Collaboration Policy: You may collaborate with other students on these problems. Collaboration is limited to discussion of ideas only, and you should write up the solutions entirely on your own and list your collaborators as well as cite any references (book, paper, etc.) you may have used. Limit your answers for each problem to two pages or less (one page for most problems) – you need to give enough detail to convince the grader.

1. Suppose we throw $n$ balls randomly into $n$ bins. Show that the expected number of bins with no balls is approximately $1/e$. Give an approximate estimate (a clean number like $1/e$) for the expected number of bins with $i$ balls where $i$ is small. Also, identify a function $c(n)$ such that the number of bins with $c(n)$ balls is zero with high probability.

Repeat the above calculations when we randomly throw $n \log n$ balls into $n$ bins.

2. In class, we constructed pairwise independent hash functions using random linear functions modulo a prime $p$. It was also mentioned that if we use random univariate polynomial of degree $k - 1$ then the hash function is $k$-wise independent. Prove this. (Hint: You will need at some point the fact that a certain matrix called the Vandermonde matrix is invertible.)

3. Hashing can be viewed as throwing $n$ balls into $n$ bins, using the hash value $h(i)$ to determine which bin to throw the $i$th ball into. Prove that if we use a $k$-wise independent hash function where $k$ is a constant, then with high probability every bin has at most $O(n^{1/k})$ balls.

4. Consider the following modification of the AMS sketch. Rather than pick a dense random $m \times n$ sign matrix $\Pi$, we instead in each column of $\Pi$ pick a random location and set it to be a random sign (and zero the rest of the column out). Thus, there is a hash function $h : [n] \to [m]$ that decides where the non-zero is in each column. Note this has the advantage that we can process any stream update in constant time (as opposed to the AMS sketch, which needs $O(m)$ time). What does $m$ need to be to make this construction give a $1+\epsilon$ approximation with probability $2/3$, and how much independence do we require from $h$?
5. The \textit{k}-server problem is a well studied problem in online algorithms. Here, we are given a set of \(n\) points \(\{x_1, \ldots, x_n\}\) with distances between them (that satisfy triangle inequality). We have \(k\) servers, each at one of the \(n\) points. The request sequence is a sequence \(\sigma_1, \sigma_2, \ldots\) of points in the space. When request \(\sigma_i\) is received, the algorithm must move one of the servers to ensure that there is a server at \(\sigma_i\).

(a) Show that the \(k\)-server problem is a generalization of caching, i.e. show that the caching problem can be phrased as a \(k\)-server problem on an appropriately chosen set of points.

(b) Consider the \(k\)-server problem for \(n\) points on a line such that consecutive points have distance 1. The natural greedy algorithm is the following: move the closest server to the request, break ties arbitrarily. Show that this is not competitive.

(c) Here is an algorithm \textsc{Double Coverage} for \(k\)-server on the line: If \(\sigma_i\) falls between two servers, then both are moved at the same speed towards \(\sigma_i\) until one reaches \(\sigma_i\). Show that this algorithm is \(k\)-competitive. (\textit{Hint}: use the potential function \(\Phi\) which measures the cost of the min cost matching between the \(k\) locations of the algorithm’s servers and the \(k\) locations of the optimal algorithms servers.)