Problem Set 3

Randomized Algorithms

Due Wednesday, October 14

1. For a hash family $\mathcal{H}$ from $[U]$ to $[n]$, at a set of items $S \subset [U]$, let $X(\mathcal{H}, S)$ be the random variable denoting the load in the first bin:

$$X := |\{i \in S \mid h(i) = 1\}|$$

as a distribution over $h \in \mathcal{H}$. Further, let $f(\mathcal{H}, S)$ denote the expected max load in any bin:

$$f(\mathcal{H}, S) := \mathop{\mathbb{E}}_{h \in \mathcal{H}} \max_{j \in [n]} |\{i \in S \mid h(i) = j\}|.$$

   (a) For any $t \geq 1$, and for any k-wise independent hash family $\mathcal{H}$ with $k = O(1)$, and any set S with $|S| = n$, show that

   $$\Pr[X \geq t] \lesssim 1/t^k.$$

   Hint: bound $\mathbb{E}[X^k]$.

   (b) Show that for a k-wise independent family $\mathcal{H}$, $k = O(1)$, that

   $$f(\mathcal{H}, S) \lesssim n^{1/k}$$

   for any S with $|S| = n$.

   (c) Show that there exists a pairwise independent hash family $\mathcal{H}$ and set S with $|S| = n$ such that

   $$f(\mathcal{H}, S) \gtrsim \sqrt{n}.$$

2. [Karger.] In class, we showed that cuckoo hashing achieves worst case constant time lookups and expected constant time insertion/deletion, with $O(n)$ space to store n items. Show how to get the same guarantees, but using only $(1 + \epsilon)n$ space for a small constant ϵ. For this problem, assume that you have access to a perfectly random hash function. **Hint:** Use the following ideas:

- Probing more than twice in a table increases the chances of finding an empty cell.
- If after some probes you fail to find an empty cell, move the failed item into an "overflow" table that uses cuckoo hashing.

3. A *minimal perfect* hash function for a set S of size n is one that maps S to $[n]$ with no collisions. In class, we showed how to take S and construct a minimal perfect hash function for S that can be evaluated in constant time. The construction took expected $O(n)$ time and the resulting function took $O(n)$ words to store.

   Show that this last condition cannot be significantly improved upon. In particular, show that any procedure for storing a minimal perfect hash function requires at least $\Omega(n)$ bits for some S of size n. Assume the universe size U is polynomial in n. **Hint:** show that any particular function h is perfect for at most a $1/2^{\Omega(n)}$ fraction of the possible sets S.

4. [Karger.] Bloom filters can be used to estimate the difference between two sets. Suppose that you have sets X and Y, each with m elements, and with r elements in common. Create an n-bit Bloom filter for each, using the same k hash functions. Determine the expected number of bits where the two Bloom filters differ, as a function of m, n, k, and r. Explain how this could be used as a technique for estimating r.