Probabilistic Counting with Randomized Storage

Benjamin Van Durme and Ashwin Lall

UNIVERSITY of ROCHESTER

Data Overload

- Lots of text (, images, audio, ...) is good
- But how to process it all?
- Approximate algorithms!

Make the best of what you've got

Data Overload

More data equals better results

- Lots of text (, images, audio, ...) is good
- But how to process it all?
- Approximate algorithms!

Make the best of what you've got

Data Overload

More data equals better results

- Lots of text (, images, audio, ...) is good
- But how to process it all?
- Approximate algorithms!

Buy/rent a data center?

Make the best of what you've got

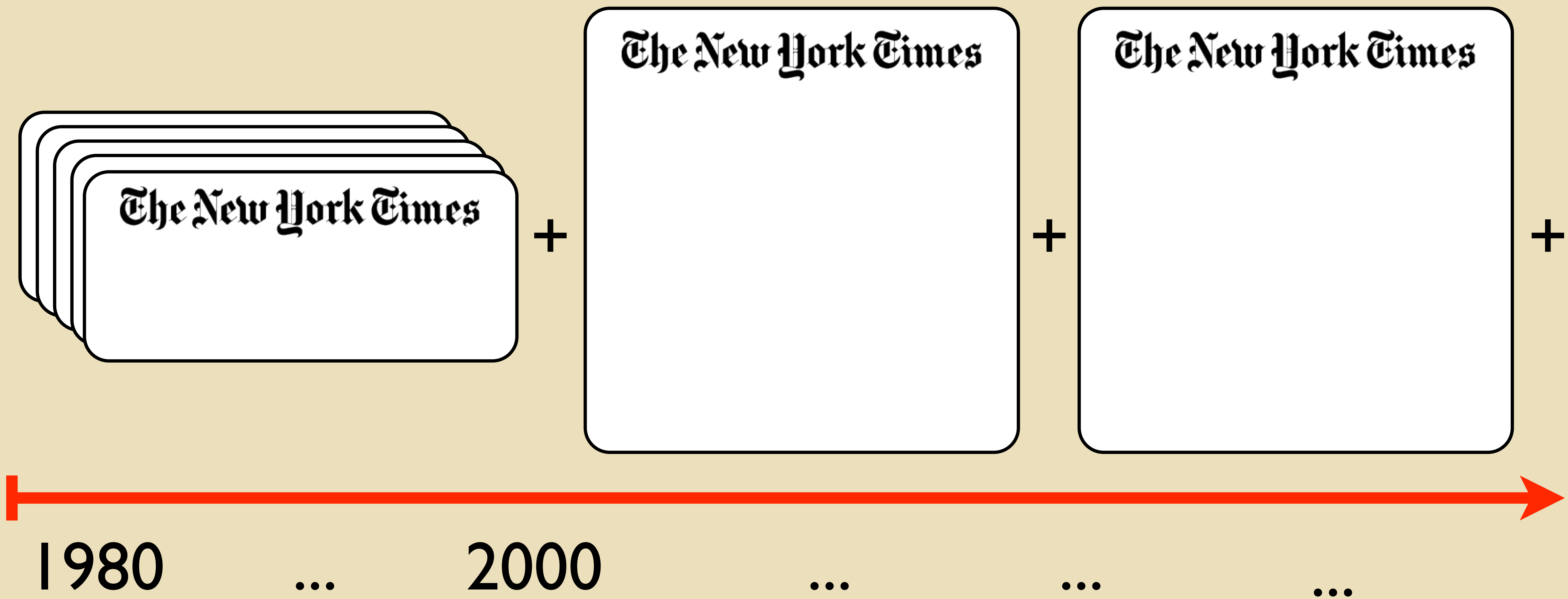
Bulky Data

1980 ... 1985 ... 1990 ... 1995 ... 2000 ... 2005

Bulky Data in Small Space

1980 ... 1985 ... 1990 ... 1995 ... 2000 ... 2005

Bulky Data in Small Space Online?



Outline

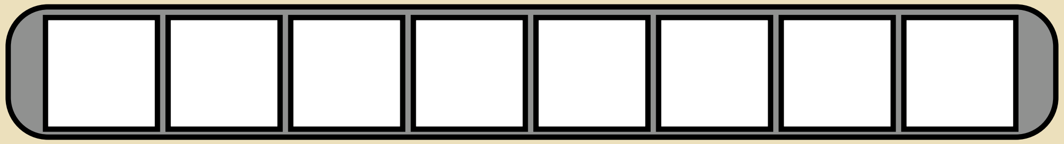
- Storing Static Counts
- Counting Online
- Experiments
- Additional Comments

Outline

- Storing Static Counts
- Counting Online
- Experiments
- Additional Comments

Bloom Filters [Bloom '70]

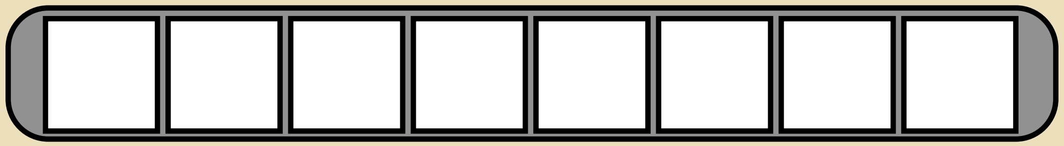
- Records set membership.
- No false negatives.
- Some false positives.
- Think hashtables, where you throw away the key.



Bloom Filters [Bloom '70]

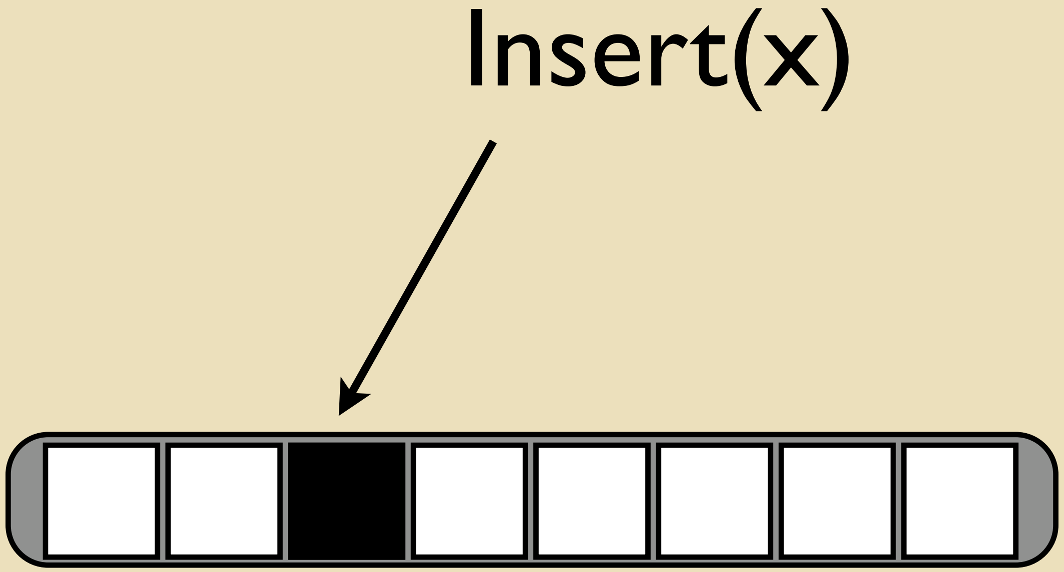
- Records set membership.
- No false negatives.
- Some false positives.
- Think hashtables, where you throw away the key.

Insert(x)



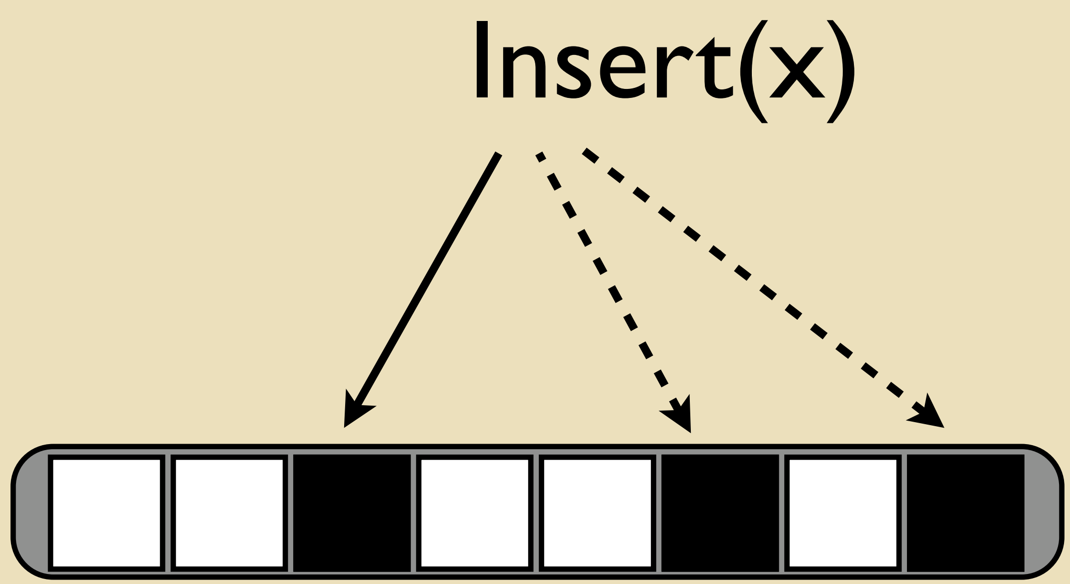
Bloom Filters [Bloom '70]

- Records set membership.
- No false negatives.
- Some false positives.
- Think hashtables, where you throw away the key.



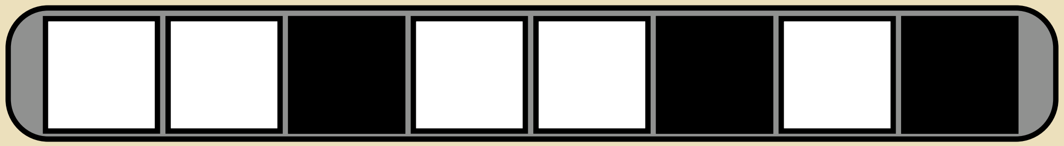
Bloom Filters [Bloom '70]

- Records set membership.
- No false negatives.
- Some false positives.
- Think hashtables, where you throw away the key.



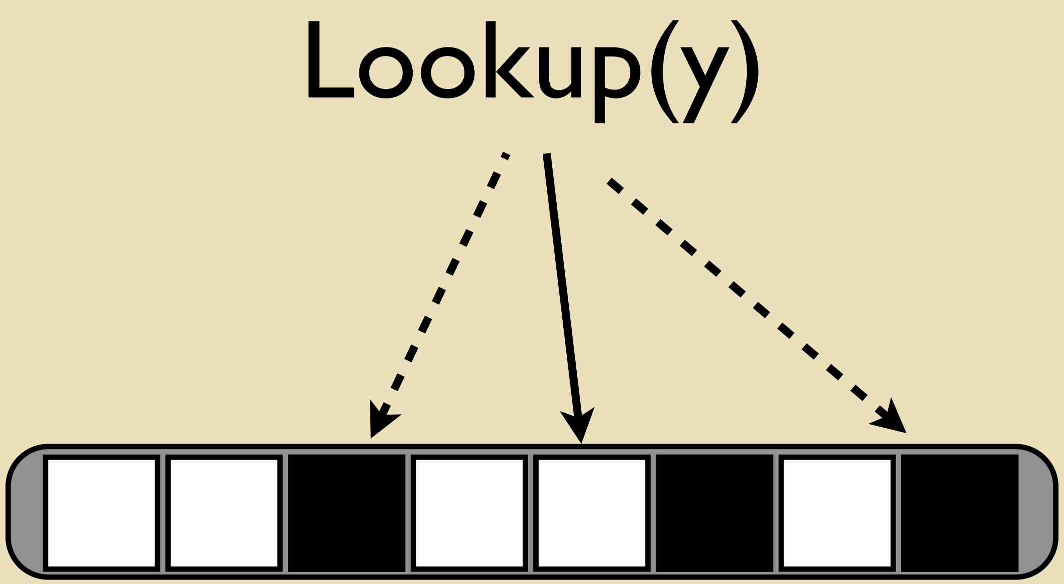
Bloom Filters [Bloom '70]

- Records set membership.
- No false negatives.
- Some false positives.
- Think hashtables, where you throw away the key.



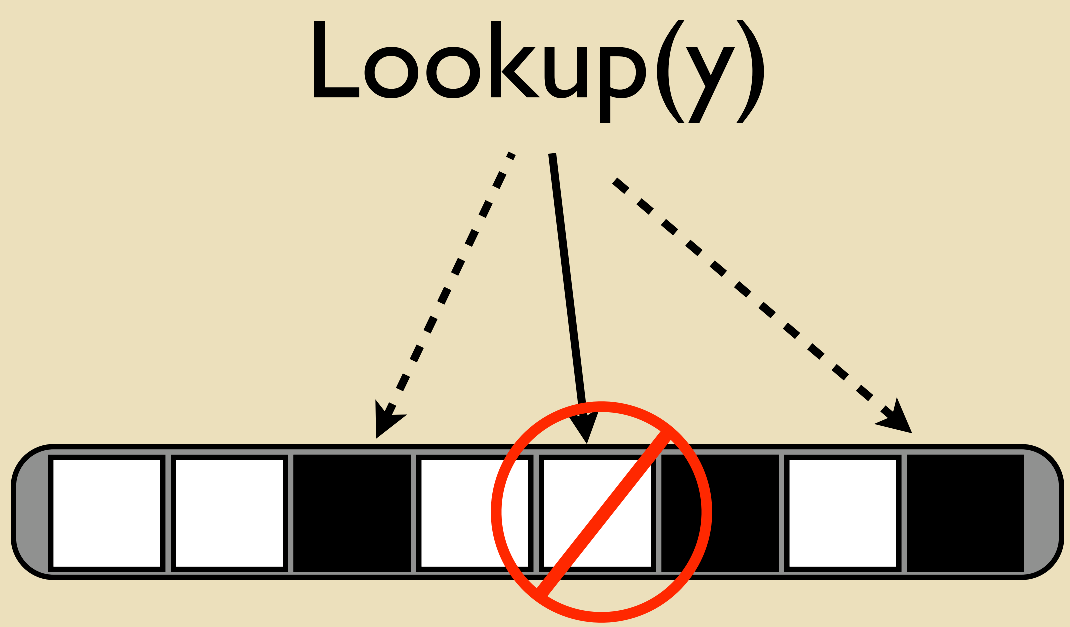
Bloom Filters [Bloom '70]

- Records set membership.
- No false negatives.
- Some false positives.
- Think hashtables, where you throw away the key.



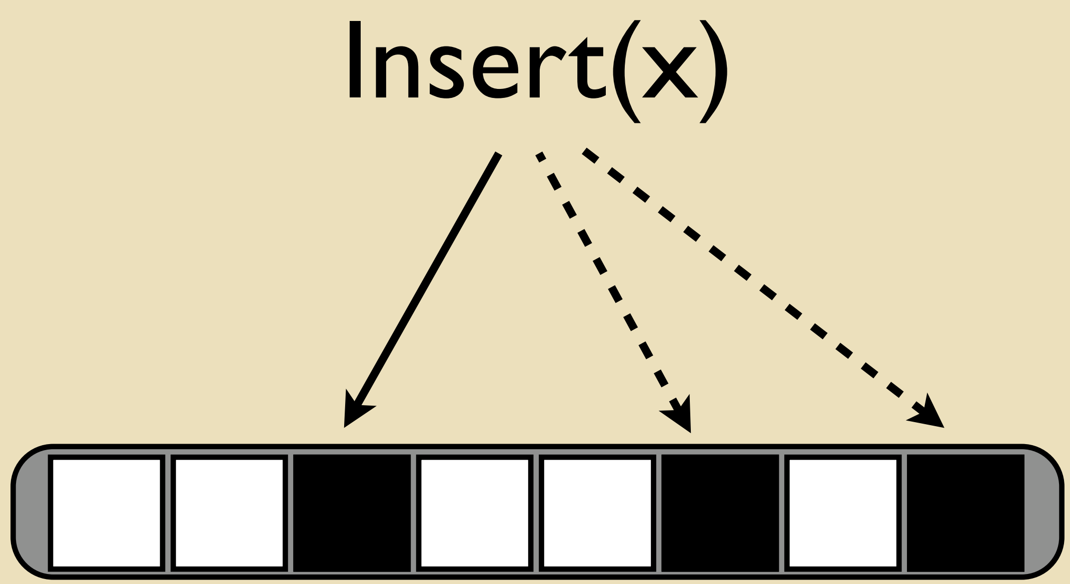
Bloom Filters [Bloom '70]

- Records set membership.
- No false negatives.
- Some false positives.
- Think hashtables, where you throw away the key.



Bloom Filters ...

- Bloom filters are nice when you can tolerate small false positives.
- And your x's are large.
- For example, Language Modeling.



Motivation: n~grams for MT

...
the dog
dog barked
barked at
...

Motivation: n-grams for MT

...

the dog 97

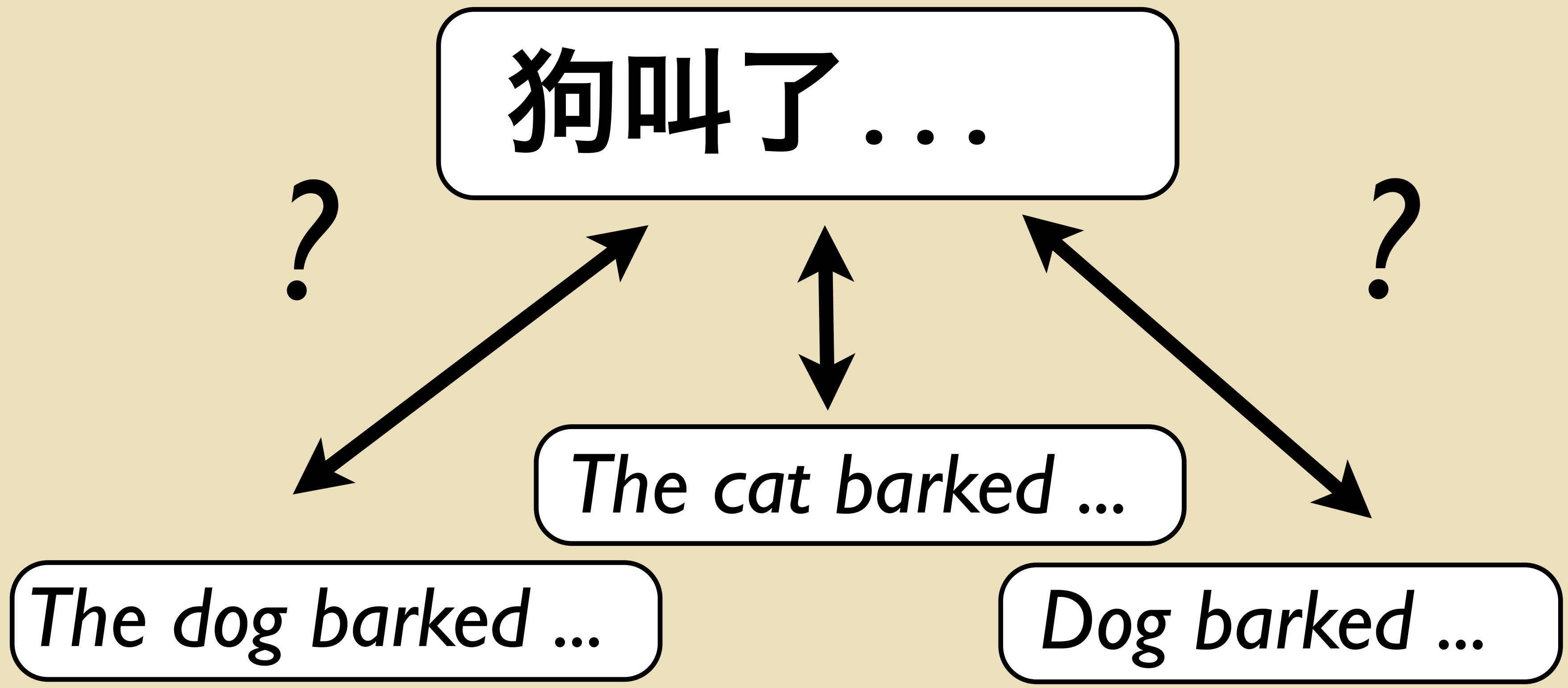
dog barked 42

barked at 58

...

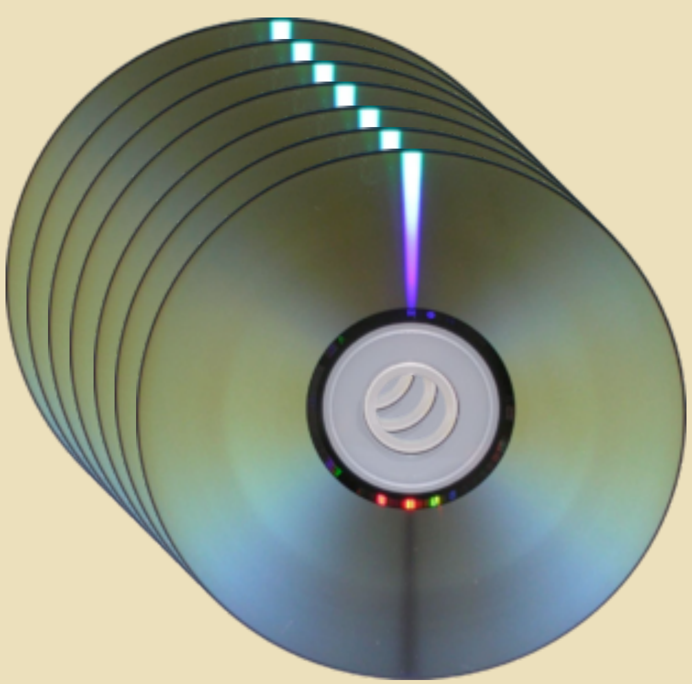
Motivation: n-grams for MT

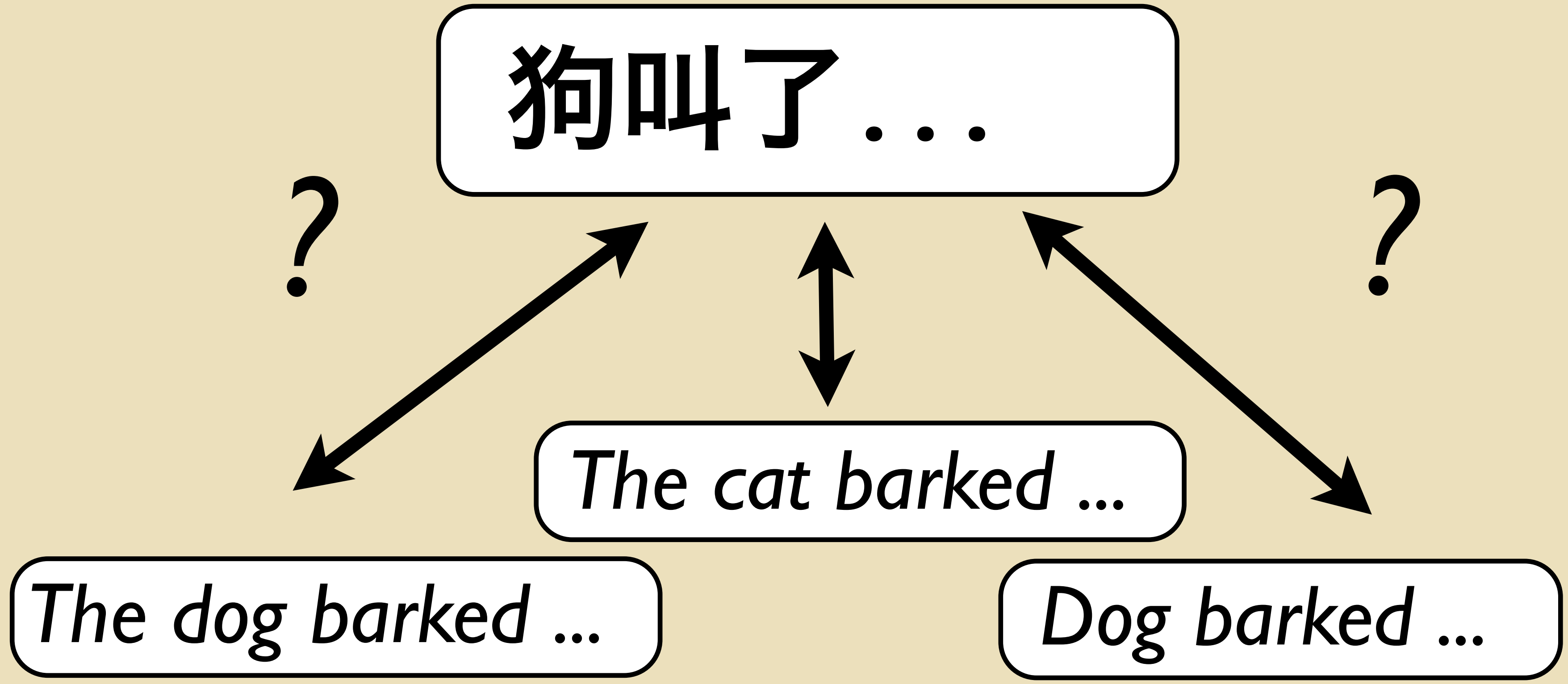
...
the dog 97
dog barked 42
barked at 58
...



Motivation: n-grams for MT

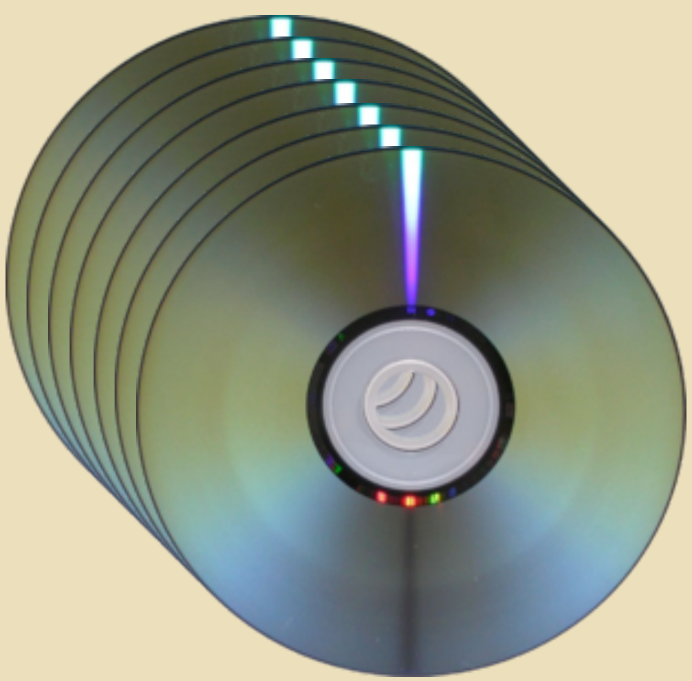
...
the dog 97
dog barked 42
barked at 58
...





Motivation: n-grams for MT

...
the dog 97
dog barked 42
barked at 58
...



狗叫了. . .

?

?

The cat barked ...

The dog barked ...

Dog barked ...

Storing Counts with Bloom Filters

Randomised Language Modelling for Statistical Machine Translation

David Talbot and Miles Osborne
School of Informatics, University of Edinburgh

We quantise raw frequencies, $c(x)$, using a logarithmic codebook as follows,

$$qc(x) = 1 + \lfloor \log_b c(x) \rfloor.$$

ACL 2007

Storing Counts

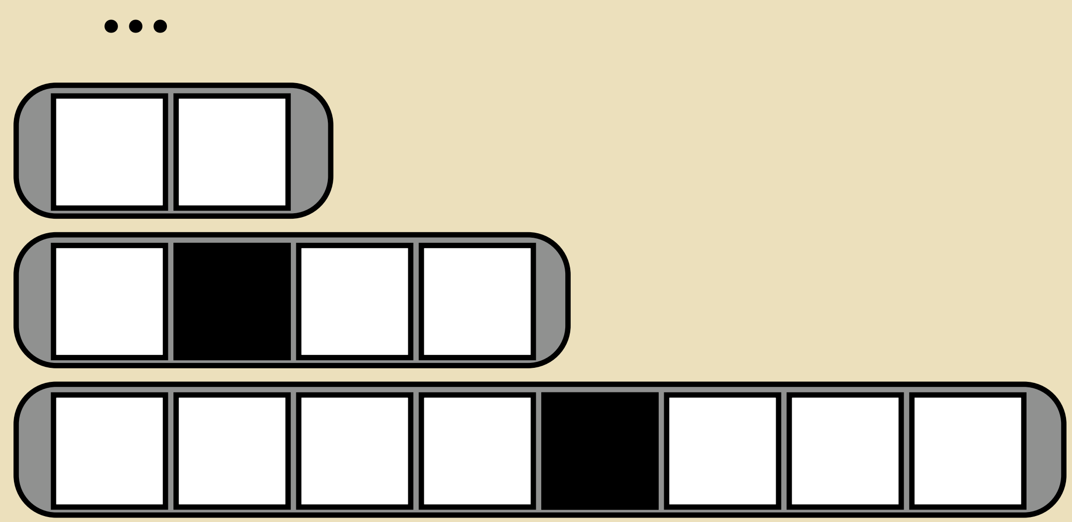
- Multiple layers of Bloom filters.
- Store exponent, in unary.

...

$$qc(x) = 3$$

$$qc(x) = 2$$

$$qc(x) = 1$$



$$c(x) \approx b^{qc(x)-1}$$

Outline

- Storing Static Counts
- Counting Online
- Experiments
- Additional Comments

Spectral Bloom Filter

Spectral Bloom Filters

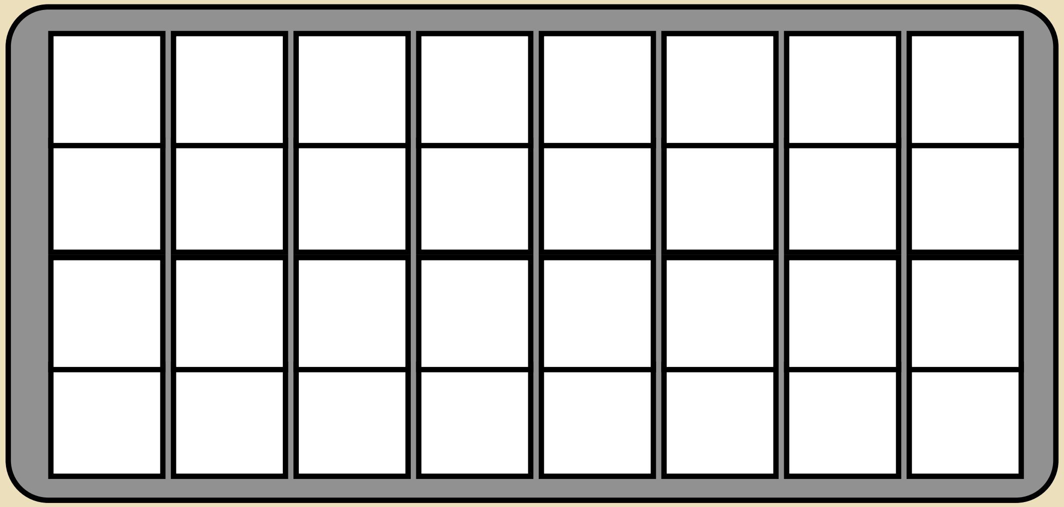
Saar Cohen
School of Computer Science
Tel Aviv University

Yossi Matias*
School of Computer Science
Tel Aviv University

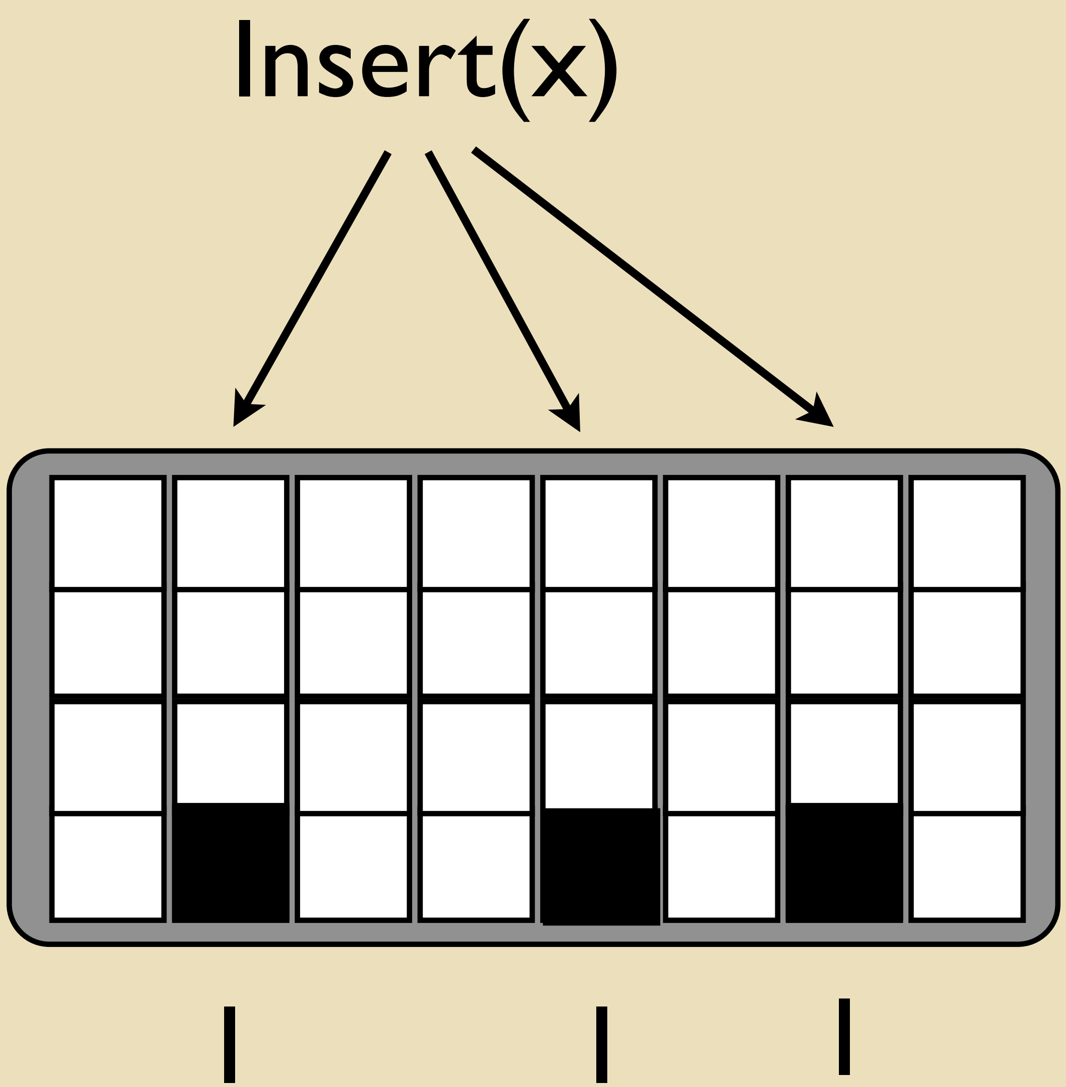
The Spectral Bloom Filter (SBF) replaces the bit vector V with a vector of m counters, C.

SIGMOD 2003

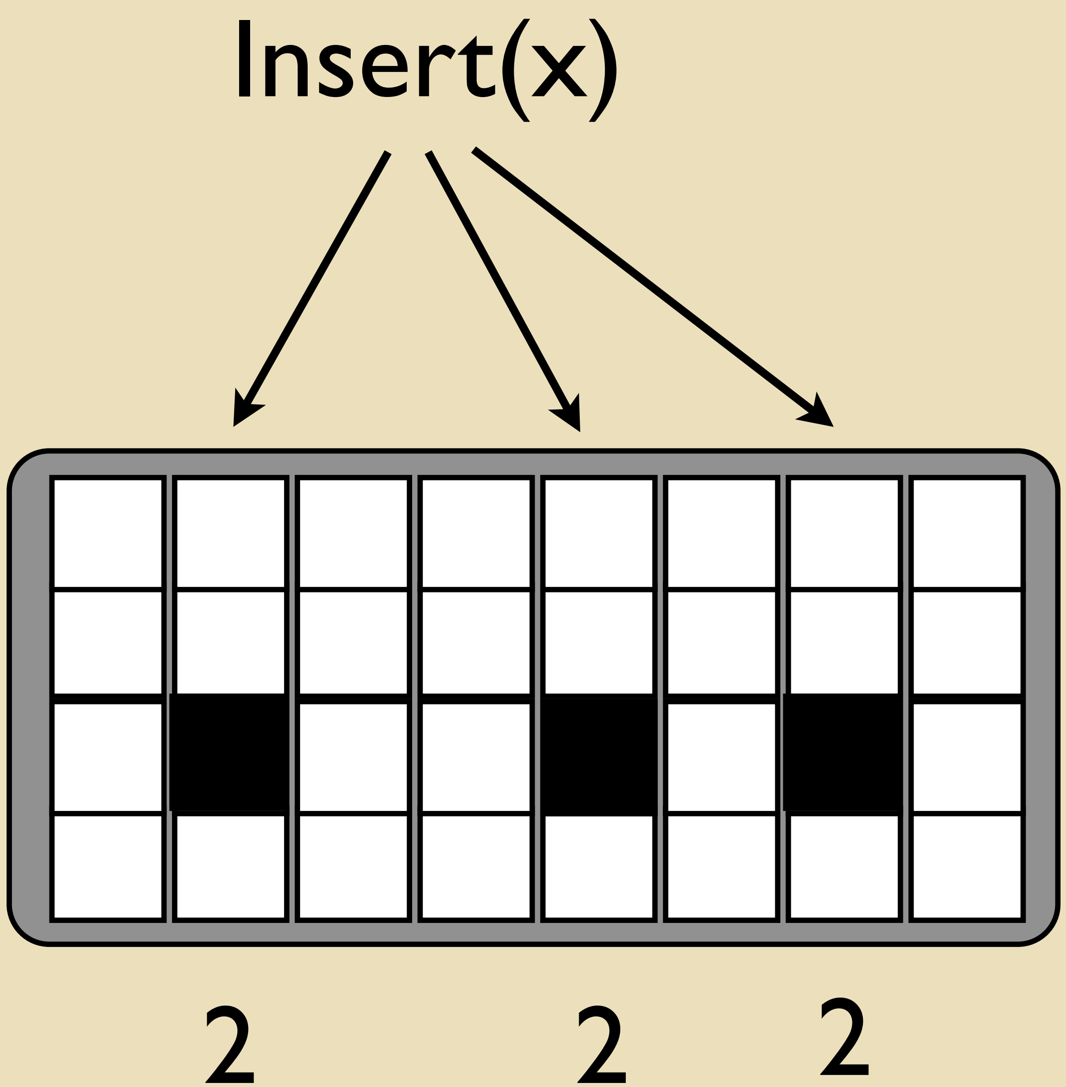
Spectral Bloom Filter [Cohen & Matias '03]



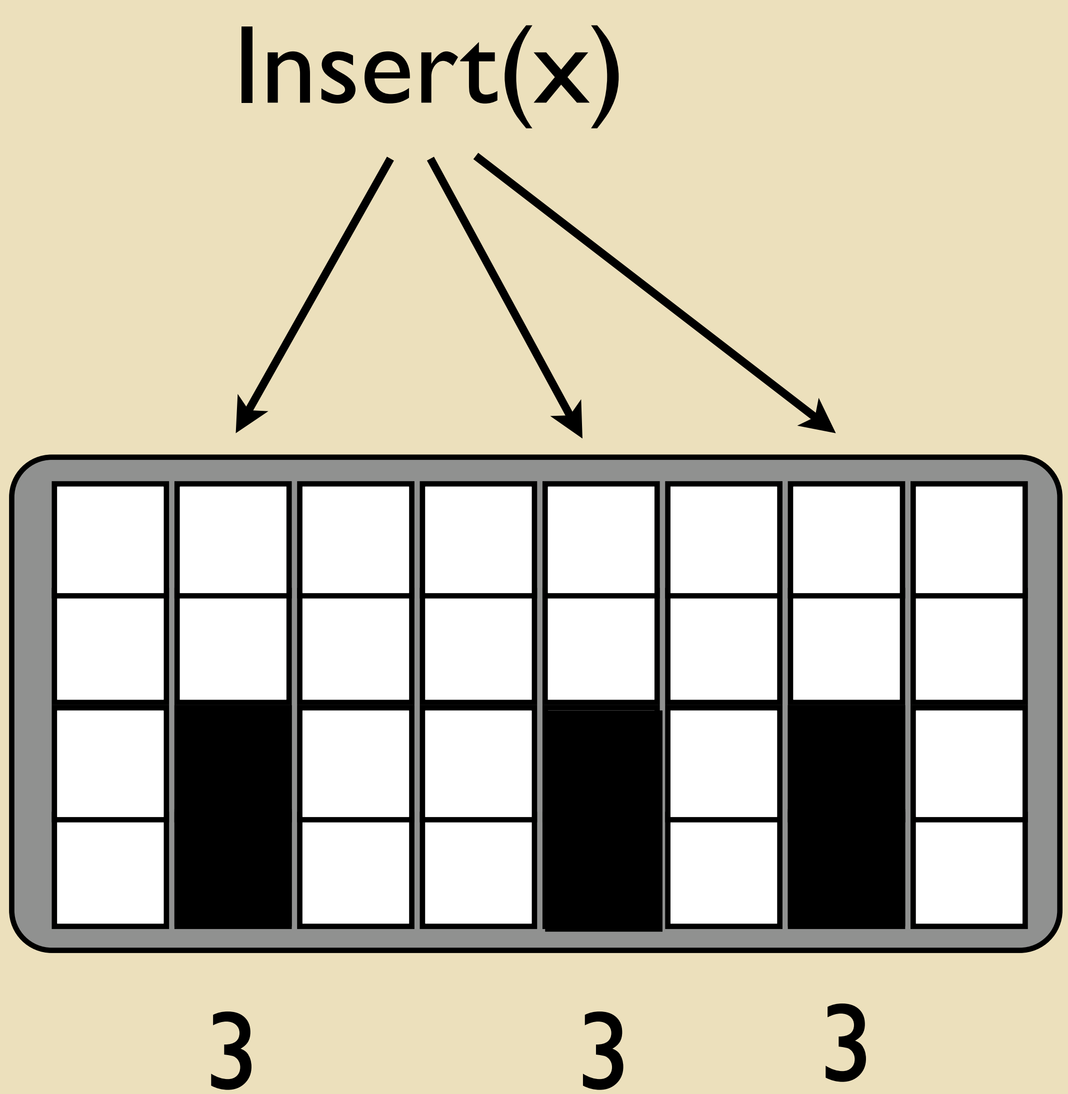
Spectral Bloom Filter [Cohen & Matias '03]



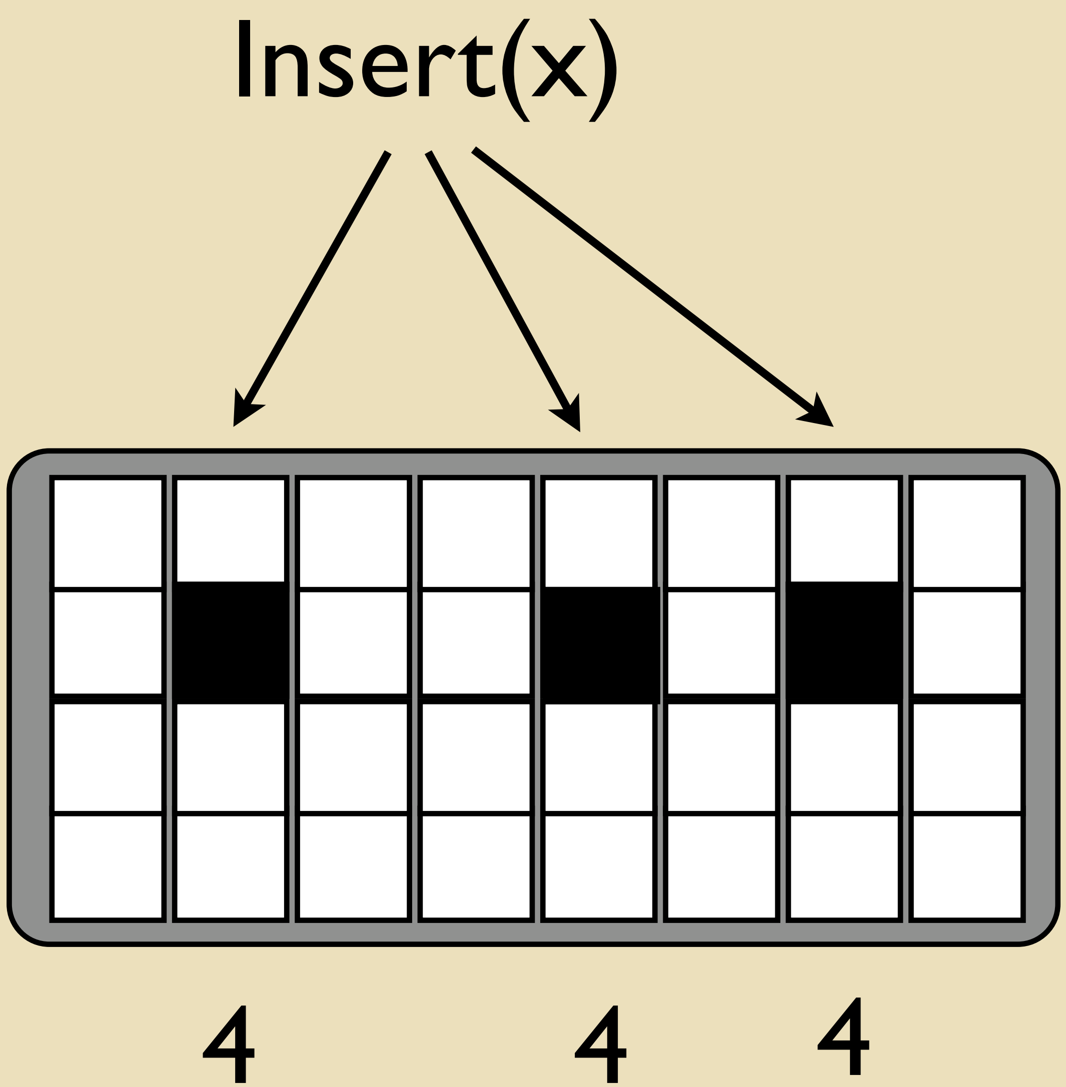
Spectral Bloom Filter [Cohen & Matias '03]



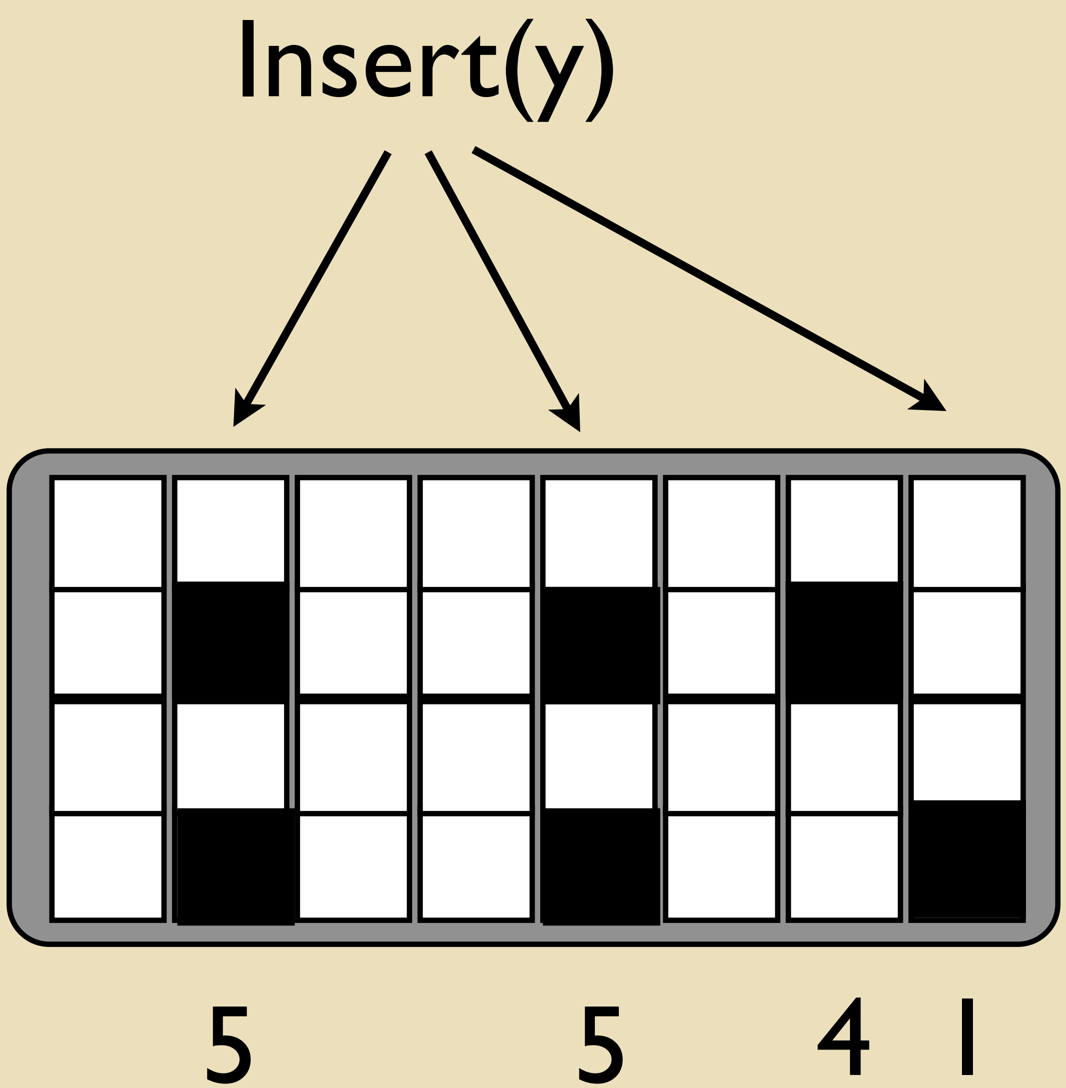
Spectral Bloom Filter [Cohen & Matias '03]



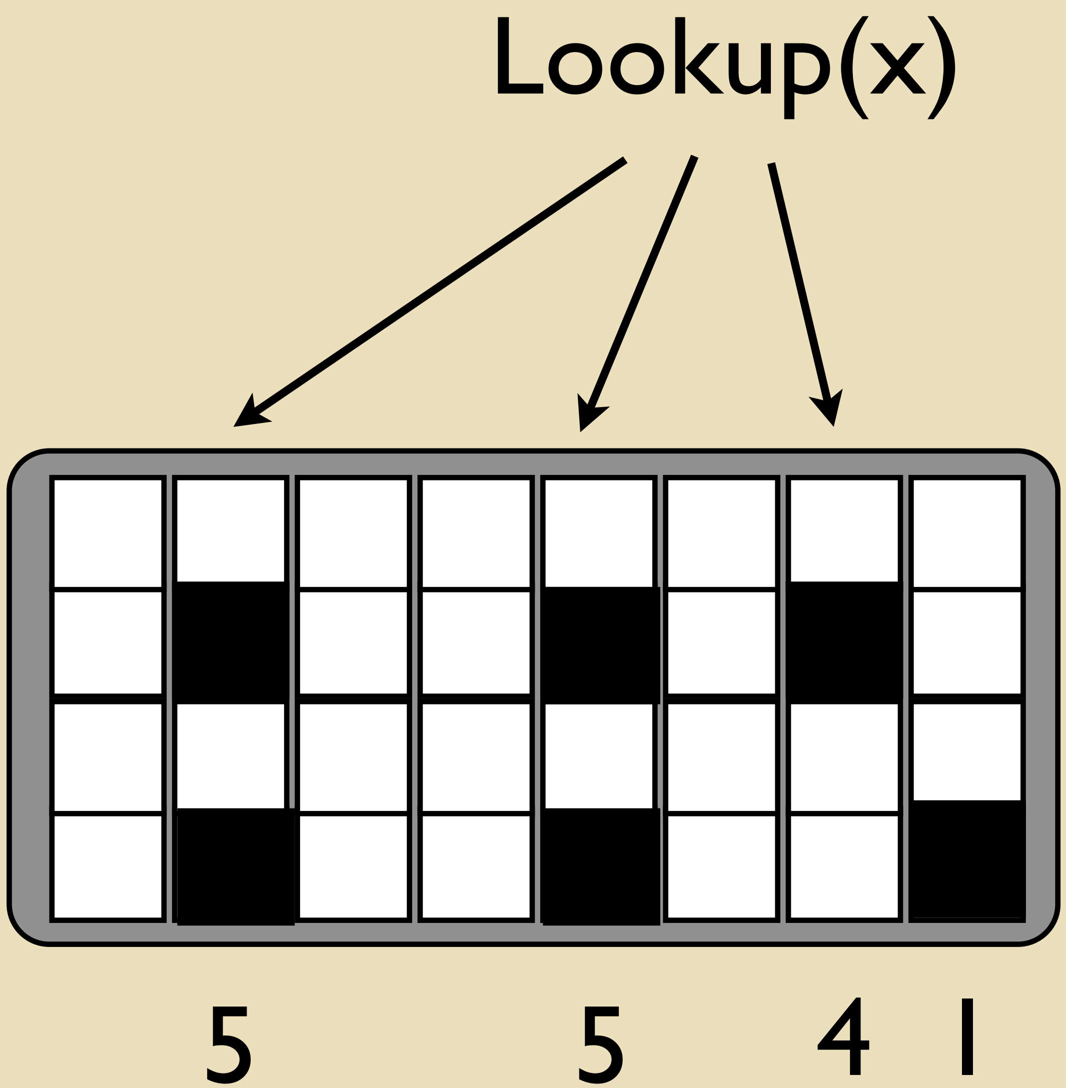
Spectral Bloom Filter [Cohen & Matias '03]



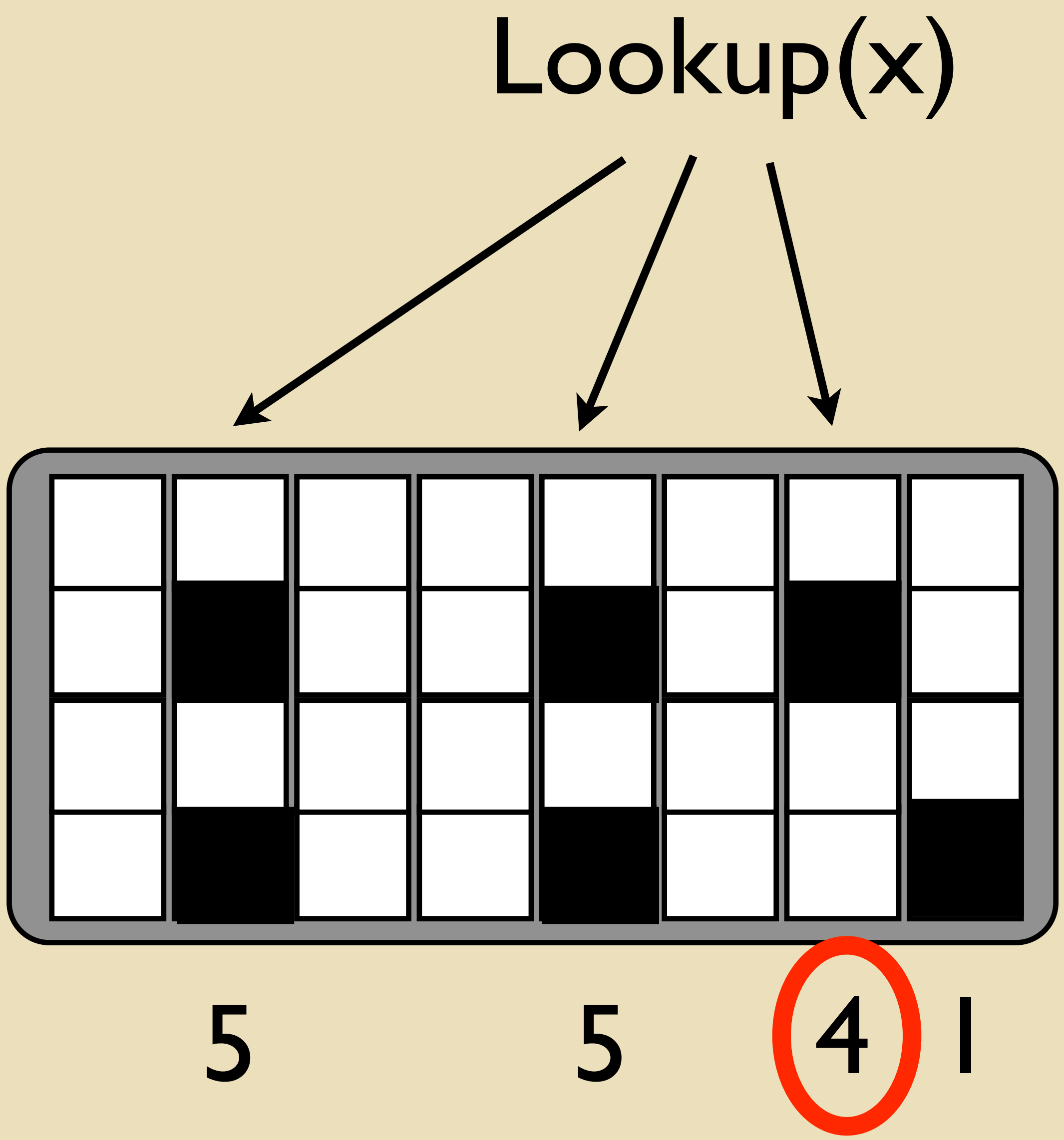
Spectral Bloom Filter [Cohen & Matias '03]



Spectral Bloom Filter [Cohen & Matias '03]



Spectral Bloom Filter [Cohen & Matias '03]

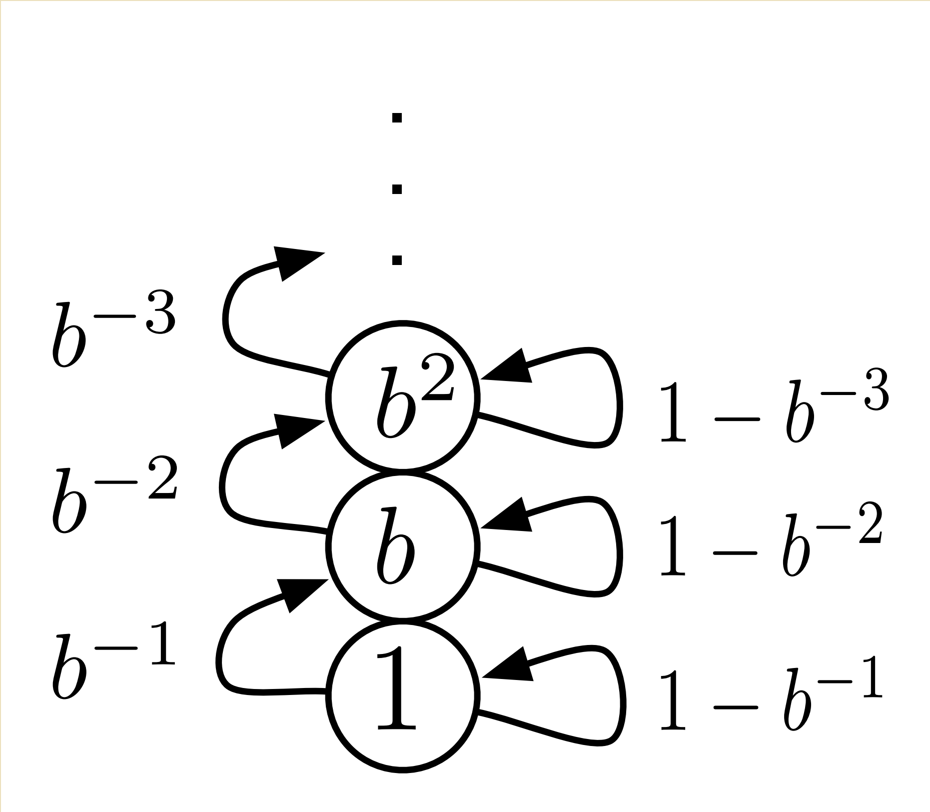


Collect Counts Online

- Count in log-scale, to save space.

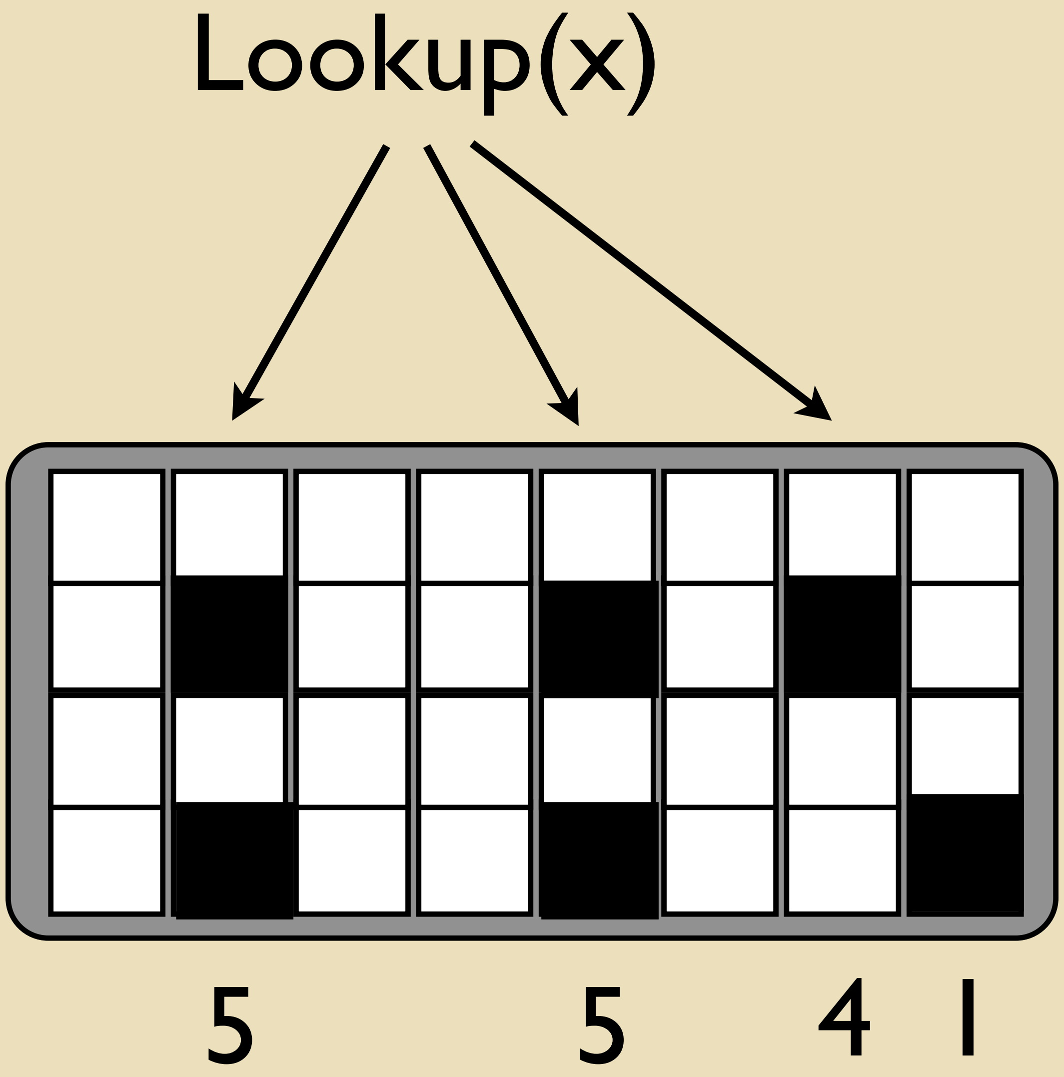
Collect Counts Online

- Count in log-scale, to save space.
- Robert Morris (1978) gave us a way to do this.



Morris Bloom Counter

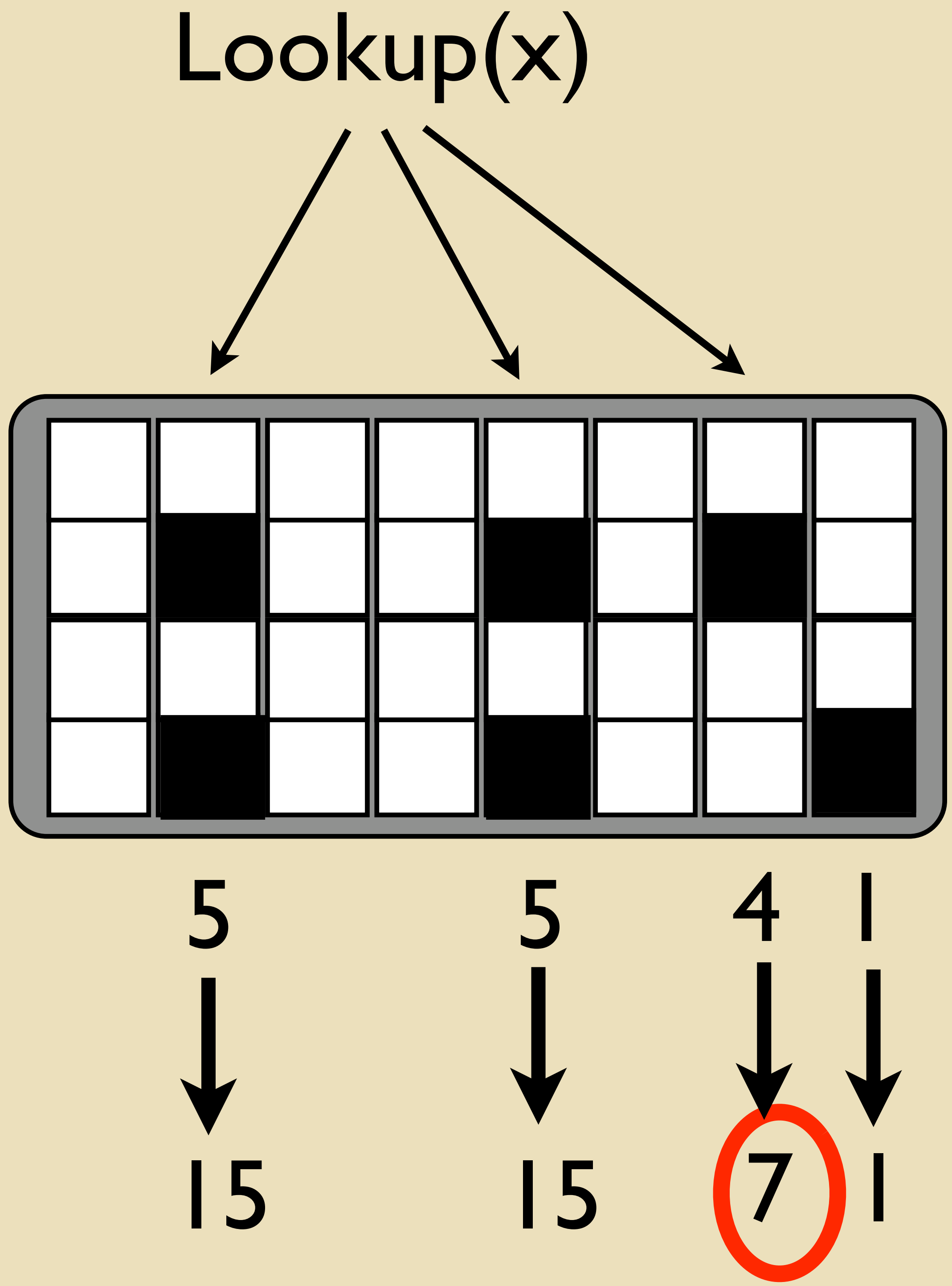
- Spectral Bloom Filter,
- but with Morris style updating.



Morris Bloom Counter

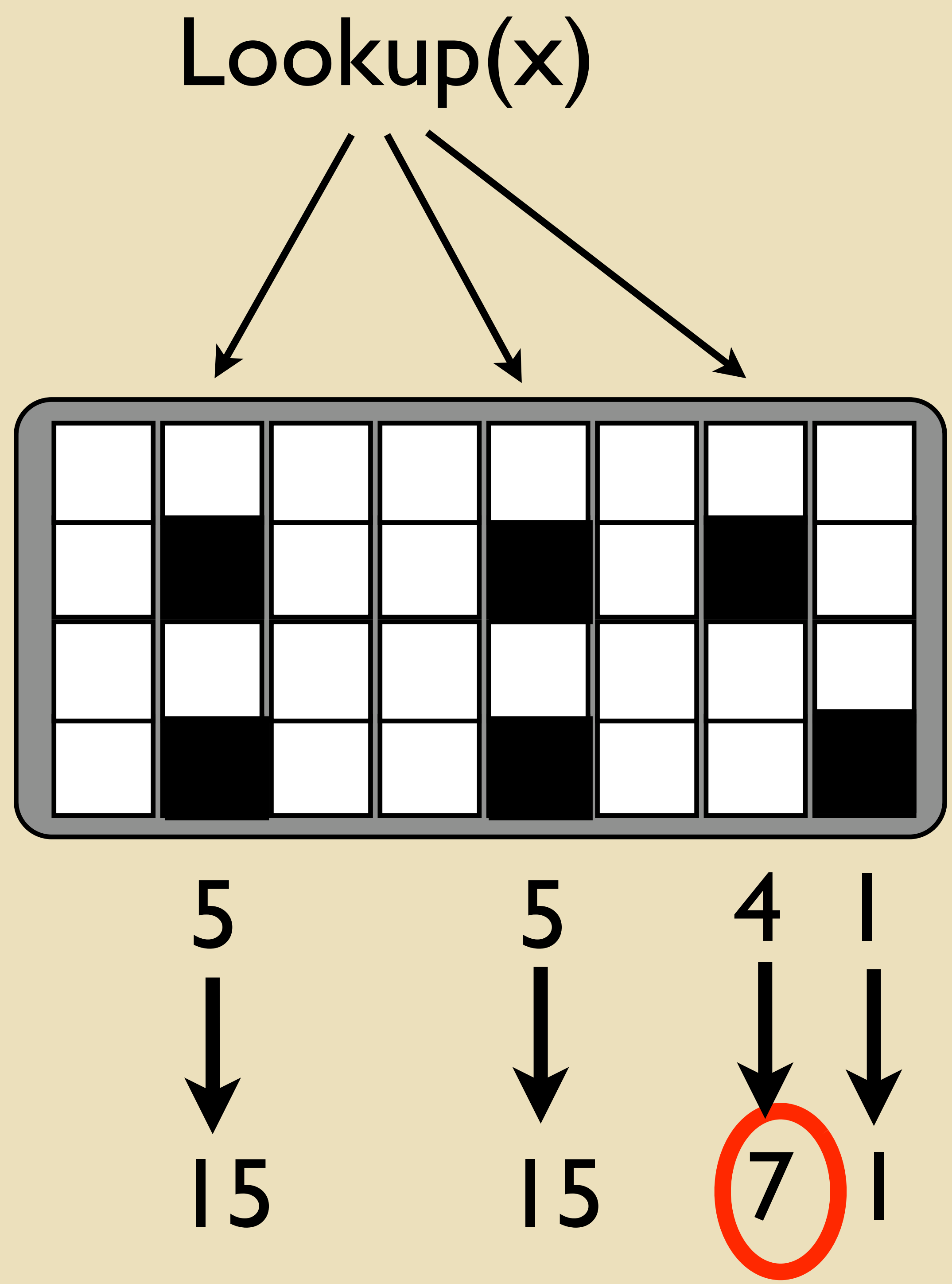
- Spectral Bloom Filter,
- but with Morris style updating.

$$c(x) \approx \frac{b^f - 1}{b - 1}$$



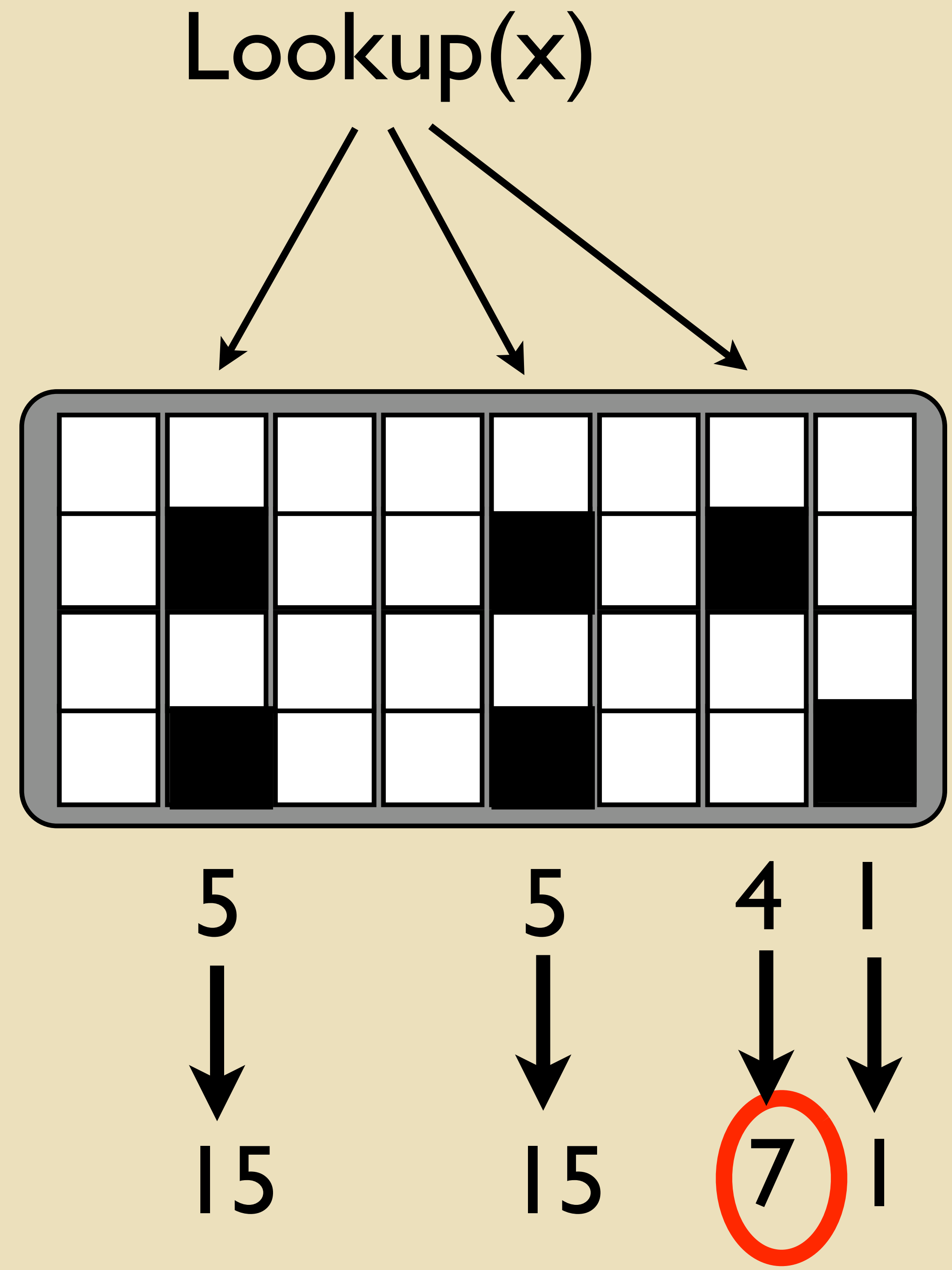
Morris Bloom Counter

- Same amount of space as Spectral Bloom Filter,



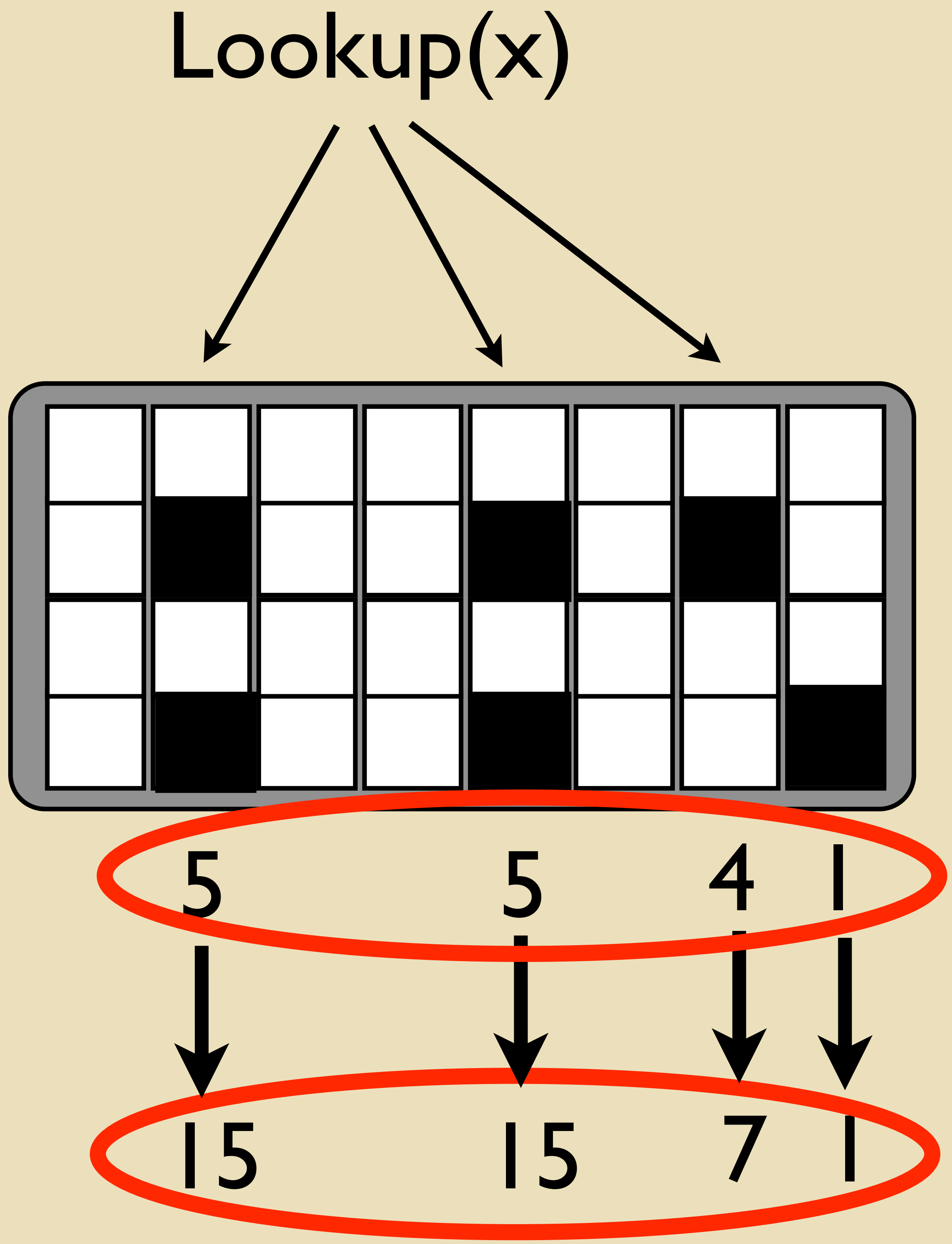
Morris Bloom Counter

- Same amount of space as Spectral Bloom Filter,
- gives exponentially larger max-count,



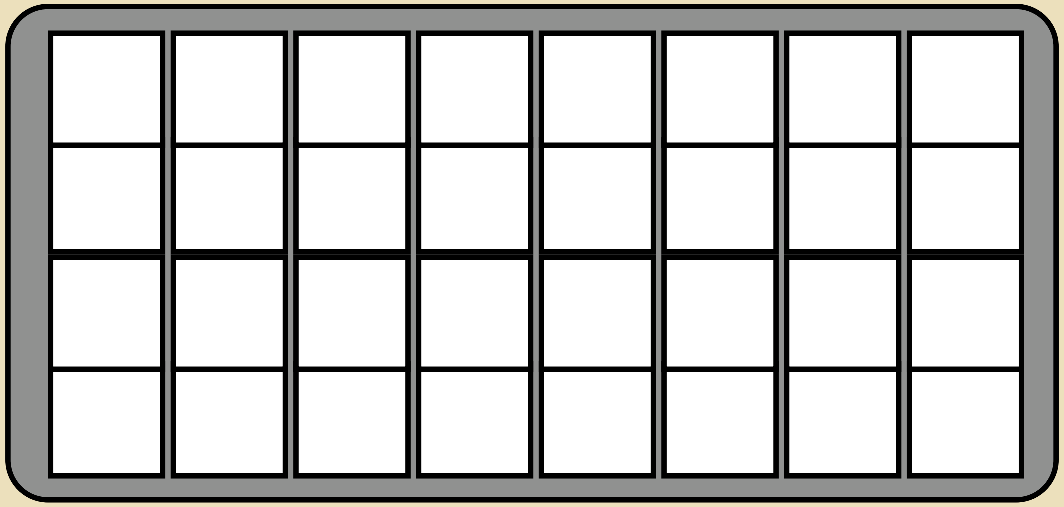
Morris Bloom Counter

- Same amount of space as Spectral Bloom Filter,
- gives exponentially larger max-count,
- but false positives can therefore have higher relative error.



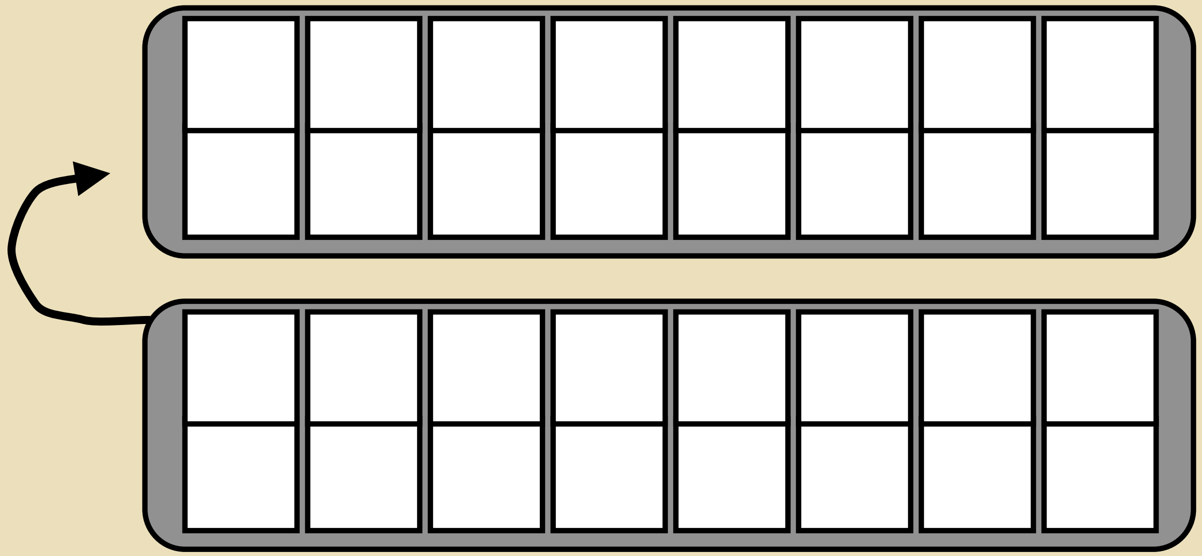
Reduce False Positive Rate

- Morris Bloom Counter,



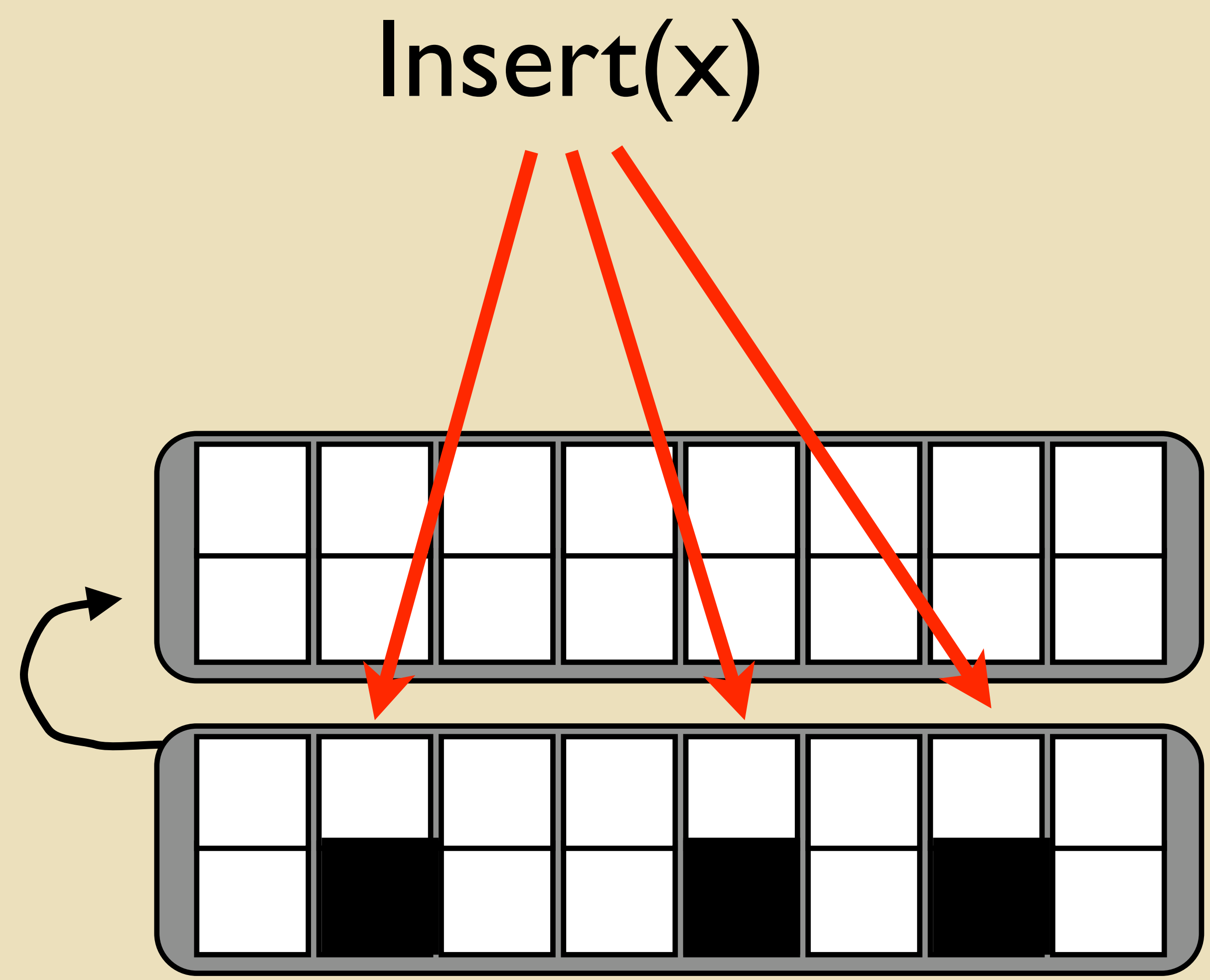
Reduce False Positive Rate

- Morris Bloom Counter,
- split into layers,



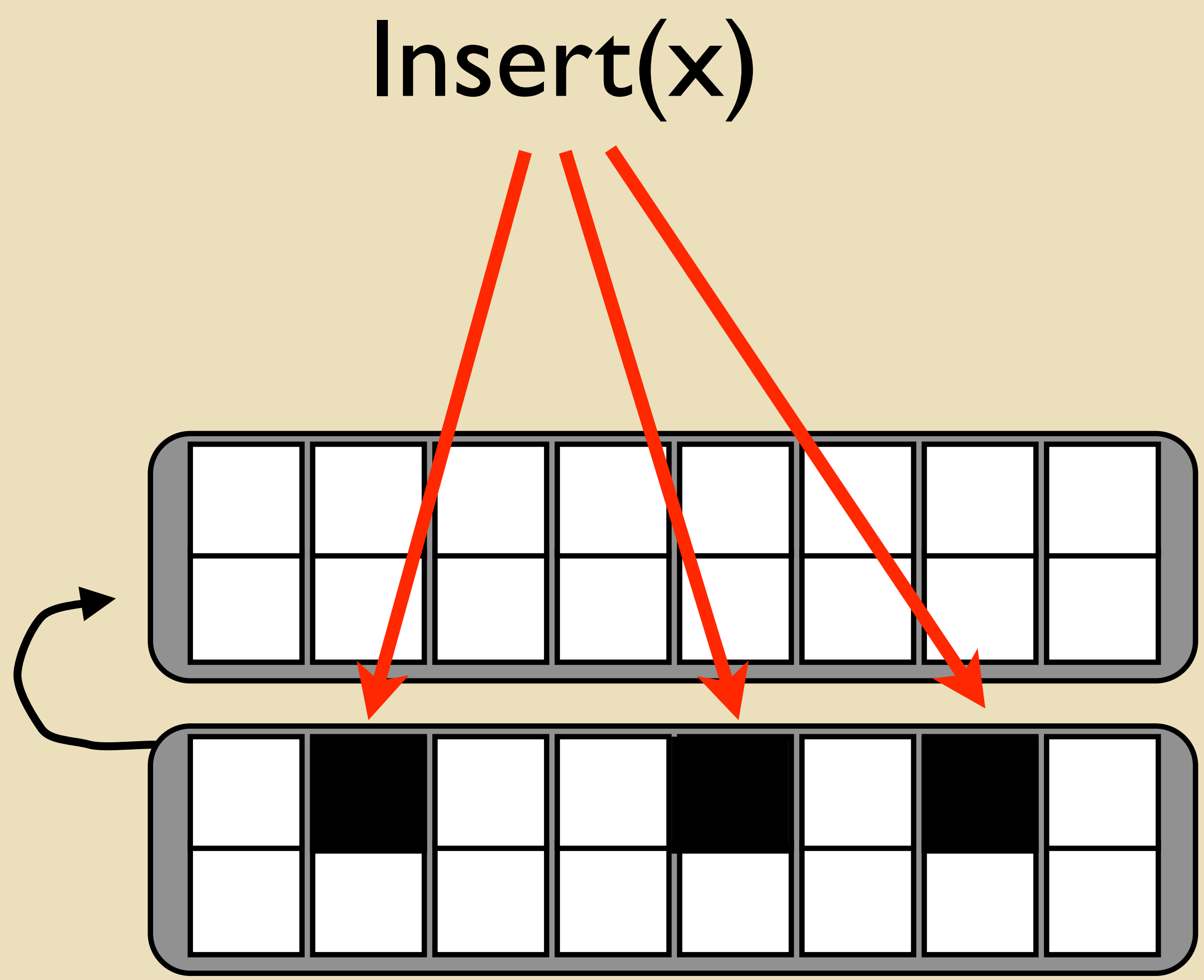
Reduce False Positive Rate

- Morris Bloom Counter,
- split into layers,
- with different hash functions per layer.



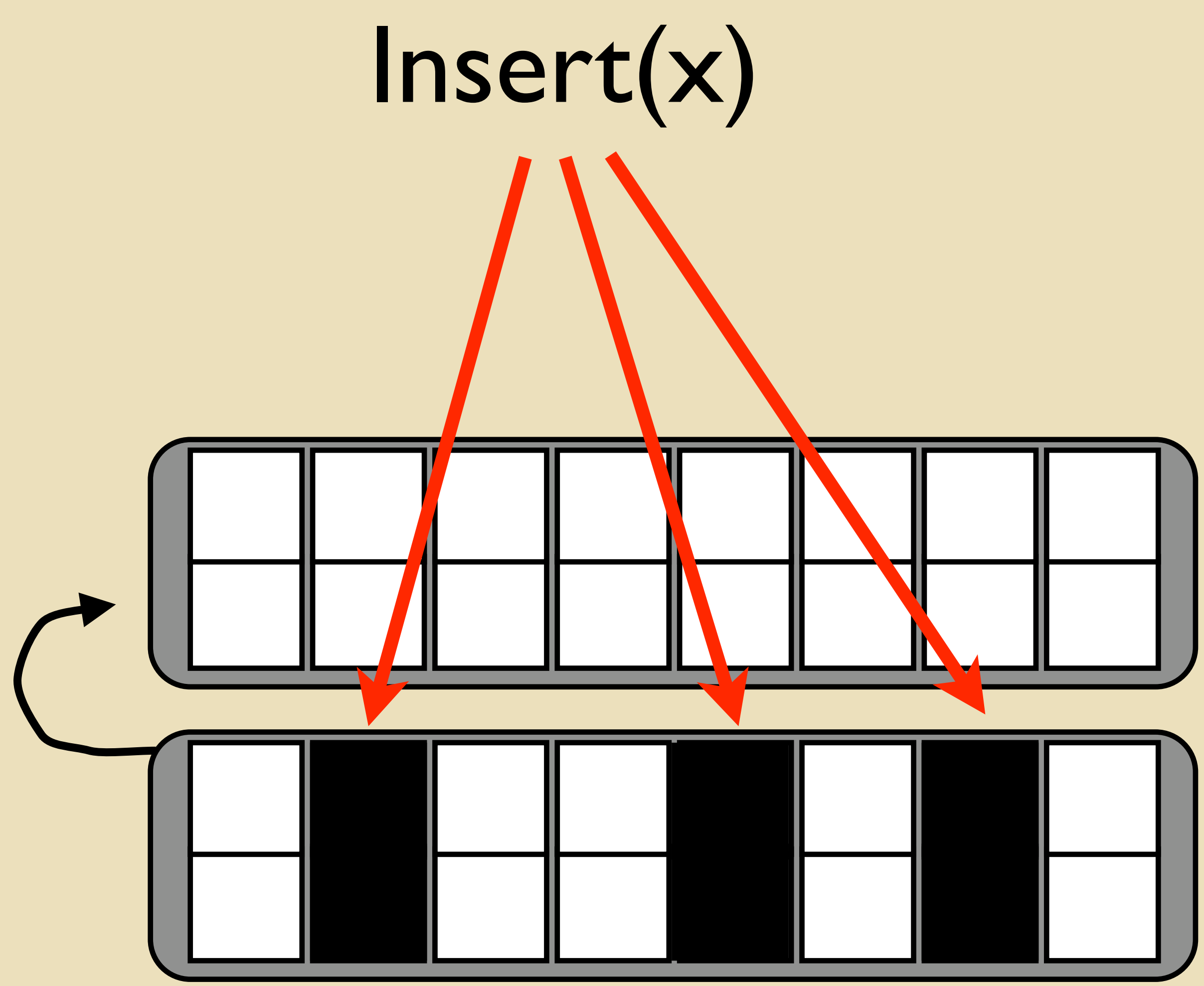
Reduce False Positive Rate

- Morris Bloom Counter,
- split into layers,
- with different hash functions per layer.



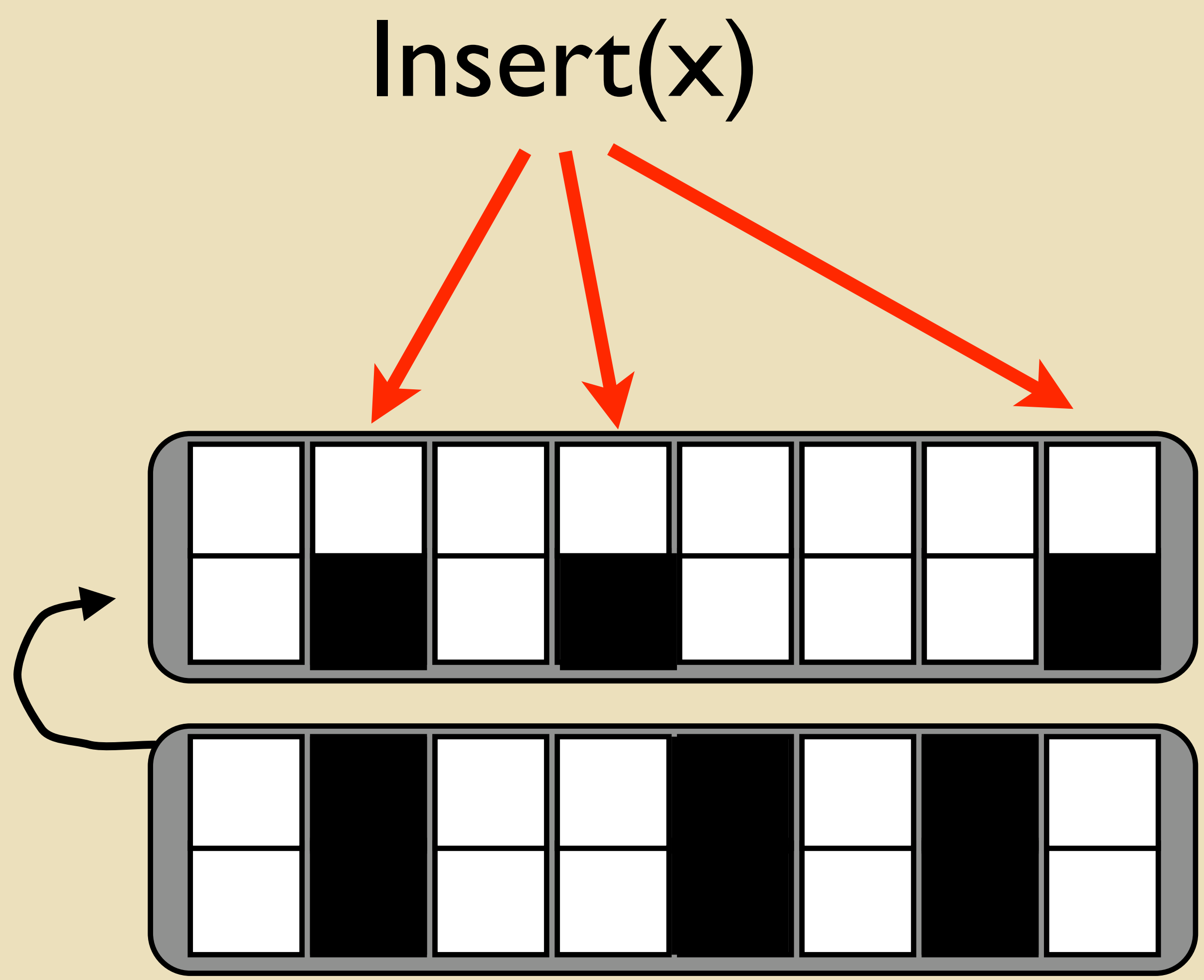
Reduce False Positive Rate

- Morris Bloom Counter,
- split into layers,
- with different hash functions per layer.



Reduce False Positive Rate

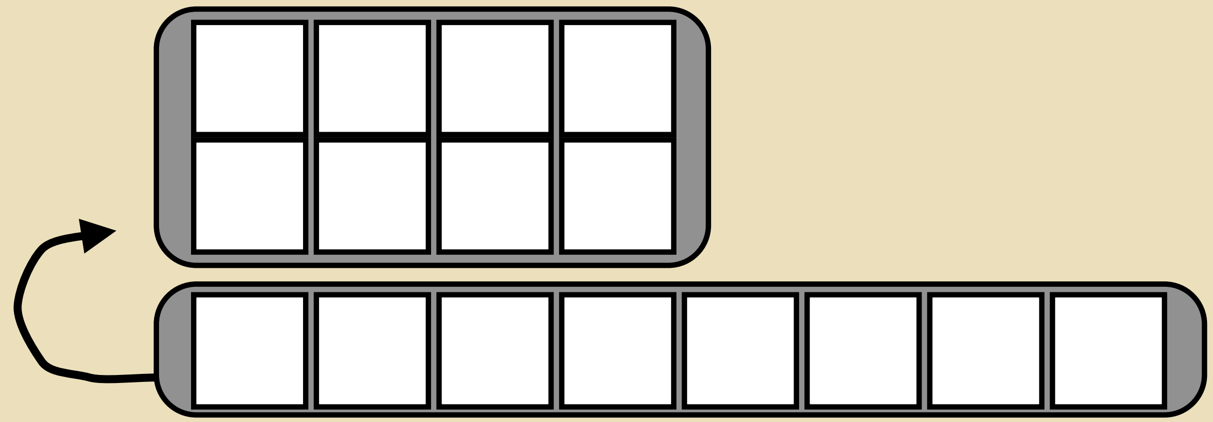
- Morris Bloom Counter,
- split into layers,
- with different hash functions per layer.



Talbot Osborne Morris Bloom (TOMB) Counter

- Combination of Morris Bloom Counter with Talbot Osborne count storage.

- Stay tuned for related work by Talbot.



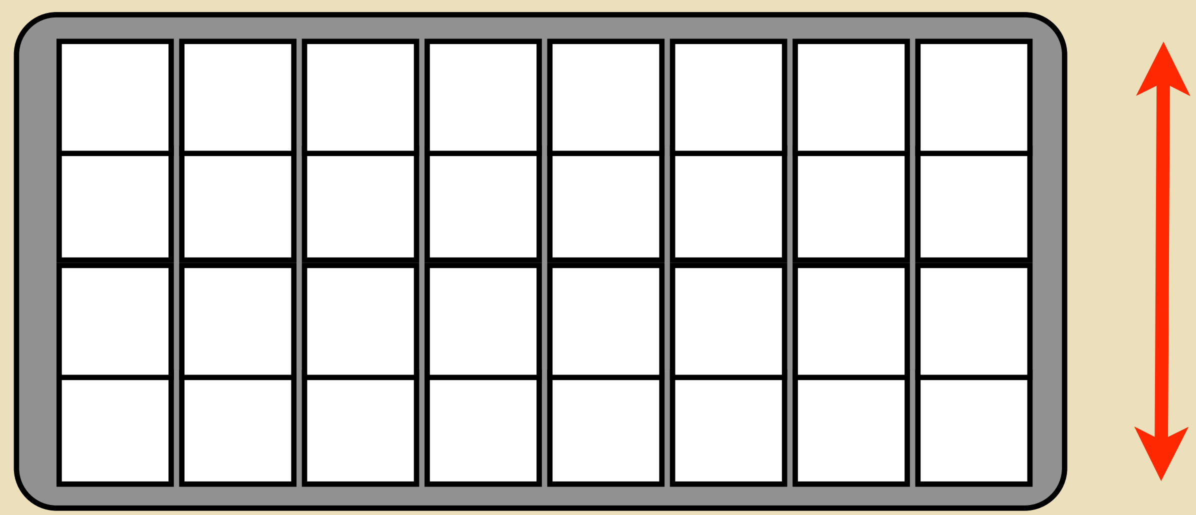
Tradeoff

- Trade number of layers for expressivity.

$$M = \sum_i 2^{h_i} - 1$$

Tradeoff

- Trade number of layers for expressivity.



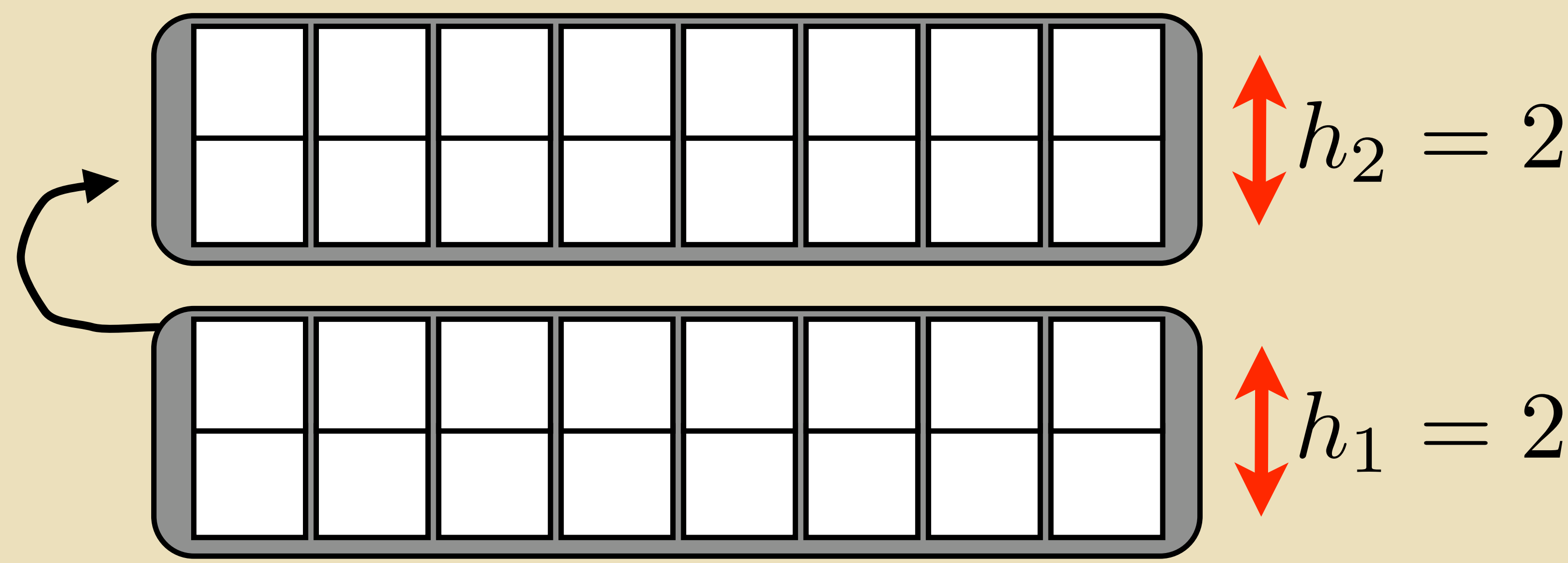
$h = 4$

$$0, 1, 2, \ldots, 14, 15$$

$$M = \sum_i 2^{h_i} - 1$$

Tradeoff

- Trade number of layers for expressivity.

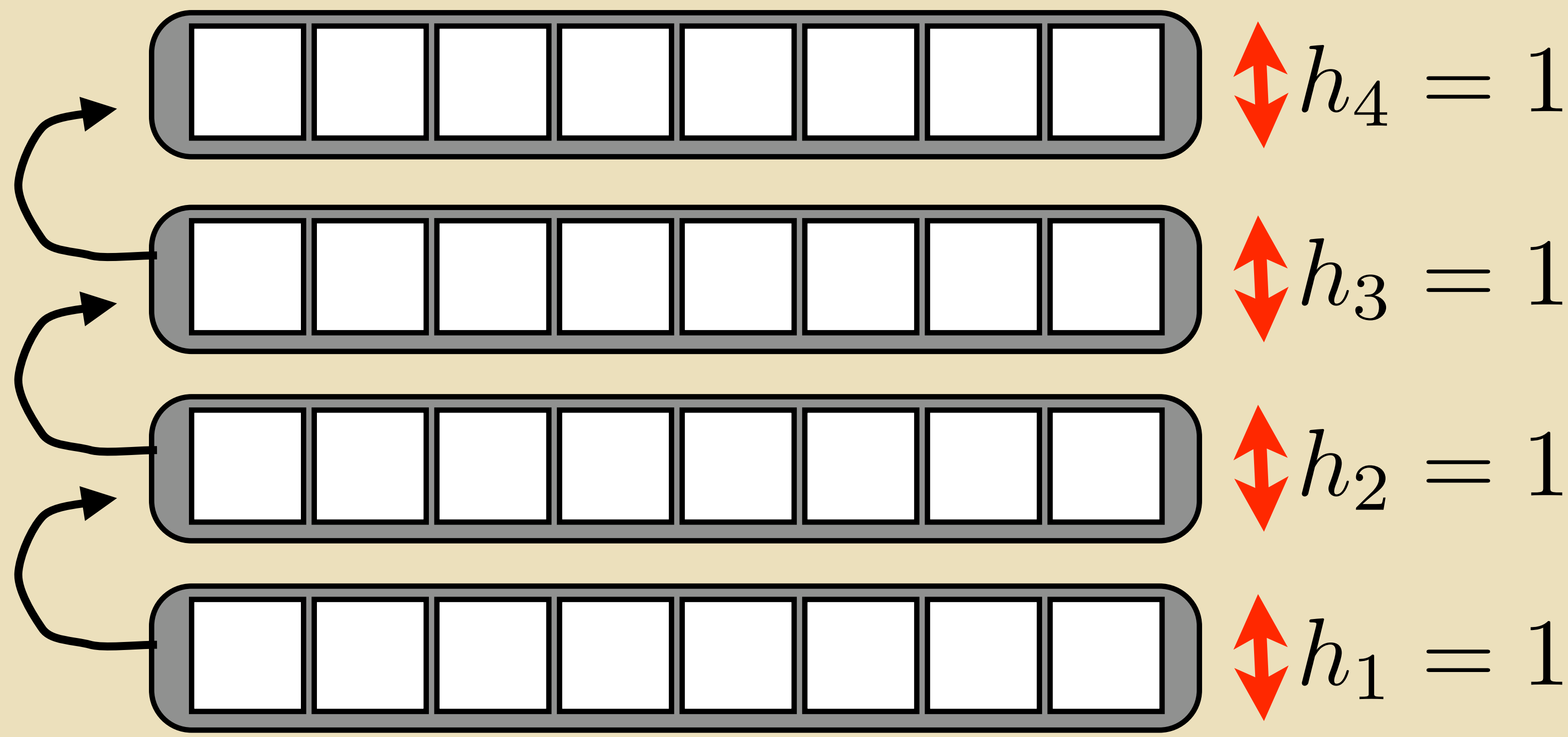


$$0, 1, 2, 3, 4, 5, 6$$

$$M = \sum_i 2^{h_i} - 1$$

Tradeoff

- Trade number of layers for expressivity.

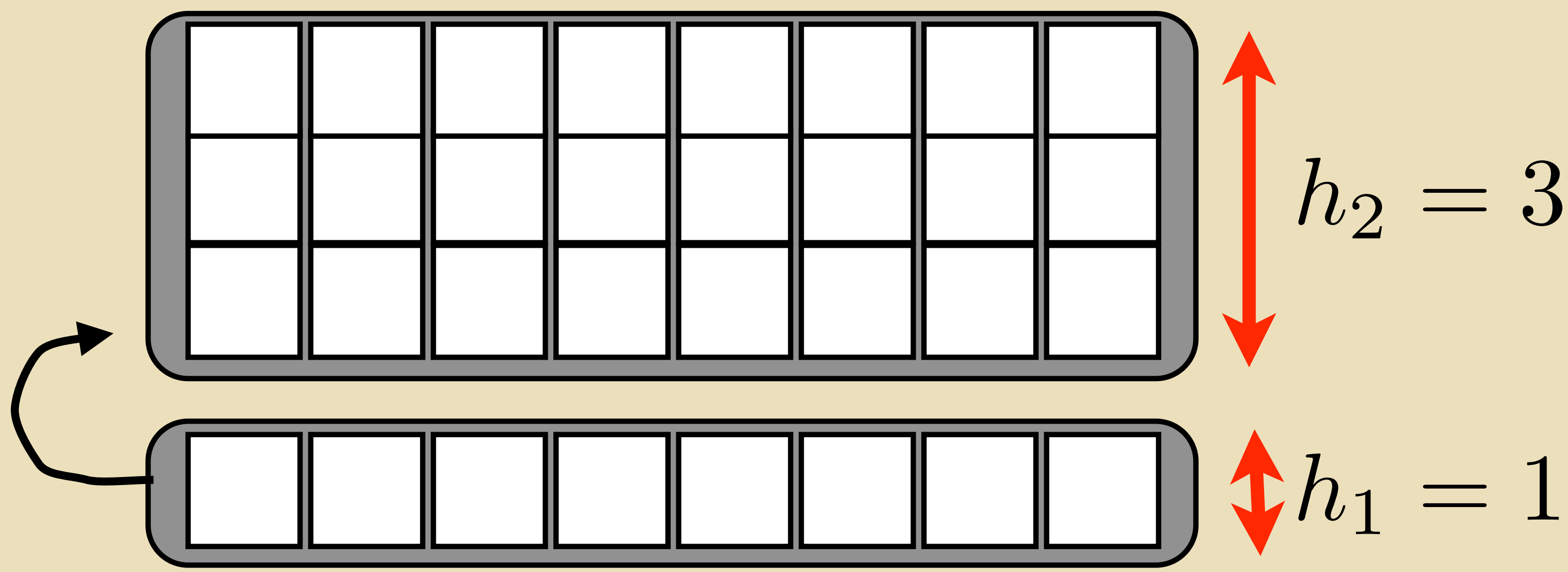


$$0, 1, 2, 3, 4$$

$$M = \sum_i 2^{h_i} - 1$$

Tradeoff

- Trade number of layers for expressivity.

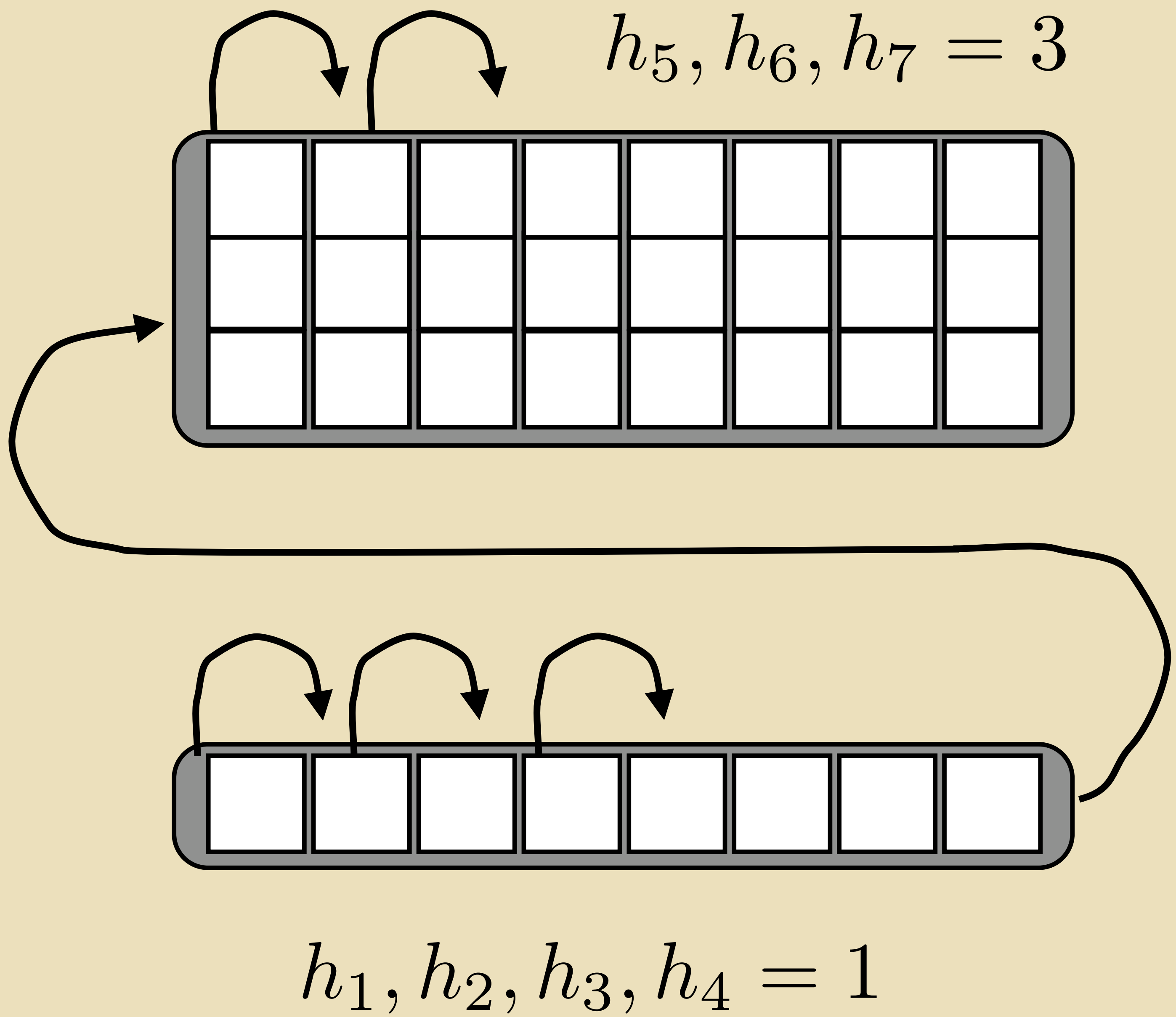


$$1, 2, 3, \ldots, 7, 8$$

$$M = \sum_i 2^{h_i} - 1$$

"Layers"

- Layers are a useful visualization.
- In practice, consecutive layers of equal height are implemented as single vectors with sets of hash functions.

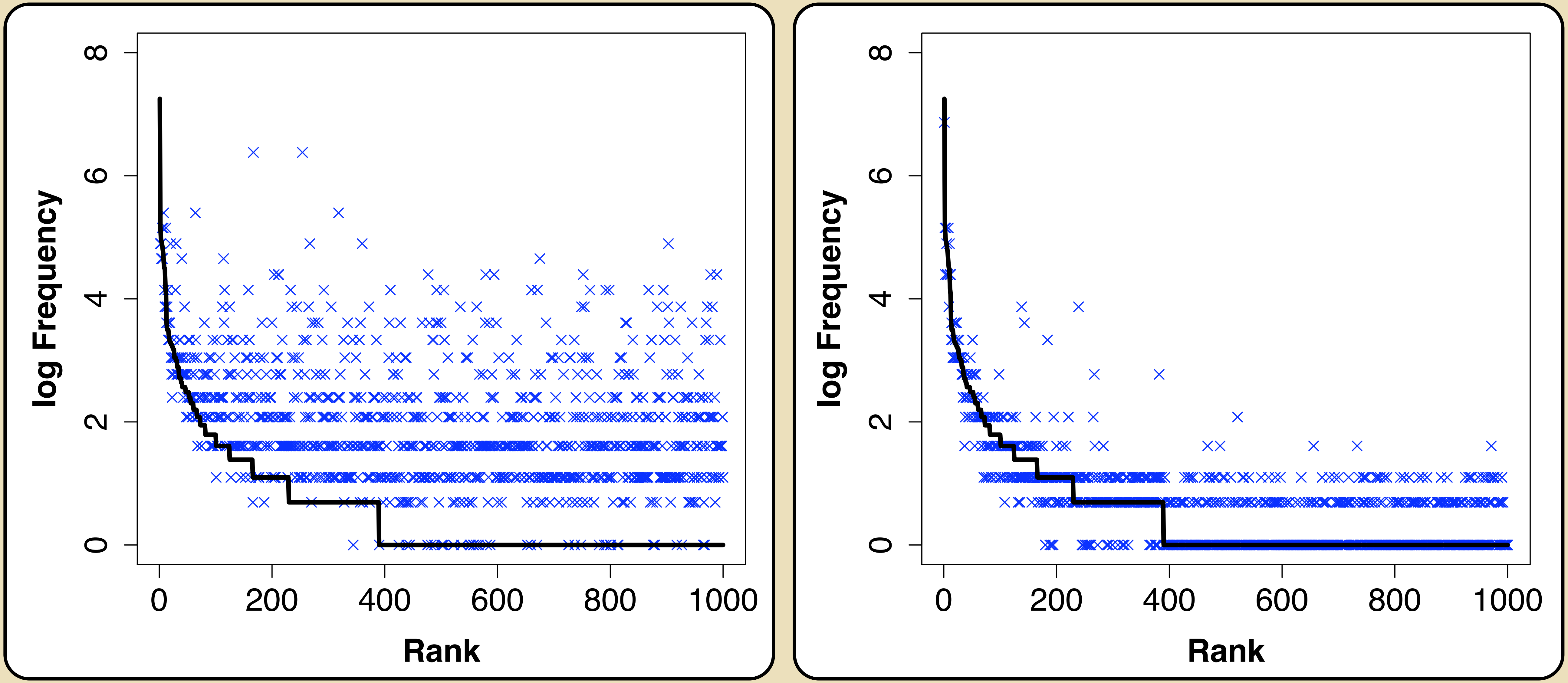


Outline

- Storing Static Counts
- Counting Online
- Experiments
- Additional Comments

Experiment: Count Accuracy

Count all trigrams in Gigaword, randomly query 1,000 values, compare to truth



100 MB

500 MB

Experiment: MT

Build counters with varying amounts of memory

TRUE	260MB	100MB	50MB	25MB	NO LM
22.75	22.93	22.27	21.59	19.06	17.35
-	22.88	21.92	20.52	18.91	-
-	22.34	21.82	20.37	18.69	-

(based on system of Post & Gildea '08)

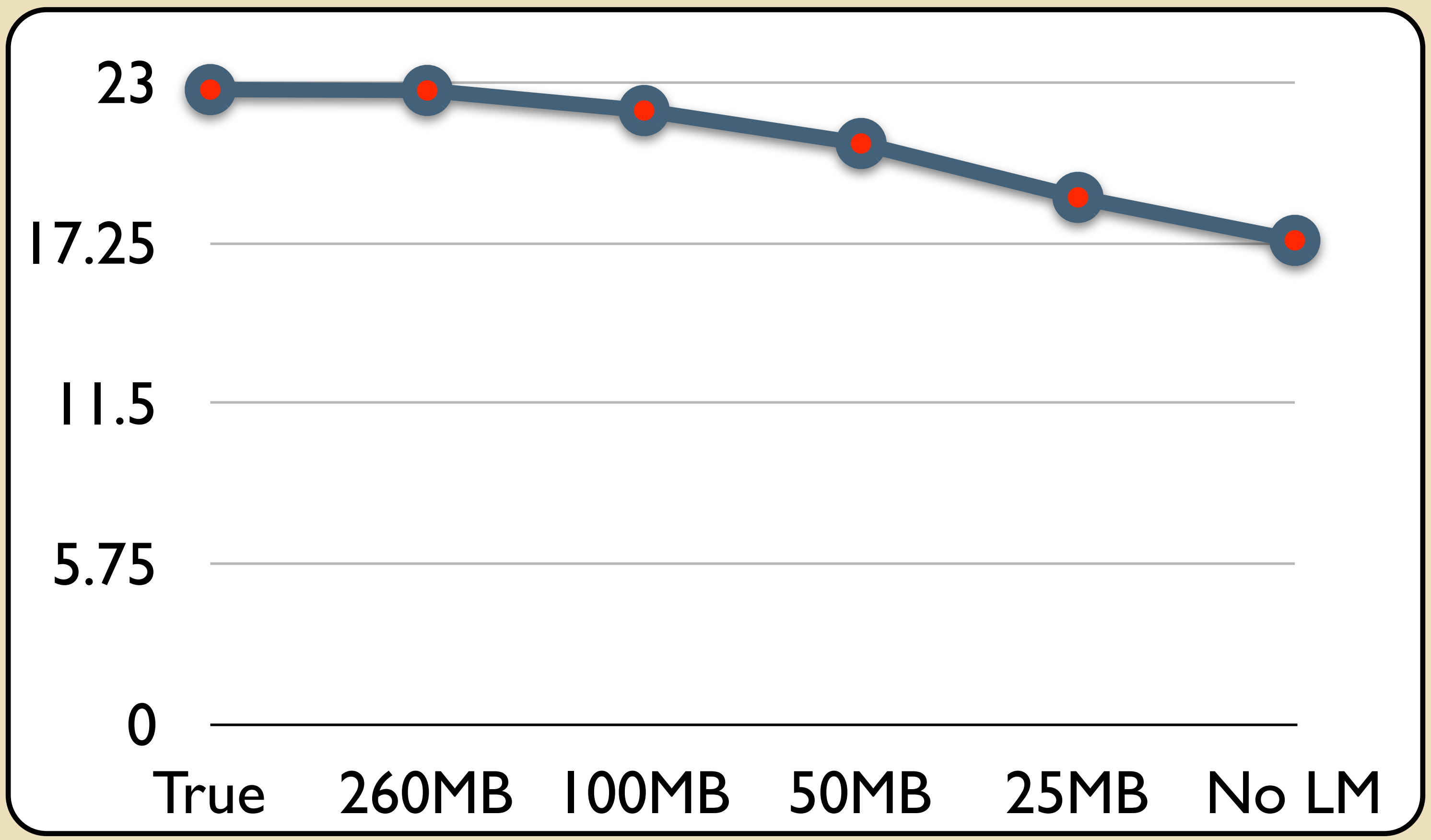
Experiment: MT ...

TRUE	260MB	100MB	50MB	25MB	NO LM
22.75	22.93	22.27	21.59	19.06	17.35
-	22.88	21.92	20.52	18.91	-
-	22.34	21.82	20.37	18.69	-

Three runs per counter size

Experiment: MT ...

TRUE	260MB	100MB	50MB	25MB	NO LM
22.75	22.72	22.00	20.83	18.89	17.35
-	(average)				-
-					-



Outline

- Storing Static Counts
- Counting Online
- Experiments
- Additional Comments

Related

- Applies method of Manku and Motwani ’02.
- Track most frequent elements in stream.
- Rare elements discarded.
- Strong guarantee on counts for top elements.

Streaming for large scale NLP: Language Modeling

Amit Goyal, Hal Daumé III, and Suresh Venkatasubramanian
University of Utah, School of Computing

NAACL 2009

Data that is not text

- Not just for Comp. Ling.
- E.g., count n-grams over "vocabularies" based on SIFT features.

Humans

- People store large amounts of information in their heads,
- and they do it online.
- Space efficient online counting provides additional area for interfacing with Cog. Sci. community.

Acknowledgements

- Ashwin Lall (co-author)

UNIVERSITY of ROCHESTER

Georgia Tech

Acknowledgements

- Ashwin Lall (co-author)
- David Talbot,
  Miles Osborne

Acknowledgements

- Ashwin Lall (co-author)
- David Talbot,
  Miles Osborne
- Matt Post,
  Nick Morsillo,
  Dan Gildea

UNIVERSITY of ROCHESTER

Questions?

www.cs.rochester.edu/~vandurme

www.cc.gatech.edu/~alall