

# 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:

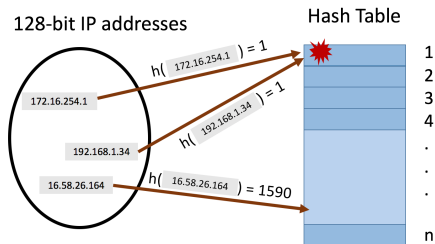
- Markov's inequality: the most fundamental **concentration bound**.  $\Pr(X \geq t \cdot \mathbb{E}[X]) \leq 1/t$ .
- Algorithmic applications of Markov's inequality, linearity of expectation, and indicator random variables:
  - Counting collisions to estimate CAPTCHA database size.
  - Counting collisions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

# HASH TABLES

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



- Want to show that when  $h : U \rightarrow [n]$  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.

## COLLISION FREE HASHING

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

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

$O(m^2) = O(\sum_i 1)$

- For  $n = \underline{4m^2}$  we have:  $\mathbb{E}[C] = \frac{m(m-1)}{8m^2} \leq \frac{1}{8}$ .
- 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...

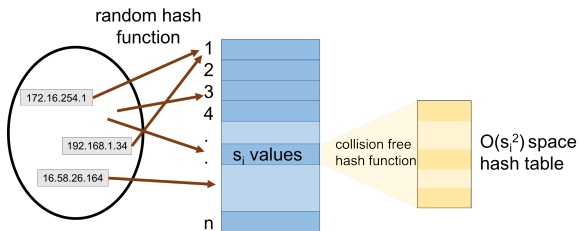
$m$ : total number of stored items,  $n$ : hash table size,  $C$ : total pairwise collisions in table.

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

# TWO LEVEL HASHING

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

Two-Level Hashing:

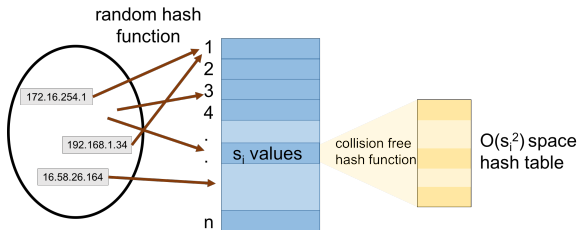




# TWO LEVEL HASHING

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

## Two-Level Hashing:

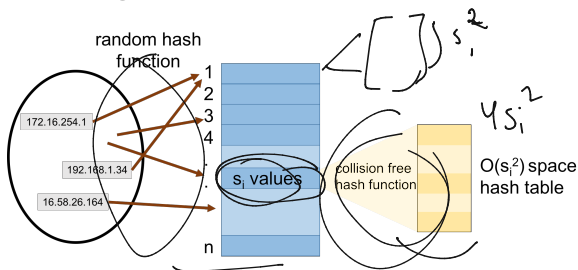


- For each bucket with  $s_i$  values, pick a collision free hash function mapping  $[s_i] \rightarrow [s_i^2]$ .

# TWO LEVEL HASHING

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  $[s_i] \rightarrow [s_i^2]$ .
- **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.

## SPACE USAGE

Query time for two level hashing is  $O(1)$ : requires evaluating two hash functions.

$O(m)$

$x_j, x_R$ : stored items,  $n$ : hash table size,  $h$ : random hash function,  $S$ : space usage of two level hashing,  $s_i$ : # items stored in hash table at position  $i$ .

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

$x_j, x_R$ : stored items,  $n$ : hash table size,  $h$ : random hash function,  $S$ : space usage of two level hashing,  $s_i$ : # items stored in hash table at position  $i$ .

## SPACE USAGE

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:  $\underline{S = n + \sum_{i=1}^n s_i^2}$

$x_j, x_R$ : stored items,  $n$ : hash table size,  $h$ : random hash function,  $S$ : space usage of two level hashing,  $s_i$ : # items stored in hash table at position  $i$ .

## SPACE USAGE

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:  $\mathbb{E}[S] = n + \sum_{i=1}^n \mathbb{E}[s_i^2]$

$x_j, x_R$ : stored items,  $n$ : hash table size,  $h$ : random hash function,  $S$ : space usage of two level hashing,  $s_i$ : # items stored in hash table at position  $i$ .

## SPACE USAGE

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:  $\mathbb{E}[\mathbf{S}] = n + \sum_{i=1}^n \mathbb{E}[\mathbf{s}_i^2]$

$x_j, x_R$ : 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  $i$ .

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:  $\mathbb{E}[S] = n + \sum_{i=1}^n \mathbb{E}[s_i^2]$

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

$$\mathbb{I}_{h(x_j)=i} = \begin{cases} 1 & \text{if } h(x_j) = i \\ 0 & \text{if } h(x_j) \neq i \end{cases}$$

$$s_i = \sum_{j=1}^m \mathbb{I}_{h(x_j)=i}$$

$x_j, x_k$ : stored items,  $n$ : hash table size,  $h$ : random hash function,  $S$ : space usage of two level hashing,  $s_i$ : # items stored in hash table at position  $i$ .



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:  $\mathbb{E}[S] = n + \sum_{i=1}^n \mathbb{E}[s_i^2]$

$$\begin{aligned} \mathbb{E}[s_i^2] &= \mathbb{E} \left[ \left( \sum_{j=1}^m \mathbb{I}_{h(x_j)=i} \right)^2 \right] && (a+b+c)^2 \\ &= \mathbb{E} \left[ \sum_{j,k \in [m]} \underbrace{\mathbb{I}_{h(x_j)=i} \cdot \mathbb{I}_{h(x_k)=i}}_{\text{Collisions again!}} \right] && (a:a + a \cdot b + b \cdot a + a \cdot c + c \cdot a) \end{aligned}$$

$x_j, x_k$ : stored items,  $n$ : hash table size,  $h$ : random hash function,  $S$ : space usage of two level hashing,  $s_i$ : # items stored in hash table at position  $i$ .

## SPACE USAGE

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:  $\mathbb{E}[S] = n + \sum_{i=1}^n \mathbb{E}[s_i^2]$

*space used by i-th bucket hash table*

$$\begin{aligned} \mathbb{E}[s_i^2] &= \mathbb{E} \left[ \left( \sum_{j=1}^m \mathbb{I}_{h(x_j)=i} \right)^2 \right] \\ &= \mathbb{E} \left[ \sum_{j,k \in [m]} \mathbb{I}_{h(x_j)=i} \cdot \mathbb{I}_{h(x_k)=i} \right] = \sum_{j,k \in [m]} \mathbb{E} \left[ \mathbb{I}_{h(x_j)=i} \cdot \mathbb{I}_{h(x_k)=i} \right]. \end{aligned}$$

$x_j, x_k$ : stored items,  $n$ : hash table size,  $h$ : random hash function,  $S$ : space usage of two level hashing,  $s_i$ : # items stored in hash table at position  $i$ .

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:  $\mathbb{E}[\mathbf{S}] = n + \sum_{i=1}^n \mathbb{E}[\mathbf{s}_i^2]$

$$\begin{aligned} \mathbb{E}[\mathbf{s}_i^2] &= \mathbb{E} \left[ \left( \sum_{j=1}^m \mathbb{I}_{\mathbf{h}(x_j)=i} \right)^2 \right] \\ &= \mathbb{E} \left[ \sum_{j,k \in [m]} \mathbb{I}_{\mathbf{h}(x_j)=i} \cdot \mathbb{I}_{\mathbf{h}(x_k)=i} \right] = \sum_{j,k \in [m]} \mathbb{E} \left[ \underbrace{\mathbb{I}_{\mathbf{h}(x_j)=i}} \cdot \underbrace{\mathbb{I}_{\mathbf{h}(x_k)=i}} \right]. \end{aligned}$$

- For  $j = k$ ,

$x_j, x_k$ : 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  $i$ .

## SPACE USAGE

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:  $\mathbb{E}[S] = n + \sum_{i=1}^n \mathbb{E}[s_i^2]$

$$\begin{aligned}\mathbb{E}[s_i^2] &= \mathbb{E} \left[ \left( \sum_{j=1}^m \mathbb{I}_{h(x_j)=i} \right)^2 \right] \\ &= \mathbb{E} \left[ \sum_{j,k \in [m]} \mathbb{I}_{h(x_j)=i} \cdot \mathbb{I}_{h(x_k)=i} \right] = \sum_{j,k \in [m]} \mathbb{E} \left[ \mathbb{I}_{h(x_j)=i} \cdot \mathbb{I}_{h(x_k)=i} \right].\end{aligned}$$

$$\cdot \text{ For } j = k, \mathbb{E} \left[ \mathbb{I}_{h(x_j)=i} \cdot \mathbb{I}_{h(x_k)=i} \right] = \mathbb{E} \left[ \left( \mathbb{I}_{h(x_j)=i} \right)^2 \right] = \mathbb{E} \left[ \mathbb{I}_{h(x_j)=i} \right]$$

$x_j, x_k$ : stored items,  $n$ : hash table size,  $h$ : random hash function,  $S$ : space usage of two level hashing,  $s_i$ : # items stored in hash table at position  $i$ .

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:  $\mathbb{E}[\mathbf{S}] = n + \sum_{i=1}^n \mathbb{E}[\mathbf{s}_i^2]$

$$\begin{aligned} \mathbb{E}[\mathbf{s}_i^2] &= \mathbb{E} \left[ \left( \sum_{j=1}^m \mathbb{I}_{\mathbf{h}(x_j)=i} \right)^2 \right] \\ &= \mathbb{E} \left[ \sum_{j,k \in [m]} \mathbb{I}_{\mathbf{h}(x_j)=i} \cdot \mathbb{I}_{\mathbf{h}(x_k)=i} \right] = \sum_{j,k \in [m]} \mathbb{E} \left[ \mathbb{I}_{\mathbf{h}(x_j)=i} \cdot \mathbb{I}_{\mathbf{h}(x_k)=i} \right]. \end{aligned}$$

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

$x_j, x_k$ : 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  $i$ .

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:  $\mathbb{E}[\mathbf{S}] = n + \sum_{i=1}^n \mathbb{E}[s_i^2]$

$$\begin{aligned} \mathbb{E}[s_i^2] &= \mathbb{E} \left[ \left( \sum_{j=1}^m \mathbb{I}_{\mathbf{h}(x_j)=i} \right)^2 \right] \\ &= \mathbb{E} \left[ \sum_{j,k \in [m]} \mathbb{I}_{\mathbf{h}(x_j)=i} \cdot \mathbb{I}_{\mathbf{h}(x_k)=i} \right] = \sum_{j,k \in [m]} \mathbb{E} \left[ \mathbb{I}_{\mathbf{h}(x_j)=i} \cdot \mathbb{I}_{\mathbf{h}(x_k)=i} \right]. \end{aligned}$$

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

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

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:  $\mathbb{E}[\mathbf{S}] = n + \sum_{i=1}^n \mathbb{E}[s_i^2]$

$$\begin{aligned}\mathbb{E}[s_i^2] &= \mathbb{E} \left[ \left( \sum_{j=1}^m \mathbb{I}_{\mathbf{h}(x_j)=i} \right)^2 \right] \\ &= \mathbb{E} \left[ \sum_{j,k \in [m]} \mathbb{I}_{\mathbf{h}(x_j)=i} \cdot \mathbb{I}_{\mathbf{h}(x_k)=i} \right] = \sum_{j,k \in [m]} \mathbb{E} \left[ \mathbb{I}_{\mathbf{h}(x_j)=i} \cdot \mathbb{I}_{\mathbf{h}(x_k)=i} \right].\end{aligned}$$

- 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  $j \neq k$ ,

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

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:  $\mathbb{E}[S] = n + \sum_{i=1}^n \mathbb{E}[s_i^2]$

$$\begin{aligned} \mathbb{E}[s_i^2] &= \mathbb{E} \left[ \left( \sum_{j=1}^m \mathbb{I}_{\mathbf{h}(x_j)=i} \right)^2 \right] \\ &= \mathbb{E} \left[ \sum_{j,k \in [m]} \mathbb{I}_{\mathbf{h}(x_j)=i} \cdot \mathbb{I}_{\mathbf{h}(x_k)=i} \right] = \sum_{j,k \in [m]} \mathbb{E} \left[ \mathbb{I}_{\mathbf{h}(x_j)=i} \cdot \mathbb{I}_{\mathbf{h}(x_k)=i} \right]. \end{aligned}$$

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

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



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:  $\mathbb{E}[S] = n + \sum_{i=1}^n \mathbb{E}[s_i^2]$

$$\begin{aligned} \mathbb{E}[s_i^2] &= \mathbb{E} \left[ \left( \sum_{j=1}^m \mathbb{I}_{\mathbf{h}(x_j)=i} \right)^2 \right] \\ &= \mathbb{E} \left[ \sum_{j,k \in [m]} \mathbb{I}_{\mathbf{h}(x_j)=i} \cdot \mathbb{I}_{\mathbf{h}(x_k)=i} \right] = \sum_{j,k \in [m]} \mathbb{E} \left[ \mathbb{I}_{\mathbf{h}(x_j)=i} \cdot \mathbb{I}_{\mathbf{h}(x_k)=i} \right]. \end{aligned}$$

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

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

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:  $\mathbb{E}[\mathbf{S}] = n + \sum_{i=1}^n \mathbb{E}[\mathbf{s}_i^2]$

$$\begin{aligned} \mathbb{E}[\mathbf{s}_i^2] &= \mathbb{E} \left[ \left( \sum_{j=1}^m \mathbb{I}_{\mathbf{h}(x_j)=i} \right)^2 \right] \\ &= \mathbb{E} \left[ \sum_{j,k \in [m]} \mathbb{I}_{\mathbf{h}(x_j)=i} \cdot \mathbb{I}_{\mathbf{h}(x_k)=i} \right] = \sum_{j,k \in [m]} \mathbb{E} \left[ \mathbb{I}_{\mathbf{h}(x_j)=i} \cdot \mathbb{I}_{\mathbf{h}(x_k)=i} \right]. \end{aligned}$$

- 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  $i$ .

$$\mathbb{E}[s_i^2] = \sum_{j,k \in [m]} \mathbb{E} \left[ \mathbb{I}_{\mathbf{h}(x_j)=i} \cdot \mathbb{I}_{\mathbf{h}(x_k)=i} \right]$$

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

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

$$\begin{aligned}\mathbb{E}[s_i^2] &= \sum_{j,k \in [m]} \mathbb{E} \left[ \mathbb{I}_{\mathbf{h}(x_j)=i} \cdot \mathbb{I}_{\mathbf{h}(x_k)=i} \right] \\ &= m \cdot \frac{1}{n} + 2 \cdot \binom{m}{2} \cdot \frac{1}{n^2}\end{aligned}$$

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

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

$$\begin{aligned}\mathbb{E}[s_i^2] &= \sum_{j,k \in [m]} \mathbb{E} \left[ \mathbb{I}_{h(x_j)=i} \cdot \mathbb{I}_{h(x_k)=i} \right] \\ &= m \cdot \frac{1}{n} + 2 \cdot \binom{m}{2} \cdot \frac{1}{n^2}\end{aligned}$$

$$\begin{aligned}(a+b+c)^2 \\ (a^2 + b^2 + c^2 + \dots \\ ab + ba\end{aligned}$$

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

$x_j, x_k$ : stored items,  $m$ : # stored items,  $n$ : hash table size,  $h$ : random hash function,  $S$ : space usage of two level hashing,  $s_i$ : # items stored at pos  $i$ .

$$\begin{aligned}\mathbb{E}[s_i^2] &= \sum_{j,k \in [m]} \mathbb{E} \left[ \mathbb{I}_{h(x_j)=i} \cdot \mathbb{I}_{h(x_k)=i} \right] \\ &= m \cdot \frac{1}{n} + 2 \cdot \binom{m}{2} \cdot \frac{1}{n^2}\end{aligned}$$

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

$x_j, x_k$ : stored items,  $m$ : # stored items,  $n$ : hash table size,  $h$ : random hash function,  $S$ : space usage of two level hashing,  $s_i$ : # items stored at pos  $i$ .

$$\begin{aligned}
 \mathbb{E}[s_i^2] &= \sum_{j,k \in [m]} \mathbb{E} \left[ \mathbb{I}_{h(x_j)=i} \cdot \mathbb{I}_{h(x_k)=i} \right] && n = m \\
 &= m \cdot \frac{1}{n} + 2 \cdot \binom{m}{2} \cdot \frac{1}{n^2} \\
 &= \frac{m}{n} + \frac{m(m-1)}{n^2} = \frac{m}{m} + \frac{m(m-1)}{m^2} \leq 2
 \end{aligned}$$

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

$x_j, x_k$ : stored items,  $m$ : # stored items,  $n$ : hash table size,  $h$ : random hash function,  $S$ : space usage of two level hashing,  $s_i$ : # items stored at pos  $i$ .

## SPACE USAGE

items are:  $x_1, x_2, \dots, x_m$

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

$x_j, x_k$  are any 2 items were interv.

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

$x_j, x_k$ : stored items,  $m$ : # stored items,  $n$ : hash table size,  $h$ : random hash function,  $S$ : space usage of two level hashing,  $s_i$ : # items stored at pos  $i$ .



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

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

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

$$\underbrace{\mathbb{E}[\mathbf{S}]} = \underbrace{n} + \underbrace{\sum_{i=1}^n \mathbb{E}[s_i^2]}$$

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

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

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

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

$$\mathbb{E}[S] = n + \sum_{i=1}^n \mathbb{E}[s_i^2] \leq n + \underbrace{n \cdot 2} = \underbrace{3n} \neq 3m.$$

$x_j, x_k$ : stored items,  $m$ : # stored items,  $n$ : hash table size,  $h$ : random hash function,  $S$ : space usage of two level hashing,  $s_i$ : # items stored at pos  $i$ .

# SPACE USAGE

$$\binom{m}{2} \cdot \frac{m(m-1)}{2}$$

$$\mathbb{E}[s_i^2] = \sum_{j,k \in [m]} \mathbb{E}[\mathbb{I}_{h(x_j)=i} \cdot \mathbb{I}_{h(x_k)=i}]$$

$$= m \cdot \frac{1}{n} + 2 \cdot \binom{m}{2} \cdot \frac{1}{n^2}$$

$$= \frac{m}{n} + \frac{m(m-1)}{n^2} \leq 2 \text{ (if we set } n = m.)$$

cuckoo hashing  
 double hashing

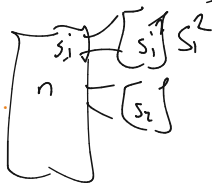
$$(I_1 + I_2 + \dots + I_m)^2$$

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

$$(I_1^2 + \dots + I_m^2 + 2I_{12} + 2I_{13} + \dots)$$

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

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



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

$x_j, x_k$ : stored items,  $m$ : # stored items,  $n$ : hash table size,  $h$ : random hash function,  $S$ : space usage of two level hashing,  $s_i$ : # items stored at pos  $i$ .

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

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

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

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

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

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

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

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

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

$$\mathbb{E} \sum_j \sum_k \mathbb{1}_{j \neq k} = \frac{1}{n^2}$$



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

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

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

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

$\forall x, y$  such that  $x \neq y$ .

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

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

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

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

$$h(x) = (\underline{ax + b} \pmod{p}) \pmod{n}.$$

Another common requirement for a hash function:

Another common requirement for a hash function:

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

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

$$\Pr(h(x) = i) \cdot \Pr(h(y) = j) = \frac{1}{n} \cdot \frac{1}{n} = \frac{1}{n^2}$$

## PAIRWISE INDEPENDENCE

Another common requirement for a hash function:  $\Pr(h(x) = h(y) = i) = \frac{1}{n^2}$

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

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

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

2-Universal: A hash function is 2-universal  
if  $\forall x, y \quad \Pr(h(x) = h(y)) \leq \frac{1}{n}$

Another common requirement for a hash function:

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

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

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

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

prob. of collision  
assuming  $h$  is PWI

Another common requirement for a hash function:

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

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

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

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

A closely related  $(ax + b) \bmod p$  construction gives pairwise independence on top of 2-universality.

Another common requirement for a hash function:

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

$\forall x, y$  such that  $x \neq y$ .

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

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

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

A closely related  $(ax + b) \bmod p$  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.

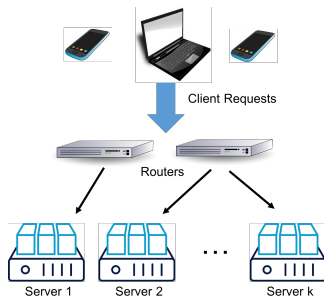


Questions on Hash Tables?

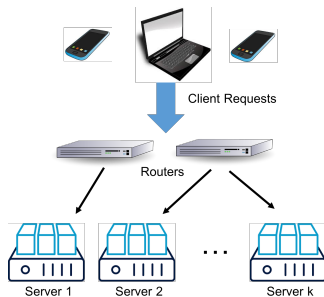
1. We'll consider an application where our toolkit of linearity of expectation + Markov's inequality doesn't give much.

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:

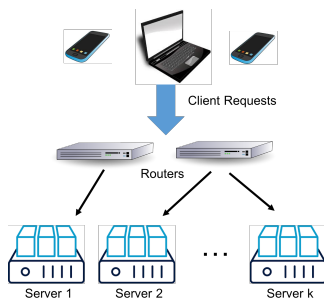


## Randomized Load Balancing:



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

## 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?