COMPSCI 514: ALGORITHMS FOR DATA SCIENCE

Cameron Musco

University of Massachusetts Amherst. Fall 2020.

Lecture 3

By Thursday:

- Sign up for Piazza.
- Sign up for Gradescope (code on class website) and fill out the Gradescope consent poll on Piazza. Contact me via email if you don't consent to use Gradescope.

By Thursday:

- Sign up for Piazza.
- Sign up for Gradescope (code on class website) and fill out the Gradescope consent poll on Piazza. Contact me via email if you don't consent to use Gradescope.

First Problem Set: released Saturday, due 9/11 at 8pm in Gradescope.

- Remember you can complete in a group of up to 3 students, who all turn in one submission with three names on it.

WEEK 1 QUIZ

91 students completed the quizzes – make sure that if you are enrolled you are doing the quiz each week.

91 students completed the quizzes – make sure that if you are enrolled you are doing the quiz each week.

Question 1: The expected number of inches of rain on Saturday is 2 and the expected number of inches on Sunday is 6. There is a 50% chance of rain on Saturday. If it rains on Saturday, there is a 75% chance of rain on Sunday. If it does not rain on Saturday, there is only a 25% chance of rain on Sunday. What is the expected number of inches of rainfall total over the weekend?

$$\mathbb{E}[R_{sat} + R_{sun}] = \mathbb{E}R_{sat} + \mathbb{E}R_{sun} = 2+6 = 8$$

91 students completed the quizzes – make sure that if you are enrolled you are doing the quiz each week.

Question 1: The expected number of inches of rain on Saturday is 2 and the expected number of inches on Sunday is 6. There is a 50% chance of rain on Saturday. If it rains on Saturday, there is a 75% chance of rain on Sunday. If it does not rain on Saturday, there is only a 25% chance of rain on Sunday. What is the expected number of inches of rainfall total over the weekend?

Concerns: Probability/linear algebra background, proofs/derivations.

Last Class We Covered:

- Markov's inequality: the most fundamental concentration bound.
- 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.

$$\frac{70^2}{2 \cdot 365} = 7$$

Last Class We Covered:

- Markov's inequality: the most fundamental concentration bound.
- 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.
- Collision counting is closely related to the birthday paradox.

$$\mathbb{E}[C] = \frac{\binom{m}{2}}{n} \approx \frac{m^2}{2n}$$

m = # people in room

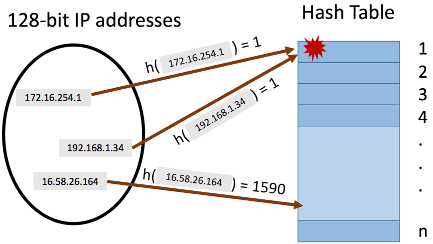
$n = 365$

Today:

- Finish up random hash functions and hash tables.
- See an applications of random hashing to load balancing in distributed systems.
- Through these applications learn about:
  - Chebyshev's inequality, which strengthens Markov's inequality.
  - The union bound, for understanding the probabilities of correlated random events.

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

insert
deletion
query



- 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:

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

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

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 $\frac{7}{8}$.

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

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 $\frac{7}{8}$.

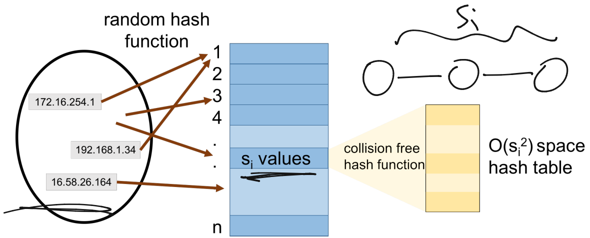
$O(1)$ query time, but we are using $O(m^2)$ space to store m items...

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.

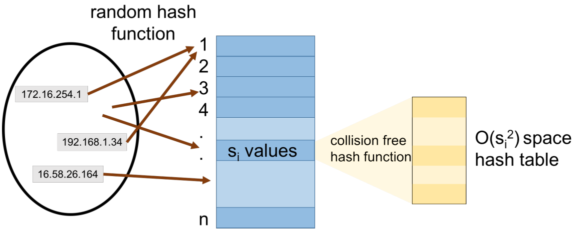
Want to preserve $O(1)$ query time while using $O(m)$ space.

Two-Level Hashing:



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

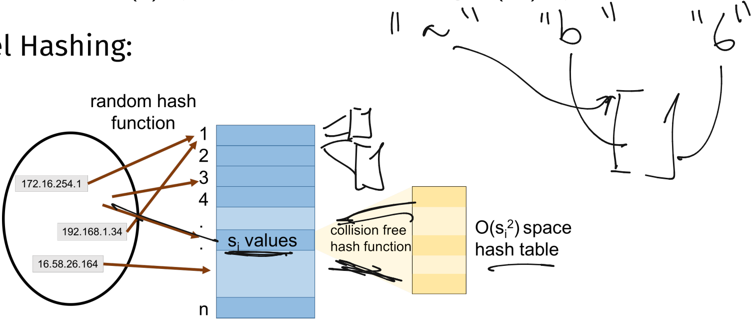
Two-Level Hashing:



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

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.

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?

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?

"static hashing"

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

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.

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

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.

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

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

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

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

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

Collisions again!

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

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

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

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

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

$$(I_1 + I_2 + \ldots I_m)(I_1 + I_2 + \ldots I_m)$$

$$I_1^2 + I_1 I_2 + I_2 \cdot I_1$$

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

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

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

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

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

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

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

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

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

  $= \mathbb{E}[\mathbb{I}_{h(x_j)=i}]$ //

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.

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

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

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

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

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

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

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

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

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

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

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

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?

$s_i = \#$ items at position i

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

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

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

X_1 X_2 X_3 X_4

0 1 1 0

- 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.

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

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

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

- 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.

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

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

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.

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

- 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, $\mathbf{S}$: space usage of two level hashing, $\mathbf{s}_i$: # items stored at pos i.

SPACE USAGE

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

- 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, $\mathbf{S}$: space usage of two level hashing, $\mathbf{s}_i$: # items stored at pos i.

SPACE USAGE

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

- 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, $\mathbf{S}$: space usage of two level hashing, $\mathbf{s}_i$: # items stored at pos i.

SPACE USAGE

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

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

$$= \frac{m}{n} + \frac{m(m-1)}{n^2}$$

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

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

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

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

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

$$\frac{m}{m} + \frac{m(m-1)}{m^2}$$

$$1 + \leq 1 \leq 2$$

- 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, $\mathbf{S}$: space usage of two level hashing, $\mathbf{s}_i$: # items stored at pos i.

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

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

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

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

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

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

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

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

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

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.

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

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

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

- 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.

EFFICIENTLY COMPUTABLE HASH FUNCTION

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 $x \neq y$.

EFFICIENTLY COMPUTABLE HASH FUNCTION

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 if we hash m values. Making our whole quest for $O(1)$ query time pointless!

trie

x	**h(x)**
x_1	45
x_2	1004
x_3	10
⋮	⋮
x_m	12

"abc" = 1365

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 $\mathbf{h} : U \to [n]$ is two universal if:

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

For fully random hash function: $\frac{1}{n}$

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

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

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)

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:

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

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 $p \ge |U|$. Choose random $\mathbf{a}, \mathbf{b} \in [p]$ with $\mathbf{a} \neq 0$. Let:

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

Another common requirement for a hash function:

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 $i, j \in [n]$:

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

Another common requirement for a hash function:

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

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

Breakout: Which is a more stringent requirement? 2-universal or pairwise independent?

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 $i, j \in [n]$:

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

Breakout: Which is a more stringent requirement? 2-universal or pairwise independent?

pairwise ⇒ 2-universal

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

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 $i, j \in [n]$:

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

Breakout: 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.

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 $i, j \in [n]$:

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

Breakout: 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.

NEXT STEP

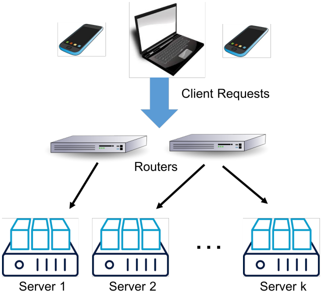
1. We'll consider an application where our toolkit of linearity of expectation + Markov's inequality doesn't give much.

NEXT STEP

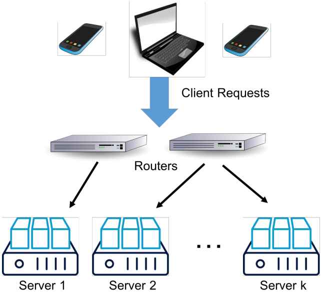
1. We'll consider an application where our toolkit of linearity of expectation + Markov's inequality doesn't give much.

2. Then we'll show how a simple twist on Markov's can give a much stronger result.

Randomized Load Balancing:

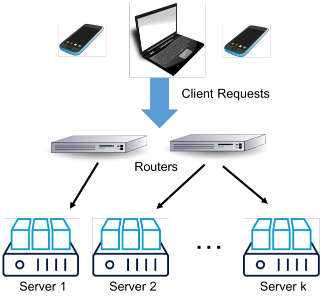


Randomized Load Balancing:



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?

Expected Number of requests assigned to server i:

$$\mathbb{E}[\mathsf{R}_i] = \sum_{j=1}^{n} \mathbb{E}[\mathbb{I}_{\text{request } j \text{ assigned to } i}] = \sum_{j=1}^{n} \Pr[j \text{ assigned to } i] = \frac{n}{k}.$$

n: total number of requests, k: number of servers randomly assigned requests, R_i: number of requests assigned to server i.

Expected Number of requests assigned to server i:

$$\mathbb{E}[R_i] = \sum_{j=1}^{n} \mathbb{E}[\mathbb{I}_{\text{request } j \text{ assigned to } i}] = \sum_{j=1}^{n} \Pr[j \text{ assigned to } i] = \frac{n}{k}.$$

If we provision each server be able to handle twice the expected load, what is the probability that a server is overloaded?

$\frac{2n}{k}$

n: total number of requests, k: number of servers randomly assigned requests, R_i: number of requests assigned to server i.

Expected Number of requests assigned to server i:

$$\mathbb{E}[R_i] = \sum_{j=1}^{n} \mathbb{E}[\mathbb{I}_{\text{request } j \text{ assigned to } i}] = \sum_{j=1}^{n} \Pr[j \text{ assigned to } i] = \frac{n}{k}.$$

If we provision each server be able to handle twice the expected load, what is the probability that a server is overloaded?

Applying Markov's Inequality

$$\Pr[R_i \geq 2\mathbb{E}[R_i]] \leq \frac{\mathbb{E}[R_i]}{2\mathbb{E}[R_i]} = \frac{1}{2}.$$

n: total number of requests, k: number of servers randomly assigned requests, R_i: number of requests assigned to server i.

WEAKNESS OF MARKOV'S

Expected Number of requests assigned to server *i*:

$$\mathbb{E}[\mathrm{R}_i] = \sum_{j=1}^{n} \mathbb{E}[\mathbb{I}_{\text{request } j \text{ assigned to } i}] = \sum_{j=1}^{n} \Pr[j \text{ assigned to } i] = \frac{n}{k}.$$

If we provision each server be able to handle twice the expected load, what is the probability that a server is overloaded?

Applying Markov's Inequality

$$\Pr[\mathrm{R}_i \geq 2\mathbb{E}[\mathrm{R}_i]] \leq \frac{\mathbb{E}[\mathrm{R}_i]}{2\mathbb{E}[\mathrm{R}_i]} = \frac{1}{2}.$$

Not great...half the servers may be overloaded.

> n: total number of requests, k: number of servers randomly assigned requests, R_i: number of requests assigned to server i.