Streaming algorithms, Apache DataSketches, and new results on corsets

Lee Rhodes
Distinguished Architect
Oath, Inc.

Kevin Lang
Principal Scientist
Oath, Inc

Edo Liberty
Founder
HyperCube

Alaxander Saydakov
Senior Software Engineer
Oath, Inc

Jon Malkin
Senior Scientist
Oath, Inc

Justin Thaler
Assistant Professor
Georgetown University.

Zohar Karnin
Principal Scientist
Amazon

Nikita Ivkin
Applied Scientist
Amazon
Single machine data processing
Distributed storage
Distributed compute (map/reduce, MPI, ...)

The World

Data + Compute

Data + Compute

Data + Compute

Data + Compute

Data + Compute

Data + Compute

Computation

Result
Distributed model (indexes, databases, Spark...)

The diagram shows a distributed model involving data and compute resources. The World is connected to multiple nodes labeled 'Data + Compute'. These nodes are further connected to 'Query' and 'Computation', leading to a 'Result'.
The streaming model
Mergeable Summaries
Unique Counting with Map Reduce

This is a shuffle operation which is very compute and network heavy!
Unique Counting with Mergeable Summaries

No shuffle operation is needed!
Data Mining with Traditional Windowing

Every dataset is processed 3 times for a model consisting of 3 days
Data Mining with Mergeable Summaries

Every dataset is processed once for a model consisting of the entire history
Dynamic Windowing with Mergeable Summaries
OLAP with Mergeable Summaries

“Median latency of IoT device call in Westeros January 2018”

“Median latency of IoT device call in The North Q1 2018”
IoT and Cloud Monitoring

Sending all the data

Logs

Database/
Analytics Engine
IoT and Cloud Monitoring

Sending only sketches

Sketch Based Analytics Engine
Some Basic Problems are Impossible
What can we do in this model?

**Items**
(terms, IP-addresses, events, clicks,...)
- Counting distinct elements
- Item frequencies
- Approximate Quantiles
- Moment and entropy estimation
- Approximate set operations
- Sampling

**Matrices**
(text corpora, recommendations, ...)
- Covariance estimation matrix
- Low rank approximation
- Sparsification

**Vectors**
(text documents, images, example features,...)
- Dimensionality reduction
- Clustering (k-means, k-median,...)
- Linear Regression
- Machine learning (some of it at least)
- Density Estimation / Anomaly detection

**Graphs**
(social networks, communications, ...)
- Connectivity
- Cut Sparsification
- Weighted Matching
Apache Data Sketches

sketches-core
Core Sketch Library.

Java  ★ 500  ⭐ 130  Apache-2.0  Updated an hour ago

>> brew tap DataSketches/sketches-cmd
>> brew install data-sketches

Production ready

Amazon Web Services™

Yahoo!

Splice Machine

Splunk

Druid
New Research

A high-performance algorithm for identifying frequent items in data streams.

[DLRT16] Anirban Dasgupta, Kevin J. Lang, Lee Rhodes, and Justin Thaler.

[KLL16] Zohar S. Karnin, Kevin J. Lang, and Edo Liberty.


[LMTU16] Edo Liberty, Michael Mitzenmacher, Justin Thaler, and Jonathan Ullman.

In this presentation

**Items**
(words, IP-addresses, events, clicks,...)
- Counting distinct elements
- Item frequencies
- Approximate Quantiles
- Moment and entropy estimation
- Approximate set operations
- Sampling

**Matrices**
(text corpora, recommendations, ...)
- Covariance estimation matrix
- Low rank approximation
- Sparsification

**Vectors**
(text documents, images, example features,...)
- Dimensionality reduction
- Clustering (k-means, k-median,...)
- Linear Regression
- Machine learning (some of it at least)
- Density Estimation / Anomaly detection

**Graphs***
(social networks, communications, ...)
- Connectivity
- Cut Sparsification
- Weighted Matching
Counting Distinct Elements

N. Alon, Y. Matias, and M. Szegedy. The space complexity of approximating the frequency moments
E. Cohen. All-distances sketches, revisited: HIP estimators for massive graphs analysis
E. Cohen and H. Kaplan. Summarizing data using bottom-k sketches
G. Cormode. Sketch techniques for massive data
P. Flajolet and G. Nigel Martin. Probabilistic counting algorithms for data base applications
D. M. Kane, J. Nelson, and D. P. Woodruff. An optimal algorithm for the distinct elements problem
M. Thorup. Bottom-k and priority sampling, set similarity and subset sums with minimal independence
Problem Definition

Approximate the number of distinct items in the stream

- # of unique IPs are important statistics of networks
- # of different customers using web services
- # of unique keys in a database join table
- ...

• # of unique IPs are important statistics of networks
• # of different customers using web services
• # of unique keys in a database join table
• ...
General Hashing Idea

- Map all entries to the interval (0,1) using a hash function
- Keep only $k$ values smaller than some threshold $\theta$.
- From $(k, \theta)$ we can approximate the number of unique items.
Our results

• Generalize a family of algorithms (Adaptive sampling, KMV)
• New Variance bounds for all such algorithms
• New tradeoffs between accuracy, space, and update time (alpha alg’)
• Very careful implementation
Experimental Results

Equal Space Comparison of \((\text{Total Processing Time}) / N\)

- KMOV
- Adapt
- Alpha

Equal Space Comparison of Standard Error

- KMOV
- Adapt
- Alpha

Time in Nanoseconds

\[ N = \text{Number of Unique Items in Stream} \]
CPC - Compressed Probabilistic Counting

- A Better algorithm than \textit{(HyperLogLog) HLL} was recently invented by \textbf{Kevin Lang}
- The result will be published by the datasketches group soon
Weighted Item frequencies

Space-optimal heavy hitters with strong error bounds. J. R. Berinde, P. Indyk, G. Cormode, and M. J. Strauss
An optimal algorithm for \(1\)-heavy hitters in insertion streams and related problems A. Bhattacharyya, P. Dey, and D. P. Woodruff
Finding frequent items in data streams M. Charikar, K. Chen, and M. Farach-Colton
Methods for finding frequent items in data streams G. Cormode and M. Hadjieleftheriou
Approximate frequency counts over data streams G. S. Manku and R. Motwani.
Efficient computation of frequent and top-k elements in data streams A. Metwally, D. Agrawal, and A. El Abbadi.
Finding repeated elements J. Misra and D. Gries.

A High-Performance Algorithm for Identifying Frequent Items in Data Streams
   Daniel Anderson Pryce Bevin, Kevin Lang, Edo Liberty, Lee Rhodes, Justin Thaler
Problem Definition

\[ w(\bullet) = \sum w_i \]

\[ |w'(x) - w(x)| \leq \varepsilon W \]

\[ W = \sum_i w_i \]
Our Contributions

• Improved streaming algorithm for weighted updates

• Improved merging procedure

• Improved Estimator

• Careful implementation
Comparable Error

Error With Equal Space

- SMED
- RBMC
- SMIN
- MHE

Maximum Error

Amount Of Bytes

8,398,782
5,179,022
3,153,850
1,478,646
9,437,238
Significantly Faster Updates
Weighted Item Frequencies Application
Ailon, Karnin, Maarek, Liberty,
Threading Machine Generated Email, WSDM 2013
If $b$ should be threaded with $a$ then the lift should be large

$$\text{lift}(a, b) = p(b|a)/p(b) \gg 1$$

Alas, computing all pair conditional probability is impossible!

$$\text{lift}(a, b) = \frac{n(b, a)}{n(b)n(a)} = n(b, a)w(b)$$

This is possible with weighted frequency sketching!
Threading Machine Generated Email

PayPal.com:
“You submitted an order in the amount of * usd to overstock.com.”

overstock.com:
“Overstock.com password reset request.”

payless.com
“Order confirmation”

payless.com
“You order is shipped”

C=193
w=12,098

C=632
w=1,221

C=652
w=1,300

C=769
w=1,490

C=753
w=1,395

overstock.com:
“Order confirmation”

overstock.com:
“You overstock.com order has shipped.”

C=153
w=704

C=1,742
w=6,446
Threading Machine Generated Email

1. Order Confirmation (retail) → Shipping Notification (64%)
2. Utility bill payment due → Payment received (44%)
3. Insurance payment due → Service cancellation (53%)
4. Insurance payment due → Service cancellation (15%)
5. Shipping Notification → Order Confirmation (retail) (19%)
Streaming quantiles

Manku, Rajagopalan, Lindsay. Random sampling techniques for space efficient online computation of order statistics of large datasets.
Munro, Paterson. Selection and sorting with limited storage.
Greenwald, Khanna. Space-efficient online computation of quantile summaries.
Wang, Luo, Yi, Cormode. Quantiles over data streams: An experimental study.
Greenwald, Khanna. Quantiles and equidepth histograms over streams.
Agarwal, Cormode, Huang, Phillips, Wei, Yi. Mergeable summaries.
Felber, Ostrovsky. A randomized online quantile summary in $O((1/\epsilon) \log(1/\epsilon))$ words.

**Lang, Karnin, Liberty, Optimal Quantile Approximation in Streams.**

**Ivking, Lang, Karnin, Liberty, Braverman, Streaming quantiles algorithms with small space and update time**
Problem Definition

Sketch the stream to estimate $|R' - R| < \varepsilon n$
<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Simple?</th>
<th>Mergeable</th>
<th>Space Complexity</th>
</tr>
</thead>
<tbody>
<tr>
<td>Uniform sampling</td>
<td>✓</td>
<td>✓</td>
<td>$1/\varepsilon^2$</td>
</tr>
<tr>
<td>Greenwald Khanna (GK)</td>
<td>✗</td>
<td>✗</td>
<td>$\log(n)/\varepsilon$</td>
</tr>
<tr>
<td>Felber-Ostrovsky</td>
<td>✗</td>
<td>✗</td>
<td>$\log(1/\varepsilon)/\varepsilon$</td>
</tr>
<tr>
<td>Manku-Rajagopalan-Lindsay (MRL)</td>
<td>✓</td>
<td>✓</td>
<td>$\log^2(n)/\varepsilon$</td>
</tr>
<tr>
<td>Agarwal, Cormode, Huang, Phillips, Wei, Yi</td>
<td>✓</td>
<td>✓</td>
<td>$\log^{3/2}(1/\varepsilon)/\varepsilon$</td>
</tr>
<tr>
<td>Karnin, Lang, Liberty</td>
<td>✓</td>
<td>✓</td>
<td>$\sqrt{\log(1/\varepsilon)/\varepsilon}$</td>
</tr>
<tr>
<td>Karnin, Lang, Liberty</td>
<td>✓</td>
<td>✗</td>
<td>$\log^2 \log(1/\varepsilon)/\varepsilon$</td>
</tr>
<tr>
<td>Karnin, Lang, Liberty</td>
<td>✗</td>
<td>✗</td>
<td>$\log \log(1/\varepsilon)/\varepsilon$</td>
</tr>
<tr>
<td>Still open...</td>
<td>✓</td>
<td>✓</td>
<td>$\log \log(1/\varepsilon)/\varepsilon$</td>
</tr>
</tbody>
</table>
The basic buffer idea

Buffer of size k
The basic buffer idea

Stores k stream entries
The basic buffer idea

The buffer sorts k stream entries
The basic buffer idea

Deletes every other item
The basic buffer idea

And outputs the rest with double the weight

5 3 0
The basic buffer idea

$$R(x) = 2$$

0 1 3 4 5 7

$$R'(x) = 2$$

0 3 5

$$R'(x) = 6$$

$$x$$

$$R'(x) = 4$$

1 4 7

$$x$$
The basic buffer idea

Repeat \( \frac{n}{k} \) time until the end of the stream

\[ |R'(x) - R(x)| < \frac{n}{k} \]
Manku-Rajagopalan-Lindsay (MRL) sketch

$\log_2(n)$ Buffers of size $k$

$|R'(x) - R(x)| \leq n \log_2(n)/k$
Manku-Rajagopalan-Lindsay (MRL) sketch

If we set \( k = \log_2(n)/\varepsilon \)

We get \( |R'(x) - R(x)| \leq \varepsilon n \)

And we maintain only \( \log_2^2(n)/\varepsilon \) items from the stream!
Greenwald-Khanna (GK) sketch

Uses a completely different construction

It gets \[ |R'(x) - R(x)| \leq \varepsilon n \]

And maintains only \( O(\log(n)/\varepsilon) \) items from the stream!
Buffers of size $k \log(1/\varepsilon)$ start sampling after $O(1/\varepsilon^2)$ items from the stream.

Reduces space usage to $\log^2(1/\varepsilon)/\varepsilon$ items from the stream.
$R(x) = 1$

$R'(x) = 2$

$R'(x) = 0$

$x$

$R'(x)$ is a random variable now and

$E[R'(x)] = R(x)$

Reduces space usage to $\log^{3/2}(1/\varepsilon)/\varepsilon$ items from the stream.
Reduces space usage to $\sqrt{\log(1/\varepsilon)/\varepsilon}$ items from the stream.
Lang, Karnin, Liberty (2)

- Exponentially decreasing buffer sizes

GK Sketch

Reduces space usage to $\log \log(1/\varepsilon)/\varepsilon$ items from the stream.

Which is Optimal!
Experimental Results

More experiments:  
https://datasketches.github.io/docs/Quantiles/KLLSketch.html  
https://datasketches.github.io/docs/Quantiles/KLLSketchVsTDigest.html
Even Newer Experimental Results

Ivking, Lang, Karnin, Liberty, Braverman, Streaming quantiles algorithms with small space and update time
What else can we do in this model?

**Items**
(words, IP-addresses, events, clicks,...)
- Counting distinct elements
- Item frequencies
- Approximate Quantiles
- Moment and entropy estimation
- Approximate set operations
- Sampling

**Vectors**
(text documents, images, example features,...)
- Dimensionality reduction
- Clustering (k-means, k-median,...)
- Linear Regression
- Machine learning (some of it at least)
- Density Estimation / Anomaly detection

**Matrices**
(text corpora, recommendations, ...)
- Covariance estimation matrix
- Low rank approximation
- Sparsification

**Graphs**
(social networks, communications, ...)
- Connectivity
- Cut Sparsification
- Weighted Matching
Very new results about Coresets and Machine Learning
What is PAC learning?

Distribution  \rightarrow\text{Sampled Data}  \quad x_i
What are Coresets?

Sampled Data

$x_i$

Coreset

$\{x_i, w_i\}$
Density Estimation

\[ F(q) = \sum_i f(x_i, q) \]

\[ \tilde{F}(q) = \sum_{i \in S} w_i f(x_i, q) \]
Classification/Regression

\[ F(q) = \sum_i f(x_i, q) \]

\[ \tilde{F}(q) = \sum_{i \in S} w_i f(x_i, q) \]
ML and Coresets are intimately connected

Rademacher Complexity

$$R_m = \mathbb{E}_\sigma \max_q \frac{1}{m} | \sum_{i=1}^m \sigma_i f(x_i, q)|$$

$$K_m \approx O(c/\sqrt{m})$$

Sample Complexity

Model Generalization

Class Discrepancy

$$D_m = \min_{\sigma} \max_q \frac{1}{m} | \sum_{i=1}^m \sigma_i f(x_i, q)|$$

$$D_m = O(c/m)$$

Coreset Complexity

Sketch Generalization
Warmup exercise...

\[
\min_{\sigma} \left\| \sum_{i=1}^{n} \sigma_i x_i \right\| \leq \sqrt{d}
\]

Does not depend on \( n \)

\[
\mathbb{E}_{\sigma} \left\| \sum_{i=1}^{n} \sigma_i x_i \right\| \approx \sqrt{n}
\]

That’s encouraging.....
Universal Vector Balancing Lemma

Lemma [Karnin, Liberty, 2019]: For any set of unit vectors $x_i \in \mathbb{R}^d$ there exist signs $\sigma$ such that for all $k$ simultaneously

$$
\left\| \sum_{i=1}^{n} \sigma_i x_i^\otimes k \right\| \leq \sqrt{d} \cdot \text{poly}(k)
$$

Still does not depend on $n$!
Results

- Sigmoid Activation Regression, Logistic Regression
- Covariance approximation, Graph Laplacians Quadratic forms
- Gaussian Kernel Density estimation

All have the above have Class Discrepancy of $D_m = O(\sqrt{d}/m)$

1) coresets of size $O(\sqrt{d}/\varepsilon)$
2) Streaming Coresets of size $O\left(\sqrt{d}/\varepsilon \cdot \log^2 \left(\varepsilon n/\sqrt{d}\right)\right)$
3) Randomized Streaming Coresets of size $O\left(\sqrt{d}/\varepsilon \cdot \log^2 \log(|Q_\varepsilon|/\delta)\right)$

Resolves the open problem
See Philips an Tai 2018
Results for density estimation
There is still a lot of work...

\[
f(x, q) = \begin{cases} 
1 & \text{if } \langle q, x \rangle > 0 \\
0 & \text{else}
\end{cases}
\]

Classification with 0-1 loss

\[
f(x, q) = \exp(-\|x - q\|)
\]

Exponential Kernel Density

\[
D_m = ?
\]

\[
D_m = ?
\]


Dan Feldman and Michael Langberg. A unified framework for approximating and clustering data. 2011

Jeff M. Phillips and Wai Ming Tai. Near-optimal coresets of kernel density estimates

Elad Tolochinsky and Dan Feldman. Coresets for monotonic functions with applications to deep learning.

Sariel Har-Peled, Dan Roth, and Dav Zimak. Maximum margin coresets for active and noise tolerant learning. *IJCAI 2007*

Sariel Har-Peled and Akash Kushal. Smaller coresets for k-median and k-means clustering. *The21st ACM Symposium on Computational Geometry, Pisa, Italy, June 6-8, 2005*

Gurmeet Singh Manku, Sridhar Rajagopalan, and Bruce G. Lindsay. Random sampling techniques for space efficient online computation of order statistics of large datasets.


Olivier Bachem, Mario Lucic, and Andreas Krause. Practical coreset constructions for machine learning