# COMPSCI 514: ALGORITHMS FOR DATA SCIENCE

Cameron Musco
University of Massachusetts Amherst. Fall 2021.
Lecture 3

- Sign up for Piazza.
- Remember to complete the quiz, released after class today and due **Monday at 8pm**.
- TA office hour schedules and locations have been posted the course website.

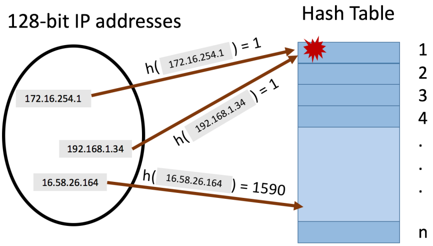
**Last Class We Covered:**

- Markov's inequality: the most fundamental concentration bound. $\Pr(X \geq t \cdot \mathbb{E}[X]) \leq 1/t$.
- Algorithmic applications of Markov's inequality, linearity of expectation, and indicator random variables:
  - Counting collisions to estimate CAPTCHA database size.
  - Counting collisions to understand the runtime of hash tables with random hash functions.

**Today:**

- Finish up random hash functions and hash tables.
- Learn about 2-level hashing.
- Learn about 2-universal and pairwise independent hash functions.
- Start on an application of random hashing to load balancing in distributed systems.
- Through this application learn about:
  - Chebyshev's inequality, which strengthens Markov's inequality.

We store $m$ items from a large universe in a hash table with $n$ positions.




- Want to show that when $\mathbf{h} : U \to [n]$ is a random hash function, query time is $O(1)$ with good probability.
- Equivalently: want to show that there are few collisions between hashed items.

When storing $m$ items in a table of size $n$, the expected number of pairwise collisions (two items stored in the same slots) is:

$$\mathbb{E}[\mathbf{C}] = \frac{m(m-1)}{2n}.$$

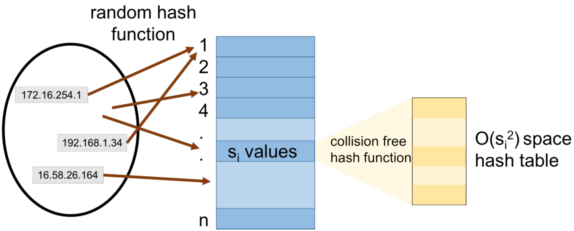
- For $n = 4m^2$ we have: $\mathbb{E}[\mathbf{C}] = \frac{m(m-1)}{8m^2} \leq \frac{1}{8}$.
- By Markov's inequality there no collisions with probability at least $\frac{7}{8}$.

$O(1)$ query time, but we are using $O(m^2)$ space to store $m$ items...

$m$: total number of stored items, $n$: hash table size, $\mathbf{C}$: total pairwise collisions in table.

Want to preserve $O(1)$ query time while using $O(m)$ space.

## Two-Level Hashing:




- For each bucket with $s_i$ values, pick a collision free hash function mapping $[s_i] \to [s_i^2]$.
- **Just Showed:** A random function is collision free with probability $\geq \frac{7}{8}$ so can just generate a random hash function and check if it is collision free.

Query time for two level hashing is $O(1)$: requires evaluating two hash functions. What is the expected space usage?

Up to constants, space used is: $\mathbf{S} = n + \sum_{i=1}^n \mathbf{s}_i^2 \mathbb{E}[\mathbf{S}] = n + \sum_{i=1}^n \mathbb{E}[\mathbf{s}_i^2]$

$$\mathbb{E}[\mathbf{s}_i^2] = \mathbb{E}\left[\left(\sum_{j=1}^{m} \mathbb{I}_{\mathbf{h}(x_j)=i}\right)^2\right]$$

$$= \mathbb{E}\left[\sum_{j,k\in[m]} \mathbb{I}_{\mathbf{h}(x_j)=i} \cdot \mathbb{I}_{\mathbf{h}(x_k)=i}\right] = \sum_{j,k\in[m]} \mathbb{E}\left[\mathbb{I}_{\mathbf{h}(x_j)=i} \cdot \mathbb{I}_{\mathbf{h}(x_k)=i}\right].$$

Collisions again!

- For $j = k$, $\mathbb{E}\left[\mathbb{I}_{\mathbf{h}(x_j)=i} \cdot \mathbb{I}_{\mathbf{h}(x_k)=i}\right] = \mathbb{E}\left[\left(\mathbb{I}_{\mathbf{h}(x_j)=i}\right)^2\right] = \Pr[\mathbf{h}(x_j) = i] = \frac{1}{n}$.
- For $j \neq k$, $\mathbb{E}\left[\mathbb{I}_{\mathbf{h}(x_j)=i} \cdot \mathbb{I}_{\mathbf{h}(x_k)=i}\right] = \Pr[\mathbf{h}(x_j) = i \cap \mathbf{h}(x_k) = i] = \frac{1}{n^2}$.

$x_j, x_k$: stored items, $n$: hash table size, $\mathbf{h}$: random hash function, $\mathbf{S}$: space usage of two level hashing, $\mathbf{s}_i$: # items stored in hash table at position $i$.

$$\mathbb{E}[\mathbf{s}_i^2] = \sum_{j,k \in [m]} \mathbb{E}\left[\mathbb{I}_{\mathbf{h}(x_j)=i} \cdot \mathbb{I}_{\mathbf{h}(x_k)=i}\right]$$
$$= m \cdot \frac{1}{n} + 2 \cdot \binom{m}{2} \cdot \frac{1}{n^2}$$
$$= \frac{m}{n} + \frac{m(m-1)}{n^2} \leq 2 \text{ (If we set } n = m.)$$

- For $j = k$, $\mathbb{E}\left[\mathbb{I}_{\mathbf{h}(x_j)=i} \cdot \mathbb{I}_{\mathbf{h}(x_k)=i}\right] = \frac{1}{n}$.
- For $j \neq k$, $\mathbb{E}\left[\mathbb{I}_{\mathbf{h}(x_j)=i} \cdot \mathbb{I}_{\mathbf{h}(x_k)=i}\right] = \frac{1}{n^2}$.

**Total Expected Space Usage:** (if we set $n = m$)

$$\mathbb{E}[\mathbf{S}] = n + \sum_{i=1}^{n} \mathbb{E}[\mathbf{s}_i^2] \leq n + n \cdot 2 = 3n = 3m.$$

Near optimal space with $O(1)$ query time!

$x_j, x_k$: stored items, $m$: # stored items, $n$: hash table size, $\mathbf{h}$: random hash function, $\mathbf{S}$: space usage of two level hashing, $\mathbf{s}_i$: # items stored at pos $i$.

**So Far:** we have assumed a **fully random hash function** $\mathbf{h}(x)$ with $\Pr[\mathbf{h}(x) = i] = \frac{1}{n}$ for $i \in 1, \ldots, n$ and $\mathbf{h}(x), \mathbf{h}(y)$ independent for $x \neq y$.

- To compute a random hash function we have to store a table of $x$ values and their hash values. Would take at least $O(m)$ space and $O(m)$ query time to look up $\mathbf{h}(x)$ if we hash $m$ values. Making our whole quest for $O(1)$ query time pointless!

| **x** | **h(x)** |
|---|---|
| $x_1$ | 45 |
| $x_2$ | 1004 |
| $x_3$ | 10 |
| $\vdots$ | $\vdots$ |
| $x_m$ | 12 |

What properties did we use of the randomly chosen hash function?

**2-Universal Hash Function** (low collision probability). A random hash function from $\mathbf{h}: U \to [n]$ is two universal if:

$$\Pr[\mathbf{h}(x) = \mathbf{h}(y)] \leq \frac{1}{n}.$$

**Exercise:** Rework the two level hashing proof to show that this property is really all that is needed.

When $\mathbf{h}(x)$ and $\mathbf{h}(y)$ are chosen independently at random from $[n]$, $\Pr[\mathbf{h}(x) = \mathbf{h}(y)] = \frac{1}{n}$ (so a fully random hash function is 2-universal)

**Efficient Alternative:** Let $p$ be a prime with $p \geq |U|$. Choose random $\mathbf{a}, \mathbf{b} \in [p]$ with $\mathbf{a} \neq 0$. Represent $x$ an an integer and let

$$\mathbf{h}(x) = (\mathbf{a}x + \mathbf{b} \mod p) \mod n.$$

Another common requirement for a hash function:

**Pairwise Independent Hash Function.** A random hash function from $\mathbf{h} : U \to [n]$ is pairwise independent if for all $i, j \in [n]$:

$$\Pr[\mathbf{h}(x) = i \cap \mathbf{h}(y) = j] = \frac{1}{n^2}.$$

**Think-Pair-Shair:** Which is a more stringent requirement? 2-universal or pairwise independent?

$$\Pr[\mathbf{h}(x) = \mathbf{h}(y)] = \sum_{i=1}^{n} \Pr[\mathbf{h}(x) = i \cap \mathbf{h}(y) = i] = n \cdot \frac{1}{n^2} = \frac{1}{n}.$$

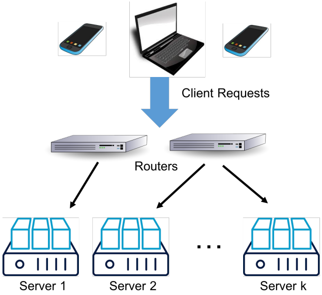
A closely related $(\mathbf{a}x + \mathbf{b}) \mod p$ construction gives pairwise independence on top of 2-universality.

**Remember:** A fully random hash function is both 2-universal and pairwise independent. But it is not efficiently implementable.

# Questions on Hash Tables?

1. We'll consider an application where our toolkit of linearity of expectation + Markov's inequality doesn't give much.
2. Then we'll show how a simple twist on Markov's can give a much stronger result.

Randomized Load Balancing:




**Simple Model:** $n$ requests randomly assigned to $k$ servers. How many requests must each server handle?

- Often assignment is done via a random hash function. Why?