CSE 312
Foundations of Computing II

Lecture 12: Finish up Bloom Filters, Zoo of Discrete RVs, part I

Slido.com/4694375
Agenda

• Bloom Filters Recap & Analysis
• Zoo of Discrete RVs
  – Uniform Random Variables
  – Bernoulli Random Variables
  – Binomial Random Variables
  – Geometric Random Variables
Basic Problem

**Problem:** Store a subset $S$ of a large set $U$.

**Example.** $U = \text{set of 128 bit strings}$
$S = \text{subset of strings of interest}$

$$|U| \approx 2^{128}$$
$$|S| \approx 1000$$

Two goals:
1. **Very fast** (ideally constant time) answers to queries “Is $x \in S$?” for any $x \in U$.
2. **Minimal storage** requirements.
Bloom Filters

- Stores information about a set of elements $S \subseteq U$.
- Supports two operations:
  1. $\text{add}(x)$ - adds $x \in U$ to the set $S$
  2. $\text{contains}(x)$ – ideally: true if $x \in S$, false otherwise

Possible false positives

Combine with fallback mechanism – can distinguish false positives from true positives with extra cost
Bloom Filters – Ingredients

<table>
<thead>
<tr>
<th></th>
<th>t₁</th>
<th>t₂</th>
<th>t₃</th>
</tr>
</thead>
<tbody>
<tr>
<td>t₁</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>t₂</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>t₃</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

Basic data structure is a $k \times m$ binary array “the Bloom filter”

- $k$ rows $t₁, \ldots, tₖ$, each of size $m$
- Think of each row as an $m$-bit vector

$k$ different hash functions $h₁, \ldots, hₖ : U \rightarrow [m]$
Bloom Filters – Three operations

• Set up Bloom filter for \( S = \emptyset \)

  \[
  \text{function INITIALIZE}(k, m) \\
  \text{for } i = 1, \ldots, k: \text{ do} \\
  t_i = \text{new bit vector of } m \text{ 0s}
  \]

• Update Bloom filter for \( S \leftarrow S \cup \{x\} \)

  \[
  \text{function ADD}(x) \\
  \text{for } i = 1, \ldots, k: \text{ do} \\
  t_i[h_i(x)] = 1
  \]

• Check if \( x \in S \)

  \[
  \text{function CONTAINS}(x) \\
  \text{return } t_1[h_1(x)] = 1 \land t_2[h_2(x)] = 1 \land \ldots \land t_k[h_k(x)] = 1
  \]
function INITIALIZE($k, m$)
    for $i = 1, \ldots, k$: do
        $t_i =$ new bit vector of $m$ 0s

Number of hash functions

Size of array associated to each hash function.

for each hash function, initialize an empty bit vector of size $m$
Bloom Filters: Example

Bloom filter $t$ of length $m = 5$ that uses $k = 3$ hash functions

**function** \textsc{initialize}($k, m$) \\
\textbf{for} $i = 1, \ldots, k$: \textbf{do} \\
\hspace{1em} $t_i$ = new bit vector of $m$ 0s

<table>
<thead>
<tr>
<th>Index $\rightarrow$</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>$t_1$</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>$t_2$</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>$t_3$</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>
Bloom Filters: Add

function ADD(x)

for i = 1, ..., k: do

t_i[h_i(x)] = 1

for each hash function h_i

Index into i-th bit-vector, at index produced by hash function and set to 1

h_i(x) → result of hash function h_i on x
Bloom Filters: Example

Bloom filter t of length $m = 5$ that uses $k = 3$ hash functions

**function** `ADD(x)`

for $i = 1, \ldots, k$: do

$t_i[h_i(x)] = 1$

```plaintext
add("thisisavirus.com")

$h_1("thisisavirus.com") \rightarrow 2$
```

<table>
<thead>
<tr>
<th>Index</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>$t_1$</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>$t_2$</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>$t_3$</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>
Bloom Filters: Example

Bloom filter $t$ of length $m = 5$ that uses $k = 3$ hash functions

add(“thisisavirus.com”)

$h_1(“thisisavirus.com”) → 2$
$h_2(“thisisavirus.com”) → 1$

function $ADD(x)$

for $i = 1, \ldots, k$: do

t$_i[h_i(x)] = 1$

<table>
<thead>
<tr>
<th>Index $→$</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>$t_1$</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>$t_2$</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>$t_3$</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>
Bloom Filters: Example

Bloom filter \( t \) of length \( m = 5 \) that uses \( k = 3 \) hash functions

**function** `ADD(x)`

for \( i = 1, \ldots, k \) do

\[ t_i[h_i(x)] = 1 \]

add("thisisavirus.com")

\[ h_1("thisisavirus.com") \rightarrow 2 \]

\[ h_2("thisisavirus.com") \rightarrow 1 \]

\[ h_3("thisisavirus.com") \rightarrow 4 \]

<table>
<thead>
<tr>
<th>Index →</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>( t_1 )</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>( t_2 )</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>( t_3 )</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>
Bloom Filters: Example

Bloom filter \( t \) of length \( m = 5 \) that uses \( k = 3 \) hash functions

**function** \( \text{ADD}(x) \)

\[
\text{for } i = 1, \ldots, k: \text{ do} \\
\quad t_i[h_i(x)] = 1
\]

```
add("thisisavirus.com")
```

<table>
<thead>
<tr>
<th>Index ( \rightarrow )</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>( t_1 )</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>( t_2 )</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>( t_3 )</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>
Bloom Filters: Contains

\[
\text{function} \ \text{CONTAINS}(x) \\
\text{return} \ t_1[h_1(x)] == 1 \land t_2[h_2(x)] == 1 \land \cdots \land t_k[h_k(x)] == 1
\]

Returns True if the bit vector \( t_i \) for each hash function has bit 1 at index determined by \( h_i(x) \),

Returns False otherwise
Bloom Filters: Example

Bloom filter \( t \) of length \( m = 5 \) that uses \( k = 3 \) hash functions

```plaintext
function contains(x)
    return \( t_1[h_1(x)] = 1 \land t_2[h_2(x)] = 1 \land \cdots \land t_k[h_k(x)] = 1 \)
```

contains("thisisavirus.com")

<table>
<thead>
<tr>
<th>Index ( \rightarrow )</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>( t_1 )</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>( t_2 )</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>( t_3 )</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>
Bloom Filters: Example

Bloom filter $t$ of length $m = 5$ that uses $k = 3$ hash functions

```
function CONTAINS(x)
    return $t_1[h_1(x)] = 1 \land t_2[h_2(x)] = 1 \land \ldots \land t_k[h_k(x)] = 1$
```

contains("thisisavirus.com")

$h_1("thisisavirus.com") \rightarrow 2$

True

<table>
<thead>
<tr>
<th>Index</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>$t_1$</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>$t_2$</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>$t_3$</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>
Bloom Filters: Example

Bloom filter $t$ of length $m = 5$ that uses $k = 3$ hash functions.

```plaintext
function CONTAINS(x)
    return $t_1[h_1(x)] = 1 \land t_2[h_2(x)] = 1 \land \cdots \land t_k[h_k(x)] = 1$
```

contains("thisisavirus.com")

$h_1("thisisavirus.com") \rightarrow 2$
$h_2("thisisavirus.com") \rightarrow 1$

<table>
<thead>
<tr>
<th>Index →</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>$t_1$</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>$t_2$</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>$t_3$</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>
Bloom Filters: Example

Bloom filter \( t \) of length \( m = 5 \) that uses \( k = 3 \) hash functions

**Function**

\[
\text{contains}(x) = \begin{cases} 
\text{True} & \text{if } t_1[h_1(x)] = 1 \land t_2[h_2(x)] = 1 \land \cdots \land t_k[h_k(x)] = 1 
\end{cases}
\]

- \( h_1(\text{"thisisavirus.com"}) \rightarrow 2 \)
- \( h_2(\text{"thisisavirus.com"}) \rightarrow 1 \)
- \( h_3(\text{"thisisavirus.com"}) \rightarrow 4 \)

<table>
<thead>
<tr>
<th>Index ( \rightarrow )</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>( t_1 )</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>( t_2 )</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>( t_3 )</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>
**Bloom Filters: Example**

Bloom filter $t$ of length $m = 5$ that uses $k = 3$ hash functions

```python
function CONTAINS(x)
    return $t_1[h_1(x)] = 1 \land t_2[h_2(x)] = 1 \land \cdots \land t_k[h_k(x)] = 1$
```

Table for Bloom filter:

<table>
<thead>
<tr>
<th>Index</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>$t_1$</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>$t_2$</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>$t_3$</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

Since all conditions satisfied, returns True (correctly)

Contains("thisisavirus.com")

- $h_1("thisisavirus.com")$ → 2
- $h_2("thisisavirus.com")$ → 1
- $h_3("thisisavirus.com")$ → 4
Bloom Filters: False Positives

Bloom filter $t$ of length $m = 5$ that uses $k = 3$ hash functions

add(“totallynotsuspicious.com”)

**function** $\text{ADD}(x)$

for $i = 1, \ldots, k$: do

$t_i[h_i(x)] = 1$

<table>
<thead>
<tr>
<th>Index →</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>$t_1$</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>$t_2$</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>$t_3$</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>
Bloom Filters: False Positives

Bloom filter $t$ of length $m = 5$ that uses $k = 3$ hash functions

$$\text{add}(\text{"totallynotsuspicious.com")}$$

$$h_1(\text{"totallynotsuspicious.com")} \rightarrow 1$$

$$\text{function } \text{ADD}(x)\text{ for } i = 1, \ldots, k:\text{ do}$$

$$t_i[h_i(x)] = 1$$

<table>
<thead>
<tr>
<th>Index $\rightarrow$</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>$t_1$</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>$t_2$</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>$t_3$</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>
Bloom Filters: False Positives

Bloom filter $t$ of length $m = 5$ that uses $k = 3$ hash functions

**function** $\text{ADD}(x)$

for $i = 1, \ldots, k$: do

$t_i[h_i(x)] = 1$

add(“totallynotsuspicious.com”)

$h_1(“totallynotsuspicious.com”) \rightarrow 1$

$h_2(“totallynotsuspicious.com”) \rightarrow 0$

<table>
<thead>
<tr>
<th>Index $\rightarrow$</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>$t_1$</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>$t_2$</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>$t_3$</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>
**Bloom Filters: False Positives**

Bloom filter $t$ of length $m = 5$ that uses $k = 3$ hash functions

**function** \( \text{ADD}(x) \)
\[
\text{for } i = 1, \ldots, k: \text{ do}
\]
\[
t_i[h_i(x)] = 1
\]

\[
\text{add(“totallynotsuspicious.com”)}
\]

\[
h_1(“totallynotsuspicious.com”) \rightarrow 1
\]
\[
h_2(“totallynotsuspicious.com”) \rightarrow 0
\]
\[
h_3(“totallynotsuspicious.com”) \rightarrow 4
\]

<table>
<thead>
<tr>
<th>Index →</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>$t_1$</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>$t_2$</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>$t_3$</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>
Bloom Filters: False Positives

Bloom filter $t$ of length $m = 5$ that uses $k = 3$ hash functions

**function** $\text{ADD}(x)$

for $i = 1, \ldots, k$: do

\[
t_i[h_i(x)] = 1
\]

add(“totallynotsuspicious.com”)

- $h_1(“totallynotsuspicious.com”) \rightarrow 1$
- $h_2(“totallynotsuspicious.com”) \rightarrow 0$
- $h_3(“totallynotsuspicious.com”) \rightarrow 4$

<table>
<thead>
<tr>
<th>Index $\rightarrow$</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>$t_1$</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>$t_2$</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>$t_3$</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>
Bloom Filters: False Positives

Bloom filter $t$ of length $m = 5$ that uses $k = 3$ hash functions

```
function contains(x)
    return $t_1[h_1(x)] \land t_2[h_2(x)] \land \cdots \land t_k[h_k(x)] = 1$
```

contains(“verynormalsite.com”)

<table>
<thead>
<tr>
<th>Index $\rightarrow$</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>$t_1$</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>$t_2$</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>$t_3$</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>
Bloom Filters: False Positives

Bloom filter $t$ of length $m = 5$ that uses $k = 3$ hash functions

\[
\text{function } \text{CONTAINS}(x) \\
\text{return } t_1[h_1(x)] == 1 \land t_2[h_2(x)] == 1 \land \cdots \land t_k[h_k(x)] == 1
\]

contains(“verynormalsite.com”) = True

\[
h_1(“verynormalsite.com”) \rightarrow 2
\]

<table>
<thead>
<tr>
<th>Index</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>$t_1$</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>$t_2$</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>$t_3$</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>
Bloom Filters: False Positives

Bloom filter \( t \) of length \( m = 5 \) that uses \( k = 3 \) hash functions

\[
\text{function } \text{CONTAINS}(x) \\
\text{return } t_1[h_1(x)] == 1 \land t_2[h_2(x)] == 1 \land \cdots \land t_k[h_k(x)] == 1
\]

\( h_1(“verynormalsite.com”) \rightarrow 2 \)
\( h_2(“verynormalsite.com”) \rightarrow 0 \)

\[
\begin{array}{|c|c|c|c|c|c|}
\hline
\text{Index} & 0 & 1 & 2 & 3 & 4 \\
\hline
\rightarrow & & & & & \\
\hline
\text{t}_1 & 0 & 1 & 1 & 0 & 0 \\
\hline
\text{t}_2 & 1 & 1 & 0 & 0 & 0 \\
\hline
\text{t}_3 & 0 & 0 & 0 & 0 & 1 \\
\hline
\end{array}
\]
Bloom Filters: False Positives

Bloom filter $t$ of length $m = 5$ that uses $k = 3$ hash functions

<table>
<thead>
<tr>
<th>Index</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>$t_1$</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>$t_2$</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>$t_3$</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

contains(“verynormalsite.com”)

$h_1(“verynormalsite.com”) \rightarrow 2$

$h_2(“verynormalsite.com”) \rightarrow 0$

$h_3(“verynormalsite.com”) \rightarrow 4$

function \( \textsc{contains}(x) \)
return \( t_1[h_1(x)] = 1 \land t_2[h_2(x)] = 1 \land \cdots \land t_k[h_k(x)] = 1 \)
Bloom Filters: False Positives

Bloom filter $t$ of length $m = 5$ that uses $k = 3$ hash functions

```python
def contains(x):
    return \( t_1[h_1(x)] = 1 \land t_2[h_2(x)] = 1 \land \cdots \land t_k[h_k(x)] = 1 \)
```

```
<table>
<thead>
<tr>
<th>Index</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>( t_1 )</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>( t_2 )</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>( t_3 )</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>
```

Since all conditions satisfied, returns **True** (incorrectly)

`contains(“verynormalsite.com”)`

- \( h_1(“verynormalsite.com”) \rightarrow 2 \)
- \( h_2(“verynormalsite.com”) \rightarrow 0 \)
- \( h_3(“verynormalsite.com”) \rightarrow 4 \)
Analysis: False positive probability

**Question:** For an element $x \in U$, what is the probability that \texttt{contains}(x) returns true if \texttt{add}(x) was never executed before?

Probability over what?! Over the choice of the $h_1, \ldots, h_k$

Assumptions for the analysis (somewhat stronger than for ordinary hashing):

- Each $h_i(x)$ is uniformly distributed in $[m]$ for all $x$ and $i$.
- Hash function outputs for each $h_i$ are mutually independent (not just in pairs).
- Different hash functions are independent of each other.
False positive probability – Events

Assume we perform \( \text{add}(x_1), \ldots, \text{add}(x_n) \)
+ \text{contains}(x) \text{ for } x \notin \{x_1, \ldots, x_n\}

Event \( E_i \) holds iff \( h_i(x) \in \{h_i(x_1), \ldots, h_i(x_n)\} \)

\[
P(\text{false positive}) = P(E_1 \cap E_2 \cap \cdots \cap E_k) = \prod_{i=1}^{k} P(E_i)
\]

\( h_1, \ldots, h_k \) independent
False positive probability – Events

Event $E_i$ holds iff $h_i(x) \in \{h_i(x_1), \ldots, h_i(x_n)\}$

Event $E_i^c$ holds iff $h_i(x) \neq h_i(x_1)$ and ... and $h_i(x) \neq h_i(x_n)$

$$P(E_i^c) = \sum_{z=1}^{m} P(h_i(x) = z) \cdot P(E_i^c \mid h_i(x) = z)$$

LTP
False positive probability – Events

$P(E_i^c \mid h_i(x) = z) = P(h_i(x_1) \neq z, \ldots, h_i(x_n) \neq z \mid h_i(x) = z)$

$= P(h_i(x_1) \neq z, \ldots, h_i(x_n) \neq z)$

$= \prod_{j=1}^{n} P(h_i(x_j) \neq z)$

Event $E_i^c$ holds iff $h_i(x) \neq h_i(x_1)$ and ... and $h_i(x) \neq h_i(x_n)$

Independence of values of $h_i$ on different inputs
False positive probability – Events

Event \( E_i^C \) holds iff \( h_i(x) \neq h_i(x_1) \) and ... and \( h_i(x) \neq h_i(x_n) \)

\[
P(E_i^C \mid h_i(x) = z) = P(h_i(x_1) \neq z, \ldots, h_i(x_n) \neq z \mid h_i(x) = z)
\]

= \( P(h_i(x_1) \neq z, \ldots, h_i(x_n) \neq z) \)

= \( \prod_{j=1}^{n} P(h_i(x_j) \neq z) \)

= \( \prod_{j=1}^{n} (1 - \frac{1}{m}) = \left(1 - \frac{1}{m}\right)^n \)

\[
P(E_i^C) = \sum_{z=1}^{m} P(h_i(x) = z) \cdot P(E_i^C \mid h_i(x) = z) = \]

Independence of values of \( h_i \) on different inputs

Outputs of \( h_i \) uniformly spread
False positive probability – Events

Event $E_i^c$ holds iff $h_i(x) \neq h_i(x_1)$ and ... and $h_i(x) \neq h_i(x_n)$

$$P(E_i^c| h_i(x) = z) = P(h_i(x_1) \neq z, ..., h_i(x_n) \neq z | h_i(x) = z)$$

Independence of values of $h_i$ on different inputs

$$= P(h_i(x_1) \neq z, ..., h_i(x_n) \neq z)$$

$$= \prod_{j=1}^{n} P(h_i(x_j) \neq z)$$

Outputs of $h_i$ uniformly spread

$$= \prod_{j=1}^{n} \left(1 - \frac{1}{m}\right) = \left(1 - \frac{1}{m}\right)^n$$

$$P(E_i^c) = \sum_{z=1}^{m} P(h_i(x) = z) \cdot P(E_i^c| h_i(x) = z) = \left(1 - \frac{1}{m}\right)^n$$
False positive probability – Events

Event $E_i$ holds iff $\mathbf{h}_i(x) \in \{\mathbf{h}_i(x_1), \ldots, \mathbf{h}_i(x_n)\}$

Event $E^c_i$ holds iff $\mathbf{h}_i(x) \neq \mathbf{h}_i(x_1)$ and ... and $\mathbf{h}_i(x) \neq \mathbf{h}_i(x_n)$

$$P(E^c_i) = \left(1 - \frac{1}{m}\right)^n$$

$$\text{FPR} = \prod_{i=1}^{k} (1 - P(E^c_i)) =$$
False Positivity Rate – Example

\[ FPR = \left( 1 - \left( 1 - \frac{1}{m} \right)^n \right)^k \]

e.g., \( n = 5,000,000 \)
\( k = 30 \)
\( m = 2,500,000 \)

\[ FPR = 1.28\% \]
**Comparison with Hash Tables - Space**

- Google storing 5 million URLs, each URL 40 bytes.
- Bloom filter with $k = 30$ and $m = 2,500,000$

<table>
<thead>
<tr>
<th>Hash Table</th>
<th>Bloom Filter</th>
</tr>
</thead>
<tbody>
<tr>
<td>(optimistic)</td>
<td></td>
</tr>
<tr>
<td>$5,000,000 \times 40B = 200\text{MB}$</td>
<td>$2,500,000 \times 30 = 75,000,000 \text{bits}$</td>
</tr>
<tr>
<td></td>
<td>$&lt; 10 \text{MB}$</td>
</tr>
</tbody>
</table>
**Time**

- Say avg user visits **102,000** URLs in a year, of which **2,000** are malicious.
- **0.5** seconds to do lookup in the database, **1ms** for lookup in Bloom filter.
- Suppose the false positive rate is **3%**

\[
\text{false positives} = \frac{100000 \times 0.03 \times 500\text{ms}}{102000} + 2000 \times 500\text{ms} \approx 25.51\text{ms}
\]
Bloom Filters typical of...

... randomized algorithms and randomized data structures.

- Simple
- Fast
- Efficient
- Elegant
- Useful!
CSE 312
Foundations of Computing II
Zoo of Discrete RVs, part I

Slido.com/4694375
Motivation for “Named” Random Variables

Random Variables that show up all over the place.
- Easily solve a problem by recognizing it’s a special case of one of these random variables.

Each RV introduced today will show:
- A general situation it models
- Its name and parameters
- Its PMF, Expectation, and Variance
- Example scenarios you can use it
### Welcome to the Zoo! (Preview)

<table>
<thead>
<tr>
<th>Distribution</th>
<th>Probability Mass Function</th>
<th>Mean</th>
<th>Variance</th>
</tr>
</thead>
<tbody>
<tr>
<td>$X \sim \text{Unif}(a, b)$</td>
<td>$P(X = k) = \frac{1}{b - a + 1}$</td>
<td>$E[X] = \frac{a + b}{2}$</td>
<td>$\text{Var}(X) = \frac{(b - a)(b - a + 2)}{12}$</td>
</tr>
<tr>
<td>$X \sim \text{Ber}(p)$</td>
<td>$P(X = 1) = p, P(X = 0) = 1 - p$</td>
<td>$E[X] = p$</td>
<td>$\text{Var}(X) = p(1 - p)$</td>
</tr>
<tr>
<td>$X \sim \text{Bin}(n, p)$</td>
<td>$P(X = k) = \binom{n}{k} p^k (1 - p)^{n-k}$</td>
<td>$E[X] = np$</td>
<td>$\text{Var}(X) = np(1 - p)$</td>
</tr>
<tr>
<td>$X \sim \text{Geo}(p)$</td>
<td>$P(X = k) = (1 - p)^{k-1}p$</td>
<td>$E[X] = \frac{1}{p}$</td>
<td>$\text{Var}(X) = \frac{1 - p}{p^2}$</td>
</tr>
<tr>
<td>$X \sim \text{NegBin}(r, p)$</td>
<td>$P(X = k) = \binom{k-1}{r-1} p^r (1 - p)^{k-r}$</td>
<td>$E[X] = \frac{r}{p}$</td>
<td>$\text{Var}(X) = \frac{r(1 - p)}{p^2}$</td>
</tr>
<tr>
<td>$X \sim \text{HypGeo}(N, K, n)$</td>
<td>$P(X = k) = \frac{K}{n} \binom{N - K}{n - k} \binom{N}{n}$</td>
<td>$E[X] = n \frac{K}{N}$</td>
<td>$\text{Var}(X) = n \frac{K(N - K)(N - n)}{N^2(N - 1)}$</td>
</tr>
</tbody>
</table>
Agenda

• Bloom Filters Example & Analysis
• Zoo of Discrete RVs, Part I
  – Uniform Random Variables
  – Bernoulli Random Variables
  – Binomial Random Variables
  – Geometric Random Variables
Discrete Uniform Random Variables

A discrete random variable $X$ equally likely to take any (integer) value between integers $a$ and $b$ (inclusive), is uniform.

Notation:

PMF:

Expectation:

Variance:

Example: value shown on one roll of a fair die is $\text{Unif}(1,6)$:

- $P(X = i) = 1/6$
- $\mathbb{E}[X] = 7/2$
- $\text{Var}(X) = 35/12$
Discrete Uniform Random Variables

A discrete random variable $X$ equally likely to take any (integer) value between integers $a$ and $b$ (inclusive), is uniform.

Notation: $X \sim \text{Unif}(a, b)$

PMF: $P(X = i) = \frac{1}{b - a + 1}$

Expectation: $\mathbb{E}[X] = \frac{a + b}{2}$

Variance: $\text{Var}(X) = \frac{(b - a)(b - a + 2)}{12}$

Example: value shown on one roll of a fair die is $\text{Unif}(1, 6)$:
- $P(X = i) = 1/6$
- $\mathbb{E}[X] = 7/2$
- $\text{Var}(X) = 35/12$
Agenda

• Bloom Filters Example & Analysis
• Zoo of Discrete RVs, Part I
  – Uniform Random Variables
  – Bernoulli Random Variables
  – Binomial Random Variables
  – Geometric Random Variables
Bernoulli Random Variables

A random variable $X$ that takes value 1 ("Success") with probability $p$, and 0 ("Failure") otherwise. $X$ is called a Bernoulli random variable.

Notation: $X \sim \text{Ber}(p)$

PMF: $P(X = 1) = p$, $P(X = 0) = 1 - p$

Expectation:

Variance:

Poll:
Slido.com/4694375

<table>
<thead>
<tr>
<th>Mean</th>
<th>Variance</th>
</tr>
</thead>
<tbody>
<tr>
<td>A.</td>
<td>$p$</td>
</tr>
<tr>
<td>B.</td>
<td>$p$</td>
</tr>
<tr>
<td>C.</td>
<td>$p$</td>
</tr>
<tr>
<td>D.</td>
<td>$p$</td>
</tr>
</tbody>
</table>
Bernoulli Random Variables

A random variable $X$ that takes value 1 (“Success”) with probability $p$, and 0 (“Failure”) otherwise. $X$ is called a Bernoulli random variable.

Notation: $X \sim \text{Ber}(p)

PMF: $P(X = 1) = p$, $P(X = 0) = 1 - p$

Expectation: $\mathbb{E}[X] = p$  
Note: $\mathbb{E}[X^2] = p$

Variance: $\text{Var}(X) = \mathbb{E}[X^2] - \mathbb{E}[X]^2 = p - p^2 = p(1 - p)$

Examples:
- Coin flip
- Randomly guessing on a MC test question
- A server in a cluster fails
- Whether or not a particular share of a particular stock pays off or not
- Any indicator r.v.
Agenda

• Bloom Filters Example & Analysis
• Zoo of Discrete RVs, Part I
  – Uniform Random Variables
  – Bernoulli Random Variables
  – Binomial Random Variables
  – Geometric Random Variables
Binomial Random Variables

A discrete random variable \( X = \sum_{i=1}^{n} Y_i \) where each \( Y_i \sim \text{Ber}(p) \).
Counts number of successes in \( n \) independent trials, each with probability \( p \) of success.
\( X \) is a Binomial random variable

Examples:
- # of heads in \( n \) indep coin flips
- # of 1s in a randomly generated \( n \) bit string
- # of servers that fail in a cluster of \( n \) computers
- # of bit errors in file written to disk
- # of elements in a bucket of a large hash table
- # of \( n \) different stocks that “pay off”

Poll:

Slido.com/4694375

\( P(X = k) = \)

A. \( p^k (1 - p)^{n-k} \)
B. \( np \)
C. \( \binom{n}{k} p^k (1 - p)^{n-k} \)
D. \( \binom{n}{n-k} p^k (1 - p)^{n-k} \)
Binomial Random Variables

A discrete random variable $X = \sum_{i=1}^{n} Y_i$ where each $Y_i \sim \text{Ber}(p)$. Counts number of successes in $n$ independent trials, each with probability $p$ of success.

$X$ is a Binomial random variable

Notation: $X \sim \text{Bin}(n,p)$

PMF: $P(X = k) = \binom{n}{k} p^k (1 - p)^{n-k}$

Expectation:

Variance:

Poll:

<table>
<thead>
<tr>
<th>Mean</th>
<th>Variance</th>
</tr>
</thead>
<tbody>
<tr>
<td>A. $p$</td>
<td>$p$</td>
</tr>
<tr>
<td>B. $np$</td>
<td>$np(1 - p)$</td>
</tr>
<tr>
<td>C. $np$</td>
<td>$np^2$</td>
</tr>
<tr>
<td>D. $np$</td>
<td>$n^2 p$</td>
</tr>
</tbody>
</table>

Slido.com/4694375
Binomial Random Variables

A discrete random variable \( X = \sum_{i=1}^{n} Y_i \) where each \( Y_i \sim \text{Ber}(p) \).
Counts number of successes in \( n \) independent trials, each with probability \( p \) of success.
\( X \) is a Binomial random variable

Notation: \( X \sim \text{Bin}(n, p) \)

PMF: \( P(X = k) = \binom{n}{k} p^k (1 - p)^{n-k} \)

Expectation: \( \mathbb{E}[X] = np \)

Variance: \( \text{Var}(X) = np(1 - p) \)
Mean, Variance of the Binomial

“i.i.d.” is a commonly used phrase. It means “independent & identically distributed”

If $Y_1, Y_2, \ldots, Y_n \sim \text{Ber}(p)$ and independent (i.i.d.), then

$X = \sum_{i=1}^{n} Y_i$, \quad X \sim \text{Bin}(n, p)$

Claim $\mathbb{E}[X] = np$

$$\mathbb{E}[X] = \mathbb{E}\left[\sum_{i=1}^{n} Y_i\right] = \sum_{i=1}^{n} \mathbb{E}[Y_i] = n\mathbb{E}[Y_1] = np$$

Claim $\text{Var}(X) = np(1 - p)$

$$\text{Var}(X) = \text{Var}\left(\sum_{i=1}^{n} Y_i\right) = \sum_{i=1}^{n} \text{Var}(Y_i) = n\text{Var}(Y_1) = np(1 - p)$$
Binomial PMFs

PMF for $X \sim \text{Bin}(10, 0.5)$

PMF for $X \sim \text{Bin}(10, 0.25)$
Binomial PMFs

PMF for $X \sim \text{Bin}(30,0.5)$

PMF for $X \sim \text{Bin}(30,0.1)$
Agenda

- Bloom Filters Example & Analysis
- Zoo of Discrete RVs, Part I
  - Uniform Random Variables
  - Bernoulli Random Variables
  - Binomial Random Variables
  - Geometric Random Variables
Geometric Random Variables

A discrete random variable $X$ that models the number of independent trials $Y_i \sim \text{Ber}(p)$ before seeing the first success.

$X$ is called a Geometric random variable with parameter $p$.

**Notation:** $X \sim \text{Geo}(p)$

**PMF:**

**Expectation:**

**Variance:**

**Examples:**
- # of coin flips until first head
- # of random guesses on MC questions until you get one right
- # of random guesses at a password until you hit it
Geometric Random Variables

A discrete random variable $X$ that models the number of independent trials $Y_i \sim \text{Ber}(p)$ before seeing the first success. $X$ is called a geometric random variable with parameter $p$.

Notation: $X \sim \text{Geo}(p)$

PMF: $P(X = k) = (1 - p)^{k-1}p$

Expectation: $\mathbb{E}[X] = \frac{1}{p}$

Variance: $\text{Var}(X) = \frac{1 - p}{p^2}$

Examples:

- # of coin flips until first head
- # of random guesses on MC questions until you get one right
- # of random guesses at a password until you hit it
Agenda

- Bloom Filters Example & Analysis
- Zoo of Discrete RVs, Part I
  - Uniform Random Variables
  - Bernoulli Random Variables
  - Binomial Random Variables
  - Geometric Random Variables
  - More examples
Example

Sending a binary message of length 1024 bits over a network with probability 0.999 of correctly sending each bit in the message without corruption (independent of other bits).

Let $X$ be the number of corrupted bits.

What kind of random variable is this and what is $\mathbb{E}[X]$?

Poll:

[Slido.com/4694375]

a. 1022.99
b. 1.024
c. 1.02298
d. 1
Example: Music Lessons

Your music teacher requires you to play a 1000 note song without mistake. You have been practicing, so you have a probability of 0.999 of getting each note correct (independent of the others). If you mess up a single note in the song, you must start over and play from the beginning. Let $X$ be the number of times you have to play the song from the start. What kind of random variable is this and what is $E[X]$?