COMPSCI 514: ALGORITHMS FOR DATA SCIENCE

Andrew McGregor

Lecture 2

Today:

- Investigate linearity of expectation and variance.
- Algorithmic application 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.

- **Expectation:** $\mathbb{E}[\mathbf{X}] = \sum_{s \in S} \Pr(\mathbf{X} = s) \cdot s.$

- **Expectation:** $\mathbb{E}[\mathbf{X}] = \sum_{s \in S} \Pr(\mathbf{X} = s) \cdot s.$
- **Variance:** $\mathrm{Var}[\mathbf{X}] = \mathbb{E}[(\mathbf{X} - \mathbb{E}[\mathbf{X}])^2].$

- **Expectation:** $\mathbb{E}[\mathbf{X}] = \sum_{s \in S} \Pr(\mathbf{X} = s) \cdot s.$
- **Variance:** $\mathrm{Var}[\mathbf{X}] = \mathbb{E}[(\mathbf{X} - \mathbb{E}[\mathbf{X}])^2].$
- Two random variables $\mathbf{X}$, $\mathbf{Y}$ are **independent** if for all s, t, $\{\mathbf{X} = s\}$ and $\{\mathbf{Y} = t\}$ are independent events. In other words:

$$\Pr(\{\mathbf{X} = s\} \cap \{\mathbf{Y} = t\}) = \Pr(\mathbf{X} = s) \cdot \Pr(\mathbf{Y} = t).$$

When are the expectation and variance linear?

I.e., under what conditions on $\mathbf{X}$ and $\mathbf{Y}$ do we have:

$$\mathbb{E}[\mathbf{X} + \mathbf{Y}] = \mathbb{E}[\mathbf{X}] + \mathbb{E}[\mathbf{Y}]$$

and

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

When are the expectation and variance linear?

I.e., under what conditions on $\mathbf{X}$ and $\mathbf{Y}$ do we have:

$$\mathbb{E}[\mathbf{X} + \mathbf{Y}] = \mathbb{E}[\mathbf{X}] + \mathbb{E}[\mathbf{Y}]$$

and

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

Last time we showed that linearity of expectation is true regardless of whether the random variables were independent.

$\mathbf{X}, \mathbf{Y}$: any two random variables.

LINEARITY OF VARIANCE

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

LINEARITY OF VARIANCE

$\mathrm{Var}[\mathbf{X} + \mathbf{Y}] = \mathrm{Var}[\mathbf{X}] + \mathrm{Var}[\mathbf{Y}]$ when $\mathbf{X}$ and $\mathbf{Y}$ are independent.

LINEARITY OF VARIANCE

$\mathrm{Var}[\mathbf{X}+\mathbf{Y}] = \mathrm{Var}[\mathbf{X}] + \mathrm{Var}[\mathbf{Y}]$ when $\mathbf{X}$ and $\mathbf{Y}$ are independent.

Exercise 1: $\mathrm{Var}[\mathbf{X}] = \mathbb{E}[\mathbf{X}^2] - \mathbb{E}[\mathbf{X}]^2$

LINEARITY OF VARIANCE

$\text{Var}[\mathbf{X}+\mathbf{Y}] = \text{Var}[\mathbf{X}] + \text{Var}[\mathbf{Y}]$ when $\mathbf{X}$ and $\mathbf{Y}$ are independent.

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

LINEARITY OF VARIANCE

$\text{Var}[\mathbf{X}+\mathbf{Y}] = \text{Var}[\mathbf{X}] + \text{Var}[\mathbf{Y}]$ when $\mathbf{X}$ and $\mathbf{Y}$ are independent.

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

Exercise 2: $\mathbb{E}[\mathbf{X}\mathbf{Y}] = \mathbb{E}[\mathbf{X}] \cdot \mathbb{E}[\mathbf{Y}]$ when $\mathbf{X}, \mathbf{Y}$ are independent.

$\text{Var}[\mathbf{X} + \mathbf{Y}] = \text{Var}[\mathbf{X}] + \text{Var}[\mathbf{Y}]$ when $\mathbf{X}$ and $\mathbf{Y}$ are independent.

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

Exercise 2: $\mathbb{E}[\mathbf{XY}] = \mathbb{E}[\mathbf{X}] \cdot \mathbb{E}[\mathbf{Y}]$ when $\mathbf{X}, \mathbf{Y}$ are independent.

Together give:

LINEARITY OF VARIANCE

$\mathrm{Var}[\mathbf{X} + \mathbf{Y}] = \mathrm{Var}[\mathbf{X}] + \mathrm{Var}[\mathbf{Y}]$ when $\mathbf{X}$ and $\mathbf{Y}$ are independent.

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

Exercise 2: $\mathbb{E}[\mathbf{X}\mathbf{Y}] = \mathbb{E}[\mathbf{X}] \cdot \mathbb{E}[\mathbf{Y}]$ when $\mathbf{X}, \mathbf{Y}$ are independent.

Together give:

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

$\text{Var}[\mathbf{X} + \mathbf{Y}] = \text{Var}[\mathbf{X}] + \text{Var}[\mathbf{Y}]$ when $\mathbf{X}$ and $\mathbf{Y}$ are independent.

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

Exercise 2: $\mathbb{E}[\mathbf{X}\mathbf{Y}] = \mathbb{E}[\mathbf{X}] \cdot \mathbb{E}[\mathbf{Y}]$ when $\mathbf{X}, \mathbf{Y}$ are independent.

Together give:

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

$\text{Var}[\mathbf{X}+\mathbf{Y}] = \text{Var}[\mathbf{X}] + \text{Var}[\mathbf{Y}]$ when $\mathbf{X}$ and $\mathbf{Y}$ are independent.

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

Exercise 2: $\mathbb{E}[\mathbf{X}\mathbf{Y}] = \mathbb{E}[\mathbf{X}] \cdot \mathbb{E}[\mathbf{Y}]$ when $\mathbf{X}, \mathbf{Y}$ are independent.

Together give:

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

$\mathrm{Var}[\mathbf{X}+\mathbf{Y}] = \mathrm{Var}[\mathbf{X}] + \mathrm{Var}[\mathbf{Y}]$ when $\mathbf{X}$ and $\mathbf{Y}$ are independent.

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

Exercise 2: $\mathbb{E}[\mathbf{XY}] = \mathbb{E}[\mathbf{X}] \cdot \mathbb{E}[\mathbf{Y}]$ when $\mathbf{X}, \mathbf{Y}$ are independent.

Together give:

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

$\mathrm{Var}[\mathbf{X}+\mathbf{Y}] = \mathrm{Var}[\mathbf{X}] + \mathrm{Var}[\mathbf{Y}]$ when $\mathbf{X}$ and $\mathbf{Y}$ are independent.

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

Exercise 2: $\mathbb{E}[\mathbf{XY}] = \mathbb{E}[\mathbf{X}] \cdot \mathbb{E}[\mathbf{Y}]$ when $\mathbf{X}, \mathbf{Y}$ are independent.

Together give:

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

$\mathrm{Var}[\mathbf{X}+\mathbf{Y}] = \mathrm{Var}[\mathbf{X}] + \mathrm{Var}[\mathbf{Y}]$ when $\mathbf{X}$ and $\mathbf{Y}$ are independent.

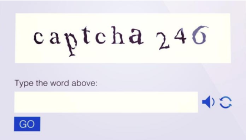
Exercise 1: $\mathrm{Var}[\mathbf{X}] = \mathbb{E}[\mathbf{X}^2] - \mathbb{E}[\mathbf{X}]^2$ (via linearity of expectation)

Exercise 2: $\mathbb{E}[\mathbf{X}\mathbf{Y}] = \mathbb{E}[\mathbf{X}] \cdot \mathbb{E}[\mathbf{Y}]$ when $\mathbf{X}, \mathbf{Y}$ are independent.

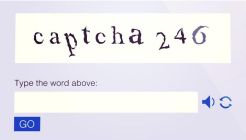
Together give:

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

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

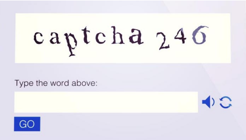


You have contracted with a new company to provide CAPTCHAS for your website.



- 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.

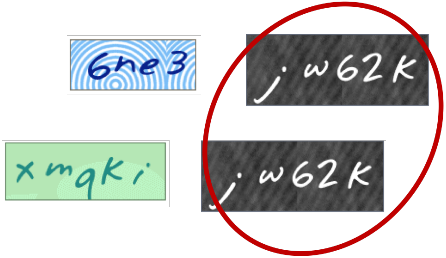
You have contracted with a new company to provide CAPTCHAS for your website.



- 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!

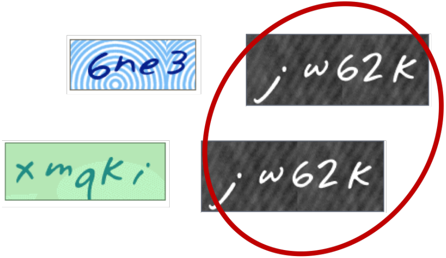
AN ALGORITHMIC APPLICATION

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.



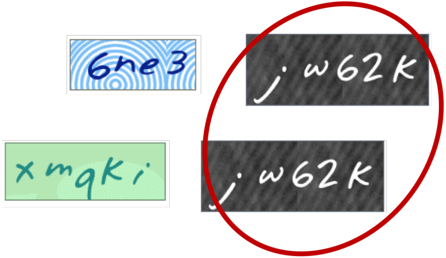
AN ALGORITHMIC APPLICATION

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.



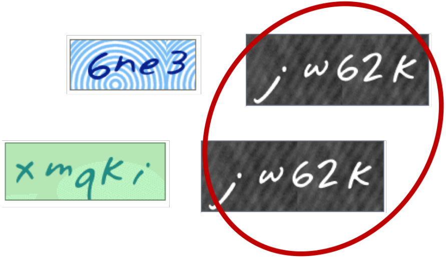
'Mark and recapture' method in ecology.

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.

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.



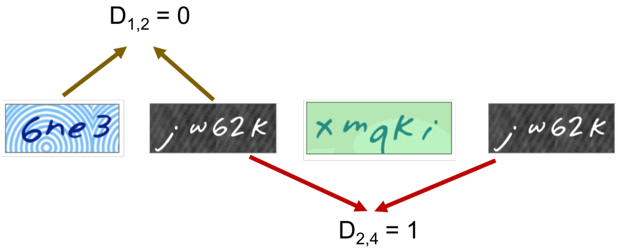
'Mark and recapture' method in ecology.

Note that if the same CAPTCHA shows up four times this counts as $\binom{4}{2}$ duplicates.

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

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

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:

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

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

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

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

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:

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

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

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

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

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

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

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

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

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.

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.

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.

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.

The simplest concentration bound: **Markov's inequality.**

The simplest concentration bound: **Markov's inequality.**

For any non-negative random variable $\mathbf{X}$ and any $t > 0$:

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

The simplest concentration bound: **Markov's inequality.**

For any non-negative random variable $\mathbf{X}$ and any $t > 0$:

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

Proof:

The simplest concentration bound: **Markov's inequality.**

For any non-negative random variable $\mathbf{X}$ and any $t > 0$:

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

Proof:

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

The simplest concentration bound: **Markov's inequality.**

For any non-negative random variable $\mathbf{X}$ and any $t > 0$:

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

Proof:

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

The simplest concentration bound: **Markov's inequality.**

For any non-negative random variable $\mathbf{X}$ and any $t > 0$:

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

Proof:

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

The simplest concentration bound: **Markov's inequality.**

For any non-negative random variable $\mathbf{X}$ and any $t > 0$:

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

Proof:

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

The simplest concentration bound: **Markov's inequality.**

For any non-negative random variable $\mathbf{X}$ and any $t > 0$:

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

Proof:

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

The simplest concentration bound: **Markov's inequality.**

For any non-negative random variable $\mathbf{X}$ and any $t > 0$:

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

Proof:

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

The larger the deviation t, the smaller the probability.

Expected number of duplicate CAPTCHAS:

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

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

n: number of CAPTCHAS in database ($n = 1000000$ 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.

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 $n = 1000000$ the probability of this happening is:

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

n: number of CAPTCHAS in database ($n = 1000000$ 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.

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 $n = 1000000$ 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 and you feel pretty sure the number of unique CAPTCHAS is much less than 1000000.

n: number of CAPTCHAS in database ($n = 1000000$ 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.

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

Want to store a set of items from some finite but massive universe 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.

Want to store a set of items from some finite but massive universe 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:

Want to store a set of items from some finite but massive universe 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.

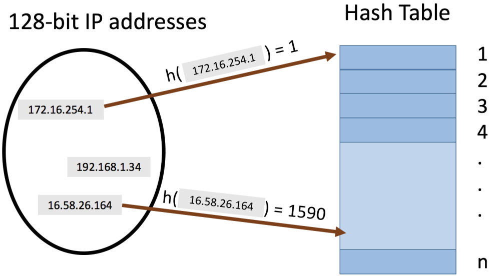
Classic Solution: Hash tables

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

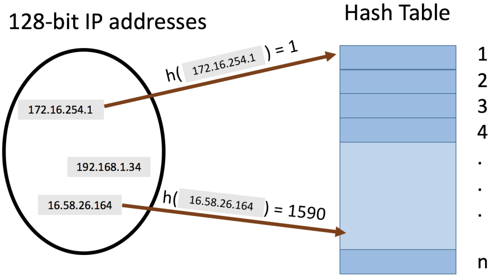
Goal: support $query(x)$ to check if x is in the set in $O(1)$ time.

Classic Solution: Hash tables

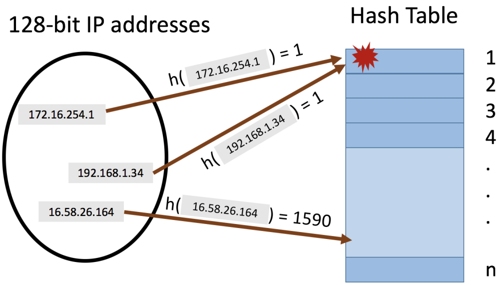
- *Static hashing* since we won't worry about insertion and deletion today.



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

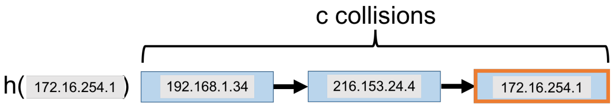


- **hash function** $h : U \rightarrow [n]$ maps elements from the universe to indices $1, \cdots, n$ of an array.
- Typically $|U| \gg n$. Many elements map to the same index.

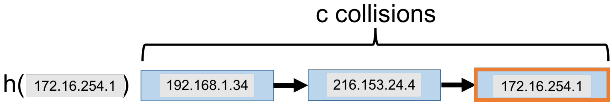


- **hash function** $h : U \rightarrow [n]$ maps elements from the universe to indices $1, \cdots, n$ of an array.
- Typically $|U| \gg n$. Many elements map to the same index.
- **Collisions:** when we insert m items into the hash table we may have to store multiple items in the same location (typically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

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).

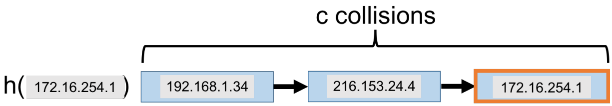


Query runtime: $O(c)$ when the maximum number of collisions in a table entry is c (i.e., must traverse a linked list of size c).



How Can We Bound c?

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). In the best case, $c \approx m/n$.

Let $\mathbf{h} : U \to [n]$ be a 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 $x \neq y$.

Let $\mathbf{h} : U \to [n]$ be a 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 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'.

LINEARITY OF EXPECTATION

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

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

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

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

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

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

LINEARITY OF EXPECTATION

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

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

For any pair i, j, $i \neq j$:

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

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

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

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

For any pair i, j, $i \neq j$:

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

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

LINEARITY OF EXPECTATION

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

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

For any pair i, j, $i \neq j$:

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

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

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

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

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

For any pair i, j, $i \neq j$:

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

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

Identical to the CAPTCHA analysis!

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

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

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

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

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

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

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

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

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

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

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

Apply Markov's Inequality:

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

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

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

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

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

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

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

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

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

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

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

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

$$\Pr[\mathbf{C} = 0] = 1 - \Pr[\mathbf{C} \geq 1]$$

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

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

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

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

$$\Pr[\mathbf{C} = 0] = 1 - \Pr[\mathbf{C} \geq 1] \geq 1 - \frac{1}{8}$$

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

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

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

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

$$\Pr[\mathbf{C} = 0] = 1 - \Pr[\mathbf{C} \geq 1] \geq 1 - \frac{1}{8} = \frac{7}{8}.$$

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

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

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

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

$$\Pr[\mathbf{C} = 0] = 1 - \Pr[\mathbf{C} \geq 1] \geq 1 - \frac{1}{8} = \frac{7}{8}.$$

Pretty good but we are using $O(m^2)$ space to store m items.

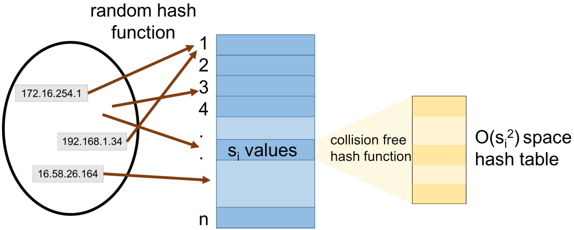
> m: total number of stored items, n: hash table size, $\mathbf{C}$: total pairwise collisions in table.

TWO LEVEL HASHING

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

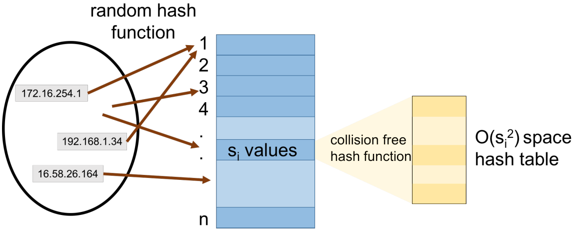
Want to preserve $O(1)$ query time while using $O(m)$ space.

Two-Level Hashing:



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

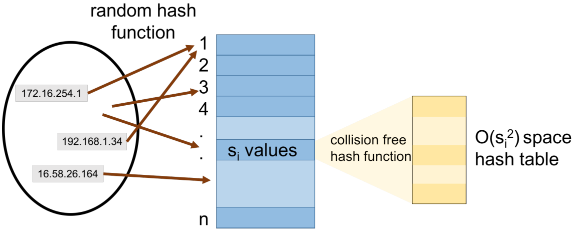
Two-Level Hashing:



- 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 $[s_i] \to [s_i^2]$.
- **Just Showed:** A random function is collision free with probability $\geq \frac{7}{8}$ so only requires checking $O(1)$ random functions in expectation to find a collision free one.

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.

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.

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

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.

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.

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

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

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

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

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

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

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

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

- 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.

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

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.

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?

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

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.

SPACE USAGE

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

- 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

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

- 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

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

- 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

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

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

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

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

- 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 $n = m$)

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

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

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

- 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 $n = m$)

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

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.

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

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

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, **h**: random hash function, **S**: space usage of two level hashing, $\mathbf{s}_i$: # items stored at pos i.

What if we want to store a set and answer membership queries in $O(1)$ time. But we allow a small probability of a false positive: $query(x)$ says that x is in the set when in fact it isn't.

What if we want to store a set and answer membership queries in $O(1)$ time. But we allow a small probability of a false positive: $query(x)$ says that x is in the set when in fact it isn't.

Can we use even smaller space?

What if we want to store a set and answer membership queries in $O(1)$ time. But we allow a small probability of a false positive: $query(x)$ says that x is in the set when in fact it isn't.

Can we use even smaller space?

Many Applications:

- Filter spam email addresses, phone numbers, suspect IPs, duplicate Tweets.
- Quickly check if an item has been stored in a cache or is new.
- Counting distinct elements (e.g., unique search queries.)

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:

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

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

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

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

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

Efficient Alternative: Let p be a prime with $p \geq |U|$. Choose random $\mathbf{a}, \mathbf{b} \in [p]$ with $\mathbf{a} \neq 0$. 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 $i \in [n]$:

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

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 $i \in [n]$:

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

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 $i \in [n]$:

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

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

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

Another common requirement for a hash function:

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

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

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

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

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:

k-wise Independent Hash Function. A random hash function from $\mathbf{h} : U \to [n]$ is k-wise independent if for all $i \in [n]$:

$$\Pr[\mathbf{h}(x_1) = \mathbf{h}(x_2) = \ldots = \mathbf{h}(x_k) = i] = \frac{1}{n^k}.$$

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

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

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