COMPSCI 514: Algorithms for Data Science

Cameron Musco

University of Massachusetts Amherst. Fall 2023.

Lecture 2

Reminder

Reminders:

- Sign up for Piazza.
- Vote on preferred office hours times. I will fix my office hours by the end of this week.
- Find homework teammates (see Piazza Post) and sign up for Gradescope (code on course website).
- Week 1 Quiz will be available after class and is due **Monday at 8:00pm**.
- Pset 1 released in next few days (hopefully)

Overview

Last Class:

- Basic probability review. See course site for links to resources to refresh your probability background.
- Linearity of expectation: $\mathbb{E}[X+Y] = \mathbb{E}[X] + \mathbb{E}[Y]$ always.

Overview

Last Class:

- Basic probability review. See course site for links to resources to refresh your probability background.
- Linearity of expectation: $\mathbb{E}[X + Y] = \mathbb{E}[X] + \mathbb{E}[Y]$ always.

Today:

- Linearity of variance: when does $\text{Var}[X + Y] = \text{Var}[X] + \text{Var}[Y]$?
- Algorithmic applications of linearity of expectation and variance.
- Introduce Markov's inequality a fundamental concentration bound that let us prove that a random variable lies close to its expectation with good probability.
- Learn about random hash functions, which are a key tool in randomized methods for data processing. Probabilistic analysis via linearity of expectation.

Linearity of Variance

$$\mathrm{Var}[X + Y] = \mathrm{Var}[X] + \mathrm{Var}[Y]$$

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.

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.

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

$\mathrm{Var}[X] = \mathbb{E}[(X - \mathbb{E})^2]$

Linearity of Variance

$\text{Var}[X + Y] = \text{Var}[X] + \text{Var}[Y]$ when X and Y are uncorrelated, and in particular, when they are independent.

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

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

$Pr(X = s \cap Y = t)$

Linearity of Variance

$\text{Var}[X + Y] = \text{Var}[X] + \text{Var}[Y]$ when X and Y are uncorrelated, and in particular, when they are independent.

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

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

Together give:

Linearity of Variance

$\text{Var}[X+Y] = \text{Var}[X] + \text{Var}[Y]$ when X and Y are uncorrelated, and in particular, when they are independent.

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

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

Together give:

$$\text{Var}[X+Y] = \mathbb{E}[(X+Y)^2] - \mathbb{E}[X+Y]^2$$

$$\mathbb{E}[X^2 + 2XY + Y^2] - (\mathbb{E}[X] + \mathbb{E}[Y])^2$$

$$\downarrow$$

$$\mathbb{E}[X^2] + 2\mathbb{E}[XY] + \mathbb{E}[Y^2] - \mathbb{E}[X]^2 - 2\mathbb{E}X\mathbb{E}Y - \mathbb{E}[Y]^2$$

Linearity of Variance

$\text{Var}[X + Y] = \text{Var}[X] + \text{Var}[Y]$ when X and Y are uncorrelated, and in particular, when they are independent.

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

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

Together give:

$$\begin{aligned}\text{Var}[X + Y] &= \mathbb{E}[(X+Y)^2] - \mathbb{E}[X+Y]^2 \\ &= \mathbb{E}[X^2] + 2\mathbb{E}[XY] + \mathbb{E}[Y^2] - (\mathbb{E}[X] + \mathbb{E}[Y])^2\end{aligned}$$

(linearity of expectation)

Linearity of Variance

$\text{Var}[X+Y] = \text{Var}[X] + \text{Var}[Y]$ when X and Y are uncorrelated, and in particular, when they are independent.

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

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

Together give:

$$\begin{aligned}
\text{Var}[X+Y] &= \mathbb{E}[(X+Y)^2] - \mathbb{E}[X+Y]^2 \\
&= \mathbb{E}[X^2] + 2\mathbb{E}[XY] + \mathbb{E}[Y^2] - (\mathbb{E}[X] + \mathbb{E}[Y])^2 \\
&\qquad \text{(linearity of expectation)} \\
&= \mathbb{E}[X^2] + 2\mathbb{E}[XY] + \mathbb{E}[Y^2] - \mathbb{E}[X]^2 - 2\mathbb{E}[X] \cdot \mathbb{E}[Y] - \mathbb{E}[Y]^2
\end{aligned}$$

Var(X) + Var(Y)

Linearity of Variance

$\text{Var}[X + Y] = \text{Var}[X] + \text{Var}[Y]$ when X and Y are uncorrelated, and in particular, when they are independent.

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

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

Together give:

$$\begin{aligned}
\text{Var}[X + Y] &= \mathbb{E}[(X + Y)^2] - \mathbb{E}[X + Y]^2 \\
&= \mathbb{E}[X^2] + 2\mathbb{E}[XY] + \mathbb{E}[Y^2] - (\mathbb{E}[X] + \mathbb{E}[Y])^2 \\
&\qquad\qquad\qquad\qquad \text{(linearity of expectation)} \\
&= \mathbb{E}[X^2] + 2\mathbb{E}[XY] + \mathbb{E}[Y^2] - \mathbb{E}[X]^2 - 2\mathbb{E}[X] \cdot \mathbb{E}[Y] - \mathbb{E}[Y]^2
\end{aligned}$$

Linearity of Variance

$\text{Var}[X+Y] = \text{Var}[X] + \text{Var}[Y]$ when X and Y are uncorrelated, and in particular, when they are independent.

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

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

Together give:

$$\begin{aligned}
\text{Var}[X+Y] &= \mathbb{E}[(X+Y)^2] - \mathbb{E}[X+Y]^2 \\
&= \mathbb{E}[X^2] + 2\mathbb{E}[XY] + \mathbb{E}[Y^2] - (\mathbb{E}[X] + \mathbb{E}[Y])^2 \\
&\qquad \text{(linearity of expectation)} \\
&= \mathbb{E}[X^2] + 2\mathbb{E}[XY] + \mathbb{E}[Y^2] - \mathbb{E}[X]^2 - 2\mathbb{E}[X] \cdot \mathbb{E}[Y] - \mathbb{E}[Y]^2 \\
&= \mathbb{E}[X^2] + \mathbb{E}[Y^2] - \mathbb{E}[X]^2 - \mathbb{E}[Y]^2
\end{aligned}$$

Linearity of Variance

$\text{Var}[X+Y] = \text{Var}[X] + \text{Var}[Y]$ when X and Y are uncorrelated, and in particular, when they are independent.

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

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

Together give:

$$\begin{aligned}
\text{Var}[X+Y] &= \mathbb{E}[(X+Y)^2] - \mathbb{E}[X+Y]^2 \\
&= \mathbb{E}[X^2] + 2\mathbb{E}[XY] + \mathbb{E}[Y^2] - (\mathbb{E}[X] + \mathbb{E}[Y])^2 \\
&\qquad\qquad\qquad\qquad \text{(linearity of expectation)} \\
&= \mathbb{E}[X^2] + 2\mathbb{E}[XY] + \mathbb{E}[Y^2] - \mathbb{E}[X]^2 - 2\mathbb{E}[X] \cdot \mathbb{E}[Y] - \mathbb{E}[Y]^2 \\
&= \mathbb{E}[X^2] + \mathbb{E}[Y^2] - \mathbb{E}[X]^2 - \mathbb{E}[Y]^2 \\
&= \text{Var}[X] + \text{Var}[Y].
\end{aligned}$$

An Algorithmic Application

You have contracted with a new company to provide CAPTCHAS for your website.

An Algorithmic Application

You have contracted with a new company to provide CAPTCHAS for your website.

- They claim that they have a database of $1,000,000$ unique CAPTCHAS. A random one is chosen for each security check.
- You want to independently verify this claimed database size.

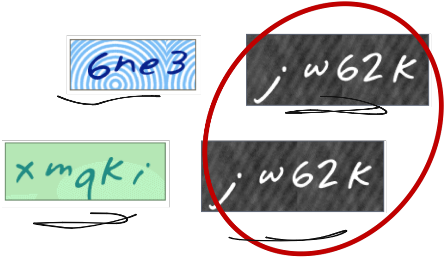
An Algorithmic Application

You have contracted with a new company to provide CAPTCHAS for your website.

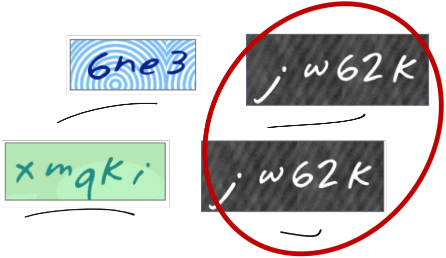
- They claim that they have a database of $1,000,000$ unique CAPTCHAS. A random one is chosen for each security check.
- You want to independently verify this claimed database size.
- You could make test checks until you see $1,000,000$ unique CAPTCHAS: would take $\geq 1,000,000$ checks!

An Algorithmic Application

An Idea: You run some test security checks and see if any duplicate CAPTCHAS show up. If you're seeing duplicates after not too many checks, the database size is probably not too big.



An Idea: You run some test security checks and see if any duplicate CAPTCHAS show up. If you're seeing duplicates after not too many checks, the database size is probably not too big.



'Mark and recapture' method in ecology.

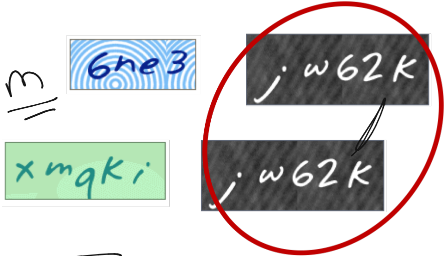
$D_{1,2} = 0$

$B_{3,4} = 1$

$D_{2,3} = 0$

An Algorithmic Application

An Idea: You run some test security checks and see if any duplicate CAPTCHAS show up. If you're seeing duplicates after not too many checks, the database size is probably not too big.



'Mark and recapture' method in ecology.

Think-Pair-Share: If you run m security checks, and there are n unique CAPTCHAS, how many pairwise duplicates do you see in expectation?

If e.g. the same CAPTCHA shows up three times, on your i^{th}, j^{th}, and k^{th} test, this is three duplicates: (i, j), (i, k) and (j, k).

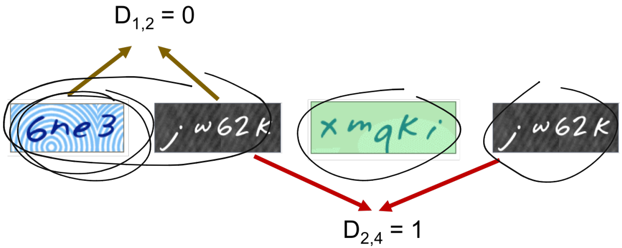
Linearity of Expectation

Let $\mathbf{D}_{i,j} = 1$ if tests i and j give the same CAPTCHA, and 0 otherwise. An indicator random variable.

n: number of CAPTCHAS in database, m: number of random CAPTCHAS drawn to check database size, $\mathbf{D}$: number of pairwise duplicates in m random CAPTCHAS

Linearity of Expectation

Let $\mathbf{D}_{i,j} = 1$ if tests i and j give the same CAPTCHA, and 0 otherwise. An indicator random variable.



n: number of CAPTCHAS in database, m: number of random CAPTCHAS drawn to check database size, $\mathbf{D}$: number of pairwise duplicates in m random CAPTCHAS

Linearity of Expectation

Let $\mathsf{D}_{i,j} = 1$ if tests i and j give the same CAPTCHA, and 0 otherwise. An indicator random variable. The number of pairwise duplicates (a random variable) is:

$$\mathsf{D} = \sum_{i,j \in [m], i<j} \mathsf{D}_{i,j}.$$

n: number of CAPTCHAS in database, m: number of random CAPTCHAS drawn to check database size, D: number of pairwise duplicates in m random CAPTCHAS

Let $\mathbf{D}_{i,j} = 1$ if tests i and j give the same CAPTCHA, and 0 otherwise. An indicator random variable. The number of pairwise duplicates (a random variable) is:

$$\mathbb{E}[\mathbf{D}] = \sum_{i,j \in [m], i<j} \mathbb{E}[\mathbf{D}_{i,j}].$$

$\frac{1}{n}$

$D_{ij} = 1 \text{ w.p. } \frac{1}{n}$

$D_{ij} = 0 \text{ o.w.}$

$\mathbb{E} D_{i,j} = \frac{1}{n}$

$D_{12} = 1$

$D_{14} = 1 \Rightarrow D_{24} = 1$

n: number of CAPTCHAS in database, m: number of random CAPTCHAS drawn to check database size, $\mathbf{D}$: number of pairwise duplicates in m random CAPTCHAS

Linearity of Expectation

Let $\mathsf{D}_{i,j} = 1$ if tests i and j give the same CAPTCHA, and 0 otherwise. An indicator random variable. The number of pairwise duplicates (a random variable) is:

$$\mathbb{E}[\mathsf{D}] = \sum_{i,j\in[m],i<j} \mathbb{E}[\mathsf{D}_{i,j}].$$

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

n: number of CAPTCHAS in database, m: number of random CAPTCHAS drawn to check database size, D: number of pairwise duplicates in m random CAPTCHAS

Linearity of Expectation

Let $\mathsf{D}_{i,j} = 1$ if tests i and j give the same CAPTCHA, and 0 otherwise. An indicator random variable. The number of pairwise duplicates (a random variable) is:

$$\mathbb{E}[\mathsf{D}] = \sum_{i,j \in [m], i<j} \mathbb{E}[\mathsf{D}_{i,j}].$$

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

$$\mathbb{E}[\mathsf{D}] = \sum_{i,j \in [m], i<j} \frac{1}{n} = \frac{\binom{m}{2}}{n} = \frac{m(m-1)}{2n}.$$

n: number of CAPTCHAS in database, m: number of random CAPTCHAS drawn to check database size, D: number of pairwise duplicates in m random CAPTCHAS

Linearity of Expectation

Let $\mathbf{D}_{i,j} = 1$ if tests i and j give the same CAPTCHA, and 0 otherwise. An indicator random variable. The number of pairwise duplicates (a random variable) is:

$$\mathbb{E}[\mathbf{D}] = \sum_{i,j \in [m], i<j} \mathbb{E}[\mathbf{D}_{i,j}].$$

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

$$\mathbb{E}[\mathbf{D}] = \sum_{i,j \in [m], i<j} \frac{1}{n} = \frac{\binom{m}{2}}{n} = \frac{m(m-1)}{2n}.$$

Note that the $\mathbf{D}_{i,j}$ random variables are not independent!

n: number of CAPTCHAS in database, m: number of random CAPTCHAS drawn to check database size, $\mathbf{D}$: number of pairwise duplicates in m random CAPTCHAS

Connection to the Birthday Paradox

If there are a 150 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?

If there are a 150 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?

$$\mathbb{E}[D] = \frac{m(m-1)}{2n} = \frac{150 \cdot 149}{2 \cdot 365} \approx 31.$$

Linearity of Expectation

You take $m = 1000$ samples. If the database size is as claimed ($n = 1,000,000$) then expected number of duplicates is:

$$\mathbb{E}[D] = \frac{m(m-1)}{2n} = .4995$$

1000²

2mil

n: number of CAPTCHAS in database, m: number of random CAPTCHAS drawn to check database size, D: number of pairwise duplicates in m random CAPTCHAS.

Linearity of Expectation

You take $m = 1000$ samples. If the database size is as claimed ($n = 1,000,000$) then expected number of duplicates is:

$$\mathbb{E}[\mathbf{D}] = \frac{m(m-1)}{2n} = .4995$$

You see 10 pairwise duplicates and suspect that something is up. But how confident can you be in your test?

n: number of CAPTCHAS in database, m: number of random CAPTCHAS drawn to check database size, $\mathbf{D}$: number of pairwise duplicates in m random CAPTCHAS.

Linearity of Expectation

You take $m = 1000$ samples. If the database size is as claimed ($n = 1,000,000$) then expected number of duplicates is:

$$\mathbb{E}[\mathbf{D}] = \frac{m(m-1)}{2n} = .4995$$

You see 10 pairwise duplicates and suspect that something is up. But how confident can you be in your test?

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

n: number of CAPTCHAS in database, m: number of random CAPTCHAS drawn to check database size, $\mathbf{D}$: number of pairwise duplicates in m random CAPTCHAS.

Linearity of Expectation

You take $m = 1000$ samples. If the database size is as claimed ($n = 1,000,000$) then expected number of duplicates is:

$$\mathbb{E}[\mathbf{D}] = \frac{m(m-1)}{2n} = .4995$$

You see 10 pairwise duplicates and suspect that something is up. But how confident can you be in your test?

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.

> n: number of CAPTCHAS in database, m: number of random CAPTCHAS drawn to check database size, $\mathbf{D}$: number of pairwise duplicates in m random CAPTCHAS.

Markov's Inequality

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

Markov's Inequality

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

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

$$\Pr[X \geq t] \leq \frac{\mathbb{E}[X]}{t}. \leq \frac{1}{S}$$

$t = S \cdot \mathbb{E}[X]$

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:

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:

$$\mathbb{E}[X] = \sum_{s} \Pr(X = s) \cdot s$$

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:

$$\mathbb{E}[X] = \sum_{s} \Pr(X = s) \cdot s \geq \sum_{s \geq t} \Pr(X = s) \cdot s$$

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:

$$\mathbb{E}[X] = \sum_{s} \Pr(X = s) \cdot s \geq \sum_{s \geq t} \Pr(X = s) \cdot s$$

$$\geq \sum_{s \geq t} \Pr(X = s) \cdot t$$

$P(X \geq t)$

Markov's Inequality

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

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

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

$t = 5$

Proof:

$\frac{1}{6} \cdot 1 + \frac{1}{6} \cdot 2 + \ldots \frac{1}{6} \cdot 6$ $\quad$ $\frac{1}{6} \cdot 5 + \frac{1}{6} \cdot 6$

$$\mathbb{E}[X] = \sum_{s} \Pr(X = s) \cdot s \geq \sum_{s \geq t} \Pr(X = s) \cdot s$$

$$\geq \sum_{s \geq t} \Pr(X = s) \cdot t$$

$\frac{1}{6} \cdot 5 + \frac{1}{6} \cdot 5$

$$= t \cdot \Pr(X \geq t).$$

$= \frac{2}{6} \cdot 5$

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

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.

Back to Our Application

Expected number of duplicate CAPTCHAS:

$\mathbb{E}[\mathsf{D}] = \frac{m(m-1)}{2n} = .4995.$

You see $\mathsf{D} = 10$ duplicates.

n: number of CAPTCHAS in database ($n = 1,000,000$ claimed) , m: number of random CAPTCHAS drawn to check database size ($m = 1000$ in this example), D: number of pairwise duplicates in m random CAPTCHAS.

Back to Our Application

Expected number of duplicate CAPTCHAS:

$\mathbb{E}[\mathsf{D}] = \frac{m(m-1)}{2n} = .4995.$

You see $\mathsf{D} = 10$ duplicates.

Applying Markov's inequality, if the real database size is $n = 1,000,000$ the probability of this happening is:

$$\Pr[\mathsf{D} \geq 10] \leq \frac{\mathbb{E}[\mathsf{D}]}{10} = \frac{.4995}{10} \approx .05$$

n: number of CAPTCHAS in database ($n = 1,000,000$ claimed) , m: number of random CAPTCHAS drawn to check database size ($m = 1000$ in this example), D: number of pairwise duplicates in m random CAPTCHAS.

Back to Our Application

Expected number of duplicate CAPTCHAS:

$\mathbb{E}[\mathrm{D}] = \frac{m(m-1)}{2n} = .4995.$

You see $\mathrm{D} = 10$ duplicates.

Applying Markov's inequality, if the real database size is $n = 1,000,000$ the probability of this happening is:

$$\Pr[\mathrm{D} \geq 10] \leq \frac{\mathbb{E}[\mathrm{D}]}{10} = \frac{.4995}{10} \approx .05$$

This is pretty small – you feel pretty sure the number of unique CAPTCHAS is much less than $1,000,000$. But how can you boost your confidence?

> n: number of CAPTCHAS in database ($n = 1,000,000$ claimed) , m: number of random CAPTCHAS drawn to check database size ($m = 1000$ in this example), D: number of pairwise duplicates in m random CAPTCHAS.

Back to Our Application

Expected number of duplicate CAPTCHAS:

$\mathbb{E}[\mathbf{D}] = \frac{m(m-1)}{2n} = .4995.$

You see $\mathbf{D} = 10$ duplicates.

Applying Markov's inequality, if the real database size is $n = 1,000,000$ the probability of this happening is:

$$\Pr[\mathbf{D} \geq 10] \leq \frac{\mathbb{E}[\mathbf{D}]}{10} = \frac{.4995}{10} \approx .05$$

This is pretty small – you feel pretty sure the number of unique CAPTCHAS is much less than $1,000,000$. But how can you boost your confidence? We'll discuss in the next few classes.

> n: number of CAPTCHAS in database ($n = 1,000,000$ claimed) , m: number of random CAPTCHAS drawn to check database size ($m = 1000$ in this example), $\mathbf{D}$: number of pairwise duplicates in m random CAPTCHAS.

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).

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.

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/maps

bloom filter

binary trees

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

Hash Tables

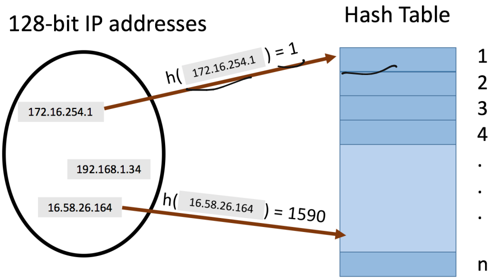
Want to store a set of items from some finite but massive universe U of items (e.g., images of a certain size, text documents, 128-bit IP addresses).

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

Classic Solution: Hash tables

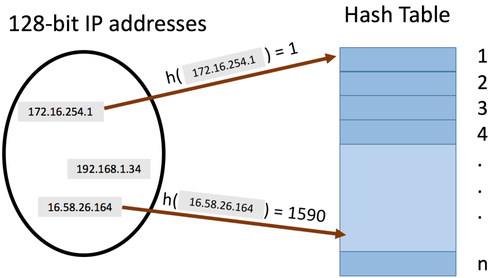
- *Static hashing* since we won't worry about insertion and deletion today.

Hash Tables



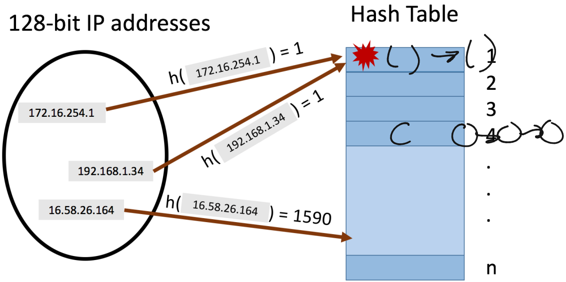
- **hash function** $h : U \to [n]$ maps elements from the universe to indices $1, \cdots, n$ of an array.

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.

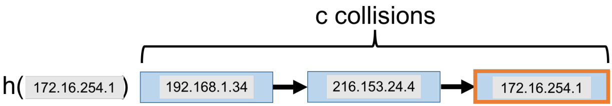
Hash Tables



- **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).

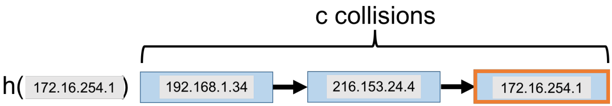
Collisions

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).



Collisions

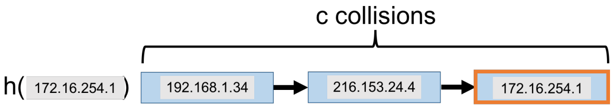
Query runtime: $O(c)$ when the maximum number of collisions in a table entry is c (i.e., must traverse a linked list of size c).



How Can We Bound c?

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).

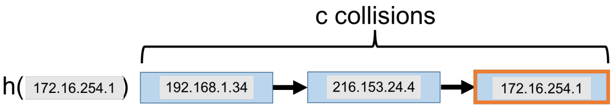


How Can We Bound c?

- In the worst case could have $c = m$ (all items hash to the same location).

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?

- 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 $\mathbf{h} : U \rightarrow [n]$ 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 $x \neq y$.

$h(1) = 7 \quad h(2) = 2 \quad h(23) = 1$

$h(1) = 7$

pick random a, b

$h(x) = a \cdot x + b \cdot x^2 \mod n$

random seeds

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'.

Random Hash Function

Let $\mathbf{h} : U \rightarrow [n]$ 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'.

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?

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

Linearity of Expectation

Let $\mathsf{C}_{i,j} = 1$ if items i and j collide $(\mathsf{h}(x_i) = \mathsf{h}(x_j))$, and 0 otherwise. The number of pairwise duplicates is:

$$\mathsf{C} = \sum_{i,j \in [m], i<j} \mathsf{C}_{i,j}.$$

x_i, x_j: pair of stored items, m: total number of stored items, n: hash table size, C: total pairwise collisions in table, h: random hash function.

Linearity of Expectation

Let $C_{i,j} = 1$ if items i and j collide ($h(x_i) = h(x_j)$), and 0 otherwise. The number of pairwise duplicates is:

$$\mathbb{E}[C] = \sum_{i,j \in [m], i<j} \mathbb{E}[C_{i,j}]. \qquad \text{(linearity of expectation)}$$

x_i, x_j: pair of stored items, m: total number of stored items, n: hash table size, C: total pairwise collisions in table, h: random hash function.

Linearity of Expectation

Let $C_{i,j} = 1$ if items i and j collide ($h(x_i) = h(x_j)$), and 0 otherwise. The number of pairwise duplicates is:

$$\mathbb{E}[C] = \sum_{i,j \in [m], i<j} \mathbb{E}[C_{i,j}]. \qquad \text{(linearity of expectation)}$$

For any pair i, j, $i < j$:

$$\mathbb{E}[C_{i,j}] = \Pr[C_{i,j} = 1] = \Pr[h(x_i) = h(x_j)] = \frac{1}{n}$$

x_i, x_j: pair of stored items, m: total number of stored items, n: hash table size, C: total pairwise collisions in table, h: random hash function.

Linearity of Expectation

Let $\mathsf{C}_{i,j} = 1$ if items i and j collide ($\mathsf{h}(x_i) = \mathsf{h}(x_j)$), and 0 otherwise. The number of pairwise duplicates is:

$$\mathbb{E}[\mathsf{C}] = \sum_{i,j \in [m], i<j} \mathbb{E}[\mathsf{C}_{i,j}]. \qquad \text{(linearity of expectation)}$$

For any pair i, j, $i < j$:

$\mathbb{E}[\mathsf{C}_{i,j}] = \Pr[\mathsf{C}_{i,j} = 1] = \Pr[\mathsf{h}(x_i) = \mathsf{h}(x_j)] = \frac{1}{n}$.

x_i, x_j: pair of stored items, m: total number of stored items, n: hash table size, C: total pairwise collisions in table, h: random hash function.

Linearity of Expectation

Let $\mathsf{C}_{i,j} = 1$ if items i and j collide ($\mathsf{h}(x_i) = \mathsf{h}(x_j)$), and 0 otherwise. The number of pairwise duplicates is:

$$\mathbb{E}[\mathsf{C}] = \sum_{i,j\in[m],i<j} \mathbb{E}[\mathsf{C}_{i,j}]. \qquad \text{(linearity of expectation)}$$

For any pair i, j, $i < j$:

$$\mathbb{E}[\mathsf{C}_{i,j}] = \Pr[\mathsf{C}_{i,j} = 1] = \Pr[\mathsf{h}(x_i) = \mathsf{h}(x_j)] = \tfrac{1}{n}.$$

$$\mathbb{E}[\mathsf{C}] = \sum_{i,j\in[m],i<j} \frac{1}{n} = \frac{\binom{m}{2}}{n} = \frac{m(m-1)}{2n}.$$

x_i, x_j: pair of stored items, m: total number of stored items, n: hash table size, C: total pairwise collisions in table, h: random hash function.

Linearity of Expectation

Let $C_{i,j} = 1$ if items i and j collide ($h(x_i) = h(x_j)$), and 0 otherwise. The number of pairwise duplicates is:

$$\mathbb{E}[C] = \sum_{i,j \in [m], i<j} \mathbb{E}[C_{i,j}]. \qquad \text{(linearity of expectation)}$$

For any pair i, j, $i < j$:
$\mathbb{E}[C_{i,j}] = \Pr[C_{i,j} = 1] = \Pr[h(x_i) = h(x_j)] = \frac{1}{n}$.

$$\mathbb{E}[C] = \sum_{i,j \in [m], i<j} \frac{1}{n} = \frac{\binom{m}{2}}{n} = \frac{m(m-1)}{2n}.$$

Identical to the CAPTCHA analysis!

x_i, x_j: pair of stored items, m: total number of stored items, n: hash table size, C: total pairwise collisions in table, h: random hash function.

Collision Free Hashing

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

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

Collision Free Hashing

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

4m²

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

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

Collision Free Hashing

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

- For $n = 4m^2$ we have: $\mathbb{E}[\mathbf{C}] = \frac{m(m-1)}{8m^2} \leq \frac{1}{8}$.
- ~~Think-Pair-Share:~~ What is an upper bound on the probability that we have any collisions, i.e., $\Pr[\mathbf{C} \geq 1]$?

$t=1$

$$\Pr[C \geq 1] \leq \frac{\mathbb{E}[C]}{1} = \frac{1/8}{1} = \frac{1}{8}$$

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

Collision Free Hashing

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

- For $n = 4m^2$ we have: $\mathbb{E}[\mathbf{C}] = \frac{m(m-1)}{8m^2} \leq \frac{1}{8}$.
- **Think-Pair-Share:** What is an upper bound on the probability that we have any collisions, i.e., $\Pr[\mathbf{C} \geq 1]$?

Apply Markov's Inequality:

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

Collision Free Hashing

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

- For $n = 4m^2$ we have: $\mathbb{E}[C] = \frac{m(m-1)}{8m^2} \leq \frac{1}{8}$.
- **Think-Pair-Share:** What is an upper bound on the probability that we have any collisions, i.e., $\Pr[C \geq 1]$?

Apply Markov's Inequality: $\Pr[C \geq 1] \leq \frac{\mathbb{E}[C]}{1}$

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

Collision Free Hashing

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

- For $n = 4m^2$ we have: $\mathbb{E}[\mathrm{C}] = \frac{m(m-1)}{8m^2} \leq \frac{1}{8}$.
- **Think-Pair-Share:** What is an upper bound on the probability that we have any collisions, i.e., $\Pr[\mathrm{C} \geq 1]$?

Apply Markov's Inequality: $\Pr[\mathrm{C} \geq 1] \leq \frac{\mathbb{E}[\mathrm{C}]}{1} = \frac{1}{8}$.

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

Collision Free Hashing

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

- For $n = 4m^2$ we have: $\mathbb{E}[\mathbf{C}] = \frac{m(m-1)}{8m^2} \leq \frac{1}{8}$.
- **Think-Pair-Share:** What is an upper bound on the probability that we have any collisions, i.e., $\Pr[\mathbf{C} \geq 1]$?

Apply Markov's Inequality: $\Pr[\mathbf{C} \geq 1] \leq \frac{\mathbb{E}[\mathbf{C}]}{1} = \frac{1}{8}$.

So with probability at least 7/8 we have no collisions and worst-case $O(1)$ query time.

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

Collision Free Hashing

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

- For $n = 4m^2$ we have: $\mathbb{E}[\mathrm{C}] = \frac{m(m-1)}{8m^2} \leq \frac{1}{8}$.
- **Think-Pair-Share:** What is an upper bound on the probability that we have any collisions, i.e., $\Pr[\mathrm{C} \geq 1]$?

Apply Markov's Inequality: $\Pr[\mathrm{C} \geq 1] \leq \frac{\mathbb{E}[\mathrm{C}]}{1} = \frac{1}{8}$.

So with probability at least 7/8 we have no collisions and worst-case $O(1)$ query time.

Pretty good...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.