, obtaining samples $x_1, x_2, \ldots, x_m$. Let $\hat{p}$ be the empirical distribution, where $\hat{p}_i = \frac{|\{x_j = i\}|}{m}$ is the fraction of samples that equals to i.

Show that if $m \geq \Omega\left(\frac{n^2 \ln(n/\delta)}{\epsilon^2}\right)$, then with probability at least $1 - \delta$, the empirical distribution $\hat{p}$ satisfies

$$\sum_{1 \leq i \leq n} |p_i - \hat{p}_i| \leq \epsilon.$$

Problem 4 : Tiny Counting

Consider the task of maintaining a counter, which is initially 0. The counter supports two operations:

- `increment`(): Increases the value of the counter by 1.
- `get`(): Returns the current value of the counter.

A simple way to implement this counter is to use a single integer variable to store the current value of the counter. This needs $\lceil \log_2(m+1) \rceil$ bits to count up to m. Here is an alternative way:

- Let X be a variable which is initially 0.
- Implement `increment`() as follows: with probability $1/2^X$, increase X by 1, otherwise, do nothing.
- Implement `get`() as follows: return $2^X - 1$.

[1] This metric is very close to a common metric called total variation distance, which is defined as $\frac{1}{2}\sum_{i=1}^{n} |p_i - \hat{p}_i|$. The factor of $1/2$ is included for technical reasons, for example, it ensures that the distance lies between 0 and 1.

(a) Show that after m calls to `increment()`, the expected value returned by `get()` is m and its variance is at most $2m^2 + 1$.

(Hint 1: Use induction on the number of calls to `increment()` to prove both the expectation and variance bounds.)

(Hint 2: Define $P_{m,k}$ to be the probability that after m calls to `increment()`, the value of X is k, and use it to compute the expectation and variance. Write a recurrence relation for $P_{m,k}$ and use it in the induction proof.)

Note that the above implies that the expected number of bits used by this counter after m calls to `increment()` is $O(\log \log m)$.

(b) Show that after m calls to `increment()`, the value returned by `get()` is within $[m - \epsilon m, m + \epsilon m]$ with probability at least $1 - \frac{3}{\epsilon^2}$.

Appendix

Union Bound. For any events $A_1, A_2, \ldots, A_n$,

$$P\left(\bigcup_{i=1}^{n} A_i\right) \leq \sum_{i=1}^{n} P(A_i).$$

Markov's Inequality. Let X be a non-negative random variable. Then, for any $a > 0$,

$$P(X \geq a) \leq \frac{\mathbb{E}[X]}{a}.$$

Chebyshev's Inequality. Let X be a random variable with mean μ and variance σ^2. Then, for any $k > 0$,

$$P(|X - \mu| \geq k\sigma) \leq \frac{1}{k^2}.$$

Chernoff Bound Let $X_1, X_2, \ldots, X_n$ be independent random variables such that $0 \leq X_i \leq 1$ for all i. Let $X = \sum_{i=1}^{n} X_i$ and let $\mu = \mathbb{E}[X]$. Then for any $t > 0$,

$$\Pr[X \geq \mu + t] \leq \exp\left(-\frac{t^2}{2\mu + t}\right).$$

and

$$\Pr[X \leq \mu - t] \leq \exp\left(-\frac{t^2}{2\mu}\right).$$

Two-Sided Chernoff Bound Let $X_1, X_2, \ldots, X_n$ be independent random variables such that $0 \leq X_i \leq 1$ for all i. Let $X = \sum_{i=1}^{n} X_i$ and let $\mu = \mathbb{E}[X]$. Then for any $t > 0$,

$$\Pr[|X - \mu| \geq t] \leq 2\exp\left(-\frac{t^2}{3\mu}\right).$$