Exercises

Exercises should be completed on your own.

1. (1 pt.) Suppose that $h : U \to \{0, \ldots, n-1\}$ is a uniformly random function. That is, $h(i)$ is distributed uniformly at random in the set $\{0, \ldots, n-1\}$ for all $i$, and $h(i)$ are independent for all $i$. Prove that for any $x \neq y \in U$,$$
abla \{h(x) = h(y)\} = \frac{1}{n}.
$$

[We are expecting: A short but rigorous proof.]

2. (2 pt.) This exercise references the IPython notebook HW4.ipynb as well as the files mysteryA.pickle and mysteryB.pickle.

In the IPython notebook, run the cells to load the hash families $A$ and $B$. Both $A$ and $B$ are lists of functions $h : \{0, \ldots, 21\} \to \{0, 1, 2\}$. As shown in the notebook, at first glance both of these seem like reasonable hash families. However, one of them is a universal hash family and one of them is not. Which one is which? Play around with both hash families in Python until you figure it out.

[We are expecting: Your answer, along with compelling numerical evidence (either numbers or a plot), and an explanation about why your numerical evidence is compelling and what it has to do with the definition of a universal hash family.]
Problems

You may talk with your fellow CS161-ers about the problems. However:

- Try the problems on your own before collaborating.
- Write up your answers yourself, in your own words. You should never share your typed-up solutions with your collaborators.
- If you collaborated, list the names of the students you collaborated with at the beginning of each problem.

1. (3 pt.) Your friend has a proposal for a new universal hash function $h : \mathcal{U} \rightarrow \{0, \ldots, n - 1\}$, where $\mathcal{U} = \{0, \ldots, n^2 - 1\}$. Your friend thinks that you can skip this whole “choose uniformly from a universal hash family” stuff, and just go with $h(x) = x \mod n$.

More precisely, your friend says, just take the hash family $\mathcal{H} = \{h\}$ to be the set with just this one function in it.

(a) (1 pt.) Your friend doesn’t have a very good track record on these homework sets, so you are dubious even before you hear their argument. Prove to your friend that their choice does not satisfy the key property of a universal hash family.

[We are expecting: A rigorous proof, using the definition of a universal hash family.]

(b) (1 pt.) Even given your proof, your friend plows on. Their first point:

Let $h = x \mod n$ be as above. If we choose $x \neq y$ uniformly at random\footnote{That is, choose $x$ uniformly at random from $\mathcal{U}$ and then choose $y$ uniformly at random from $\mathcal{U} \setminus \{x\}$} from $\mathcal{U}$, then $\mathbb{P}\{h(x) = h(y)\} \leq \frac{1}{n}$, where the probability is over the random choice of $x$ and $y$.

Do you agree?

[We are expecting: Whether the statement is true or false, and a convincing argument either way.]

(c) (1 pt.) Your friend continues:

Given the computation above, we have

$$\mathbb{P}\{h(x) = h(y)\} \leq \frac{1}{n}.$$ 

This is the definition of a universal hash family, so $\{h\}$ must be a universal hash family.

Do you agree?

[We are expecting: Whether this conclusion correctly follows from the statement in part (b), and a convincing argument either way.]
2. (5 pt.) Let $T$ be a Red-Black tree with root $x$. Let $T_L$ be the subtree rooted at $x$’s left child, and let $T_R$ be the subtree rooted at $x$’s right child.

\[
\begin{array}{c}
\text{x} \\
\text{y} & \text{z} \\
\text{T}_L & \text{T}_R
\end{array}
\]

Decide if each of the following statements are true or false. If it is true, give a proof. If it is false, give a counter-example.

(a) (2 pt.) In the set-up above, we must have

\[|T_L| \geq \frac{|T|}{2} - 1 \quad \text{and} \quad |T_R| \geq \frac{|T|}{2} - 1, \quad (1)\]

where $|T|$ denotes the number of nodes in $T$ (including the root, not including NIL nodes).

(b) (3 pt.) In the set-up above, we must have

\[|T_L| \geq \sqrt{\frac{|T|}{2}} - 1 \quad \text{and} \quad |T_R| \geq \sqrt{\frac{|T|}{2}} - 1, \quad (2)\]

where $|T|$ denotes the number of nodes in $T$ (including the root, not including NIL nodes).

[We are expecting: For each, either a rigorous proof, or an explicit counter-example.]

[HINT: You may use a claim that we proved in class.]
3. (6 pt.) A large flock of $T$ Colorful Geese will migrate south for the winter over the Gates building in the next few weeks. Colorful Geese are an interesting species. They can come in a huge number of colors—say, $M$ colors—but each flock only has $m$ colors represented, where $m < T$. You’d like to be able to answer queries about what colors of geese appeared in the flock. The birds will fly overhead one at a time, and after they have flown by they won’t come back again.

For example, if $T = 7$, $M = 100000$ and $m = 3$, then a flock of $T$ colorful geese might look like:

seabreeze, seabreeze, brick red, ultraviolet, brick red, ultraviolet, seabreeze

You’ll see this sequence in order, and only once. After the birds have gone, you’ll be asked questions like “How many brick red geese were there?” (Answer: 2), or “How many neon orange geese were there?” (Answer: 0).

You have access to a universal hash family $H$, so that each function $h \in H$ maps the set of $M$ colors into the set $\{0, \ldots, n-1\}$. For example, one function $h \in H$ might have $h(\text{seabreeze}) = 3$.

(a) (3 pt.) Suppose that $n = 10m$, and you only have space to store $n$ numbers in the set $\{0, \ldots, M-1\}$, as well as one function $h$ from $H$. Use the universal hash family $H$ to create a randomized data structure that fits in this space and that supports the following operations:

- **Update(color)**: Update the data structure when you see a good with color `color`.
- **Query(color)**: Return the number of geese of color `color` that you have seen so far. For each query, your query should be correct with probability at least $\frac{9}{10}$. That is, for all colors $i$,

$$\Pr\{\text{Query}(i) = \text{the number of geese with color } i\} \geq \frac{9}{10}.$$

You want each of these operations to be done in $O(1)$ time (in the worst case), assuming that you can evaluate a function $h \in H$ in $O(1)$ time.

[We are expecting: An explanation of how you will implement your operations, and a short but rigorous proof that your operations meet the requirements.]

(b) (3 pt.) Suppose that you now have ten times the space you had in part (a). Adapt your data structure from part (a) so that the Query operation is correct with probability $1 - \frac{1}{100}$.  

[We are expecting: An explanation of how you will implement your operations, and a short but rigorous proof that your operations meet the requirements.]