# Randomized Project - Bloom Filter

Stefan Blair
Christopher Jones
Sabbir Rashid

Rensselaer Polytechnic Institute

December 3, 2018

# Outline

# Introduction

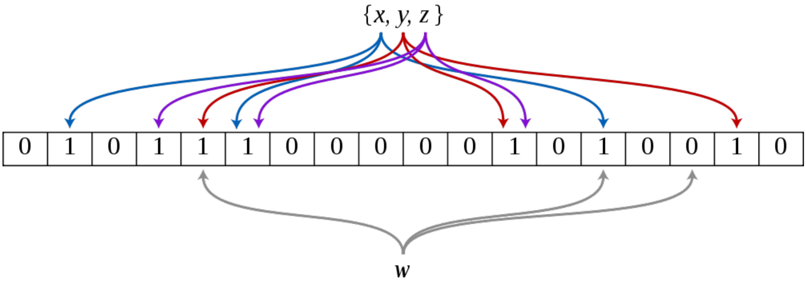
- Enhancing Collaborative Spam Detection with Bloom Filters
  - Jeff Yan & Pook Leong Cho (Newcastle University)
  - IEEE 2006 22nd Computer Security Applications Conference

Spam Detection:

- Statistical vs. Signature-based Collaborative (SCSD)
- Using Bloom filters with SCSD:
  - $O(1)$ look-ups
  - no huge database required

# Bloom Filters

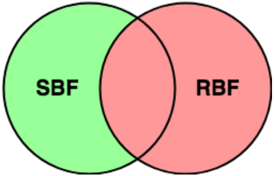
- Array of $m$ bits (all initially set to zero)
- $k$ independent hash functions map to $\{0, \ldots, m-1\}$
- Element $x$ is inserted by setting locations $h_1(x), \ldots, h_k(x)$ to one




- Can introduce false positives
- $h_1(x) = \ldots = h_k(x) = 1$, when actually $x$ was never inserted

# Results of the Paper (Razor)

- Can basically use normal bloom filters, but...
- Razor supports deletion, which requires 2 Bloom filters:
  - Spam Bloom Filter (SBF), Revocation Bloom filter (RBF)




- This achieves constant time look-up

Storing $n$ signatures of length 160 bits:

- *Bloom filters*: $2m$ bits needed
- *Hash table*: at least $160 * n$ bits

# Results of the Paper (DCC)

- Spam is "bulk". Count each time an email is reported
- Threshold value $t$ (e.g. $t = 20$) indicates spam
- Signature $x$ has count values

$$c[h_1(x)], c[h_2(x)], ..., c[h_k(x)]$$

- $\min\{c[h_1(x)], ..., c[h_k(x)]\}$ is an upper bound on the actual number of times $x$ was inserted
- Storage:
  - *Bloom filters*: $m * sizeof(cell)$ bits (e.g. $5m$ bits)
  - *Hash table*: at least $n * sizeof(signature) + n * sizeof(count) bits$ bits

# Results of the Paper (DCC)

Further heuristics when inserting signature $x$ to decrease the False-Positive rate:

1. Only increment counters that equal $\min\{c[h_1(x)], ..., c[h_k(x)]\}$.
   - **Global coincidental hits**
2. If two or more $h_i(x)$ map to the same cell, only increment that cell once.
   - **Local coincidental hits**

# Variables

- $n$ - Number of elements/emails
- $m$ - Filter size, i.e, the number of bits allowed each encoded email vector
- $k$ - Number of hash functions constructed

# Simulation and Results

They ran simulations to determine how much the 2 heuristics improved the counting Bloom filter

- Instead of actual email signatures, they used random numbers
- 8 experiments of varying insertion sequences for $n = 10,000$ distinct elements, e.g.

$$\underbrace{x_1, x_2, ..., x_{10,000}}_{Round_1}, \;......, \underbrace{x_1, x_2, ..., x_{10,000}}_{Round_{20}} \qquad \underbrace{x_1, ..., x_1}_{20}, \underbrace{x_2, ..., x_2}_{20}, \;...\;..., \underbrace{x_{10,000}, ..., x_{10,000}}_{20}$$

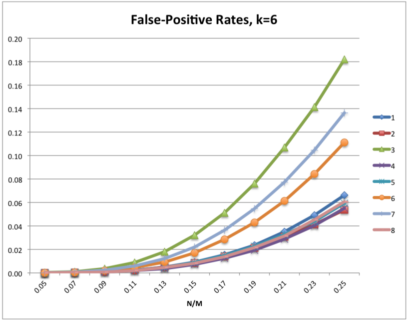
- Configurations for $k = \{4, 6, 8\}$ and $m = \{80K, 160K, 320K\}$

Findings: Heuristics significantly lowered False-Positive rates. Frequency of insertion, as well as order, also affect False-Positive rates.

# Experiments

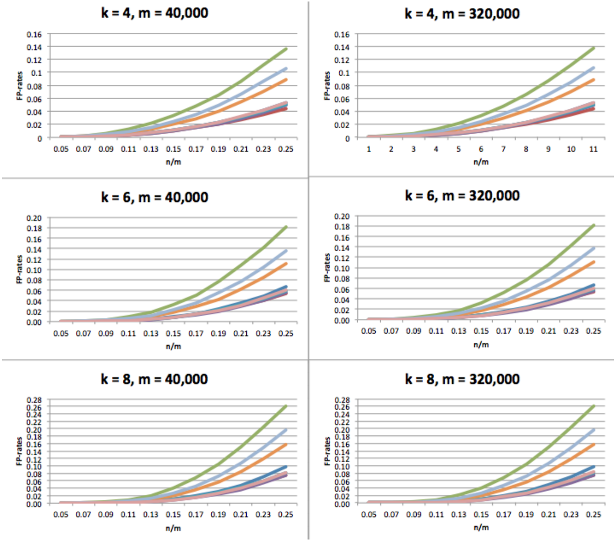
What happens as $n$ gets increasingly large?
At what ratio of $n/m$ do False-Positive rates become undesirable?
Experiment:

- $k = \{4, 6, 8\} \qquad m = \{40K, 80K, 160K, 320K\}$
- $n/m = \{0.05, 0.07, ..., 0.25\}$




$m = 40,000$

# Experiments




- Each colored line represents an experiment
- Tested by varying both $k$ and $m$
- As $n/m$ or $k$ increases, false positives increase

# Applications

- Not necessarily great for spam detection...
- Useful when querying a large database, where looking up/reading data on disk is expensive.
  - Can have a Bloom filter overhead that is first queried to see if the item you are searching for even exists in the database
  - If the Bloom filter says it doesn't exist, then you know you don't have to do an expensive look-up, otherwise, the data you are looking for most likely exists
- Determining whether an email is in your contact list - shorter strings than full emails

# Future Work

- Test experiments on actual spam detection data
  - Spambase - https://archive.ics.uci.edu/ml/datasets/Spambase
  - TREC 2007 Public Corpus - https://plg.uwaterloo.ca/cgi-bin/cgiwrap/gvcormac/foo07
  - Enron - http://www2.aueb.gr/users/ion/data/enron-spam/
  - SMS Spam - http://www.esp.uem.es/jmgomez/smsspamcorpus/ , https://www.kaggle.com/uciml/sms-spam-collection-dataset#spam.csv

# Conclusion

- Discussed Bloom Filters
- Discussed Related Papers
- Described Simulation Setup
- Described Experiments and Results
- Discussed Applications & Future Work

# Questions?