CS 473 ✧ Fall 2024

Homework 6

Due **Wednesday, October 23**, 2024 at 9pm Central Time

---

1. Consider a random walk on a path with vertices numbered $1, 2, \ldots, n$ from left to right. At each step, we flip a coin to decide which direction to walk, moving one step left or one step right with equal probability. The random walk ends when we fall off one end of the path, either by moving left from vertex 1 or by moving right from vertex n.

   (a) Prove that if we start at vertex 1, the probability that the walk ends by falling off the *right* end of the path is exactly $1/(n+1)$.

   (b) Prove that if we start at vertex k, the probability that the walk ends by falling off the *right* end of the path is exactly $k/(n+1)$.

   (c) Prove that if we start at vertex 1, the expected number of steps before the random walk ends is exactly n.

   (d) What is the *exact* expected length of the random walk if we start at vertex k, as a function of n and k? Prove your result is correct. (For partial credit, give a tight Θ-bound for the case $k = (n+1)/2$, assuming n is odd.)

   Yes, "see part (b)" is worth full credit for part (a), but only if your solution to part (b) is correct. Same for parts (c) and (d). *[Hint: Trust the recursion fairy.]*

2. ***Tabulated hashing*** uses tables of random numbers to compute hash values. Suppose $|\mathcal{U}| = 2^w \times 2^w$ and $m = 2^\ell$, so the items being hashed are pairs of w-bit strings (or $2w$-bit strings broken in half) and hash values are ℓ-bit strings.

   Let $A[0..2^w-1]$ and $B[0..2^w-1]]$ be arrays of independent random ℓ-bit strings, and define the hash function $h_{A,B} : \mathcal{U} \to [m]$ by setting

   $$h_{A,B}(x, y) := A[x] \oplus B[y]$$

   where $\oplus$ denotes bit-wise exclusive-or. Let $\mathcal{H}$ denote the set of all possible functions $h_{A,B}$. Filling the arrays A and B with independent random bits is equivalent to choosing a hash function $h_{A,B} \in \mathcal{H}$ uniformly at random.

   (a) Prove that $\mathcal{H}$ is 2-uniform.

   (b) Prove that $\mathcal{H}$ is 3-uniform. *[Hint: Solve part (a) first.]*

   (c) Prove that $\mathcal{H}$ is ***not*** 4-uniform.

   Yes, "see part (b)" is worth full credit for part (a), but only if your solution to part (b) is correct.

3. Hashing is often used in distributed systems where we want to distribute tasks among a collection of servers. What happens when a server is added or removed from a system? For most hash functions, the hash function is tailored to the number of servers n, and would change completely if n changes. This would require rehashing and reassiging all of our m tasks.

   Here we consider an approach to avoid this problem. Assume (unrealistically) that we have access to an ***ideal random*** hash function that maps any value x to a real value $h(x) \in [0,1]$. Identifying each server and each task with a different point in the domain of the hash function, we use the hash function to map both tasks and servers randomly to the unit interval $[0,1]$. Now, each task is assigned to the first server to its right on the number line (with the interval wrapping around). When a new server is added to the system, we hash it to $[0,1]$ and move tasks to it accordingly.

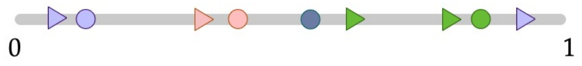

Figure 1: An example of the scheme to distribute tasks (triangles) to servers (circles). Each task is assigned to the server with matching color.

   (a) Suppose a new $(n+1)$-th server is added to the system. What is the expected number of tasks that need to be reassigned? Note that the expectation is taken with respect to the random positions of all the servers and all the tasks.

   (b) Show that, with high probability, no server "owns" more than an $O(\log n/n)$ fraction of the interval $[0,1]$.

   (c) Show that if we have n servers and m items where $m \geq 1000n$, the maximum load on any server is $O(\frac{m}{n} \cdot \log n)$ with high probability.[1]

[1]Recall that "with high probability" means with probability is at least $1 - 1/n^c$, for some constant $c \geq 1$.