COMPSCI 514: Algorithms for Data Science

Cameron Musco

University of Massachusetts Amherst. Fall 2026.

Lecture 2

In-Class Quiz 1

Question 1: Let Z be a random variable that takes values $\{-1, 0, 1\}$ each with probability $1/3$. What is $\mathbb{E}[Z]$? What is $\mathrm{Var}[Z]$? Show your calculations.

Question 2: Show that for any random variable X, $\mathrm{Var}[X] = \mathbb{E}[X^2] - \mathbb{E}[X]^2$.

Logistics

Logistics:

- Remember to sign up for Piazza.
- Find problem set teammates (see Piazza Post) and sign up for Gradescope (code on course website).
- You do not need to complete the problem sets in groups.
- Pset 1 will be released sometime this week. Deadlines on course website are approximate and will adjust as we go.

Overview

Last Class:

- Basic probability review. See course site for links to resources to refresh your probability background.
- Linearity of expectation and variance.

Today:

- Algorithmic applications of linearity of expectation and variance.
- Learn about random hash functions, which are a key tool in randomized methods for data processing. Probabilistic analysis via linearity of expectation.
- Introduce Markov's inequality a fundamental concentration bound that let us prove that a random variable lies close to its expectation with good probability.

Linearity of Expectation and Variance

Linearity of Expectation: $\mathbb{E}[X + Y] = \mathbb{E}[X] + \mathbb{E}[Y]$ for *any* random variables X and Y. Proof via definition of expectation.

Linearity of Variance: $\mathrm{Var}[X + Y] = \mathrm{Var}[X] + \mathrm{Var}[Y]$ when X and Y are uncorrelated, and in particular, when they are independent.

Proof using two claims:

Claim 1: $\mathrm{Var}[X] = \mathbb{E}[X^2] - \mathbb{E}[X]^2$ (via linearity of expectation).

Claim 2: $\mathbb{E}[XY] = \mathbb{E}[X] \cdot \mathbb{E}[Y]$ (i.e., X and Y are uncorrelated) when X, Y are independent (via definition of independence).

Hash Tables

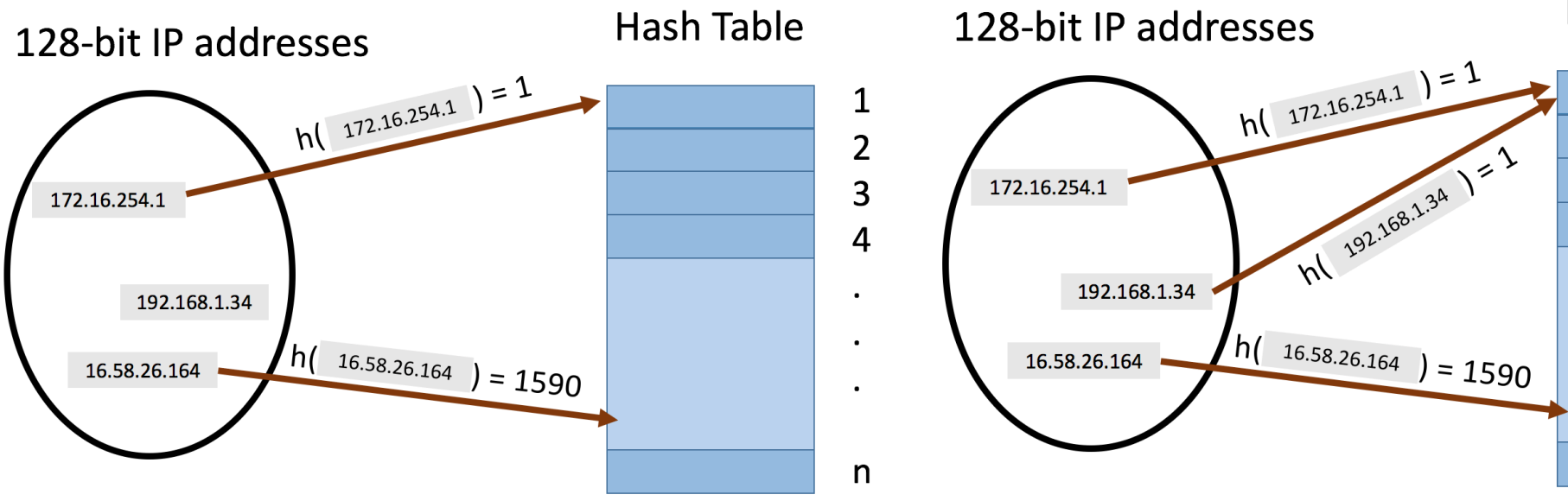
Want to store a set of items from some finite but massive universe U of items (e.g., images of a certain size, text documents, 128-bit IP addresses).

Goal: support *query*(x) to check if x is in the set in $O(1)$ time.

Classic Solution: Hash tables

- *Static hashing* since we won't worry about insertion and deletion today.

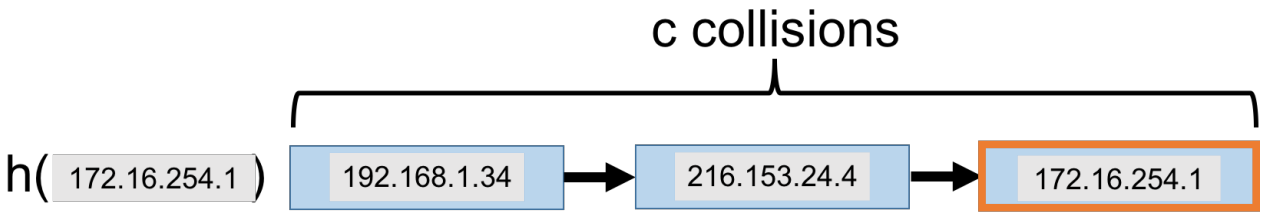
Hash Tables



- **hash function** $h : U \to [n]$ maps elements from the universe to indices $1, \cdots, n$ of an array.
- Typically $|U| \gg n$. Many elements map to the same index.
- **Collisions:** when we insert m items into the hash table we may have to store multiple items in the same location (typically as a linked list).

Collisions

Query runtime: $O(c)$ when the maximum number of collisions in a table entry is c (i.e., must traverse a linked list of size c).



How Can We Bound c (and thus our runtime)?

- In the worst case could have $c = m$ (all items hash to the same location).
- To avoid this, we'll assume the hash function is random, and so this event is very unlikely.

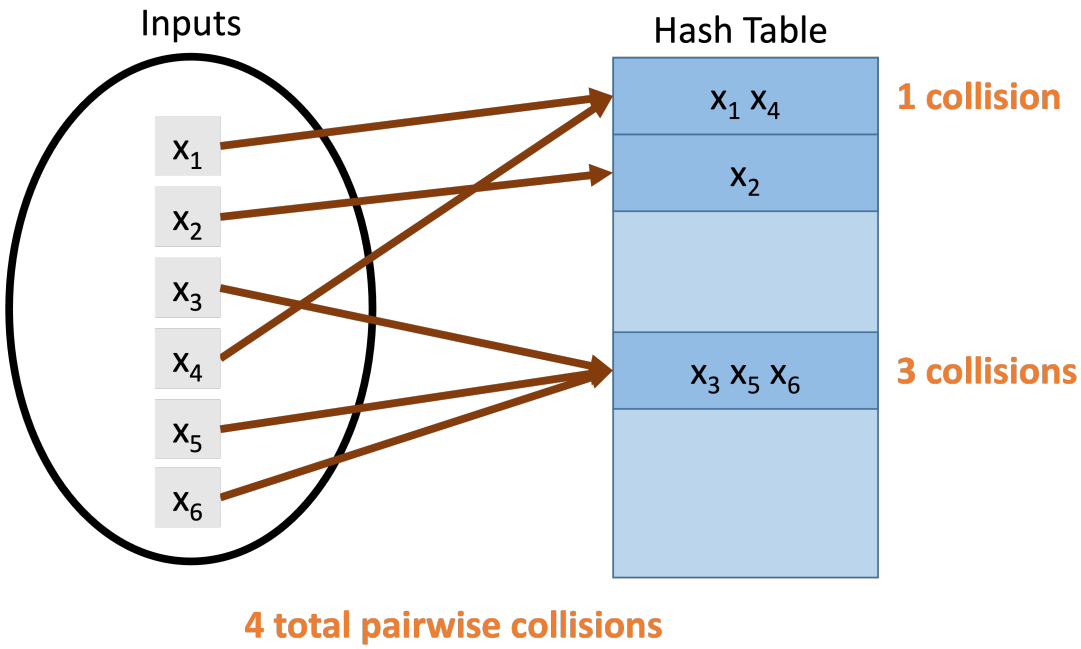
Random Hash Function

Let $\mathbf{h} : U \to [n]$ be a fully random hash function.

- I.e., for $x \in U$, $\Pr(\mathbf{h}(x) = i) = \frac{1}{n}$ for all $i = 1, \ldots, n$ and $\mathbf{h}(x), \mathbf{h}(y)$ are independent for any two items $x \neq y$.
- **Caveat 1:** It is *very expensive* to represent and compute such a random function. We will later see how a hash function computable in $O(1)$ time function can be used instead.
- **Caveat 2:** In practice, often suffices to use hash functions like MD5, SHA-2, etc. that 'look random enough'.

Bounding the Number of Collisions

We say that two items are a 'pairwise collision' if they land in the same bucket of our hash table.

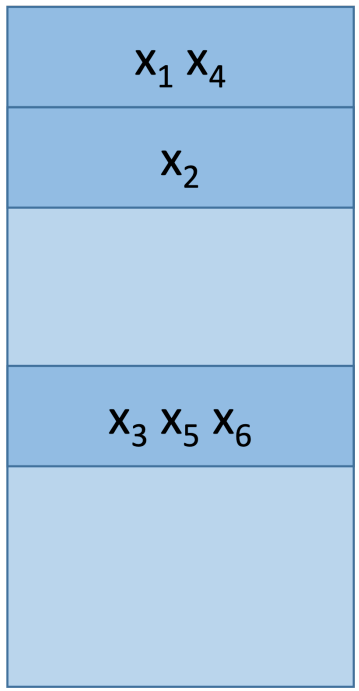


Think-Pair-Share: Assuming we insert m elements into a hash table of size n using a fully random hash function, what is the expected total number of pairwise collisions?

Student Answers

Linearity of Expectation

Let $C_{i,j} = 1$ if items i and j collide ($h(x_i) = h(x_j)$), and 0 otherwise. An indicator random variable.



$C_{1,4} = 1$

$C_{3,5} = 1, C_{3,6} = 1, C_{5,6} = 1$

The number of pairwise duplicates (a random variable) is:

$$C = \sum_{i,j \in [m], i<j} C_{i,j}. \mathbb{E}[C] = \sum_{i,j \in [m], i<j} \mathbb{E}[C_{i,j}].$$

For any pair $i, j \in [m], i < j$: $\mathbb{E}[C_{i,j}] = \Pr[C_{i,j} = 1] = \frac{1}{n}$.

Connection to the Birthday Paradox

If there are a 60 people in this room, each whose birthday we assume to be a uniformly random day of the 365 days in the year, how many pairwise duplicate birthdays do we expect there are?

$$\mathbb{E}[\mathsf{D}] = \frac{m(m-1)}{2n} = \frac{60 \cdot 59}{2 \cdot 365} \approx 5.$$

Concentration Inequalities

If you insert e.g., $m = 10$ items into a hash table of size $n = 100$, the expected number of collisions is:

$$\mathbb{E}[C] = \frac{m(m-1)}{2n} = .45$$

So on average we have < 1 collision. Query time should be very fast. But there is still some probability of a slow query time – how can we argue that it is small?

Concentration Inequalities: Bounds on the probability that a random variable deviates a certain distance from its mean.

- Useful in understanding how statistical tests perform, the behavior of randomized algorithms, the behavior of data drawn from different distributions, etc.

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

Markov's Inequality

The most fundamental concentration bound: **Markov's inequality.**

For any non-negative random variable X and any $t > 0$:

$$\Pr[X \geq t] \leq \frac{\mathbb{E}[X]}{t}.$$

Proof:

$$\begin{aligned}\mathbb{E}[X] = \sum_s \Pr(X = s) \cdot s &\geq \sum_{s \geq t} \Pr(X = s) \cdot s \\ &\geq \sum_{s \geq t} \Pr(X = s) \cdot t \\ &= t \cdot \Pr(X \geq t).\end{aligned}$$

Useful form: $\Pr[X \geq t \cdot \mathbb{E}[X]] \leq \frac{1}{t}$.

The larger the deviation t, the smaller the probability.

Application to Hash Tables

If you insert e.g., $m = 10$ items into a hash table of size $n = 100$, the expected number of collisions is:

$$\mathbb{E}[C] = \frac{m(m-1)}{2n} = .45$$

By Markov's inequality, the probability that you have ≥ 10 collisions is at most:

$$\Pr[C \geq 10] \leq \frac{\mathbb{E}[C]}{10} \leq .045.$$

I.e., there is greater than a 95% chance that your query time is bounded by 10.