Data Sketching for Real Time Analytics

Theory and Practice

Daniel Ting
Tableau Research

Jon Malkin
Apache Datasketches
Verizon

Lee Rhodes
Apache Datasketches
Verizon

Thank you

Alex Saydakov (Verizon)
Edo Liberty (Hypercube)
Justin Thaler (Georgetown)
Jelani Nelson (Berkeley)
Graham Cormode (U Warwick)
Jacques Dark (U Warwick)
Charlie Dickens (U Warwick)
Otmar Ertl (Dynatrace)
Pedro Reviriego (U Carlos III de Madrid)
Edith Cohen (Google, U Tel Aviv)

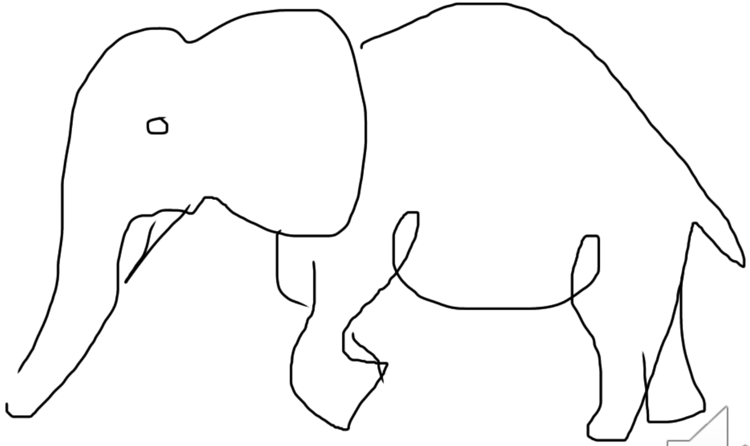
Why data sketching

- Big Data problem
- There is a tremendous amount of data right now
  - Continues to grow
  - IDC estimate 2x in 4 years

- Resulting problems
  - Scale
  - Speed
  - Cost

- Fundamental problem of Big Data is

  … that it is Big.



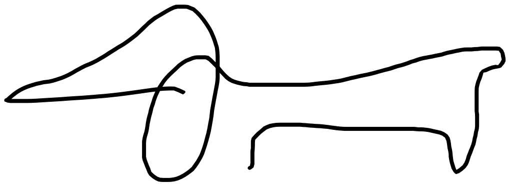
Why data sketching

- Data sketching is a way to make Big Data small

- What possibilities are opened up if speed and cost are not worries?
    - Real–time analytics?
    - New UIs? Automated analyses?
    - Potentially tackle problems that would not be tried before?

Definition

Data Sketch = Lossy compression or summarization of data that provides answers to a set of questions of interest



Main properties

Advantages

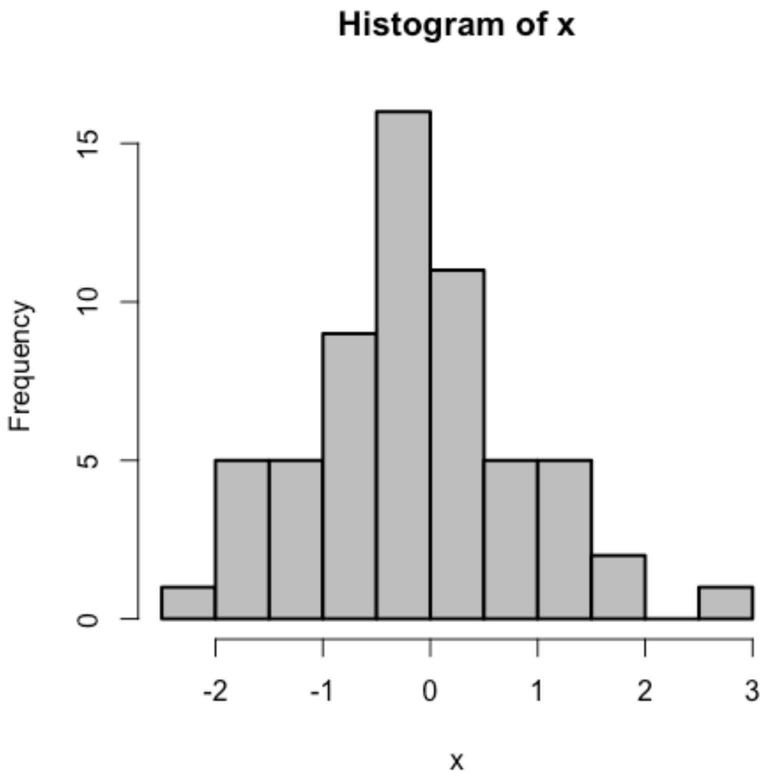
- Space: Saves (a lot) of space: can be **< 1/1000th** of the space
- Speed: Typically **1-pass** and often parallelizable when building them and nearly instantaneous when reading them

Downsides (no free lunch)

- Universe of questions / queries must often be determined in advance
- Approximation error
  - Often with a theoretical bound

Basic Examples

- Simple random samples
- Histograms
  - Size does not grow with the length of the data
  - Generates approximations for almost any univariate statistic
  - Discards any multivariate information
- Pre-computed aggregates
- Statistical models



Approximate query processing examples

Typical sketches

- Eliminate an expensive operation or data structure used in exact query processing
- Provide accuracy guarantees on worst case inputs

Examples:

- Cardinality estimation / Distinct counts
- Frequent items / Top-k
- Quantiles
- Subset sums (SELECT SUM(...) … WHERE ..)
- Approximate set membership (Bloom filters)
- Set similarity (MinHash)

Advanced methods

Numerical linear algebra

- Random projections (any L_p)
- Sketched SVD / Frequent Directions
- Leverage score sampling for accelerating regression

Other advanced methods

- Nyström approximations for low rank approximations for kernel matrices
- Core sets for accelerating statistical / ML model fitting
- Graph sketches

Wide range of applications

- Networking
  - Monitoring
  - Security: attack detection
    - Distributed Denial of Service
    - Port scanning
  - Caching
- Biology
  - Frequent k-mers
- Databases
  - Approximate set membership
    - File / partition skipping
  - Query planning
    - Join size cardinality estimation
  - AQP
- Web / Information retrieval
  - Fast similarity measures / duplicate detection
  - Blacklists
- Business Analytics
  - Fast, interactive analytics
- ML / Data science / Linear algebra
  - Fast computation
  - Large scale analyses

Applications

Sketches often help in

- Scaling up analyses
- Accelerating tasks
- Extremely memory limited situations
- Reducing communication costs

Outline

- Describe several sketches
  - Distinct counting
  - Quantiles
  - Sampling
  - Frequent items
  - Linear sketches
  - “Advanced” methods
- Privacy and sketches
- Highlight key techniques and takeaways as we go through each sketch

What to get out of the tutorial

Practitioners

- Navigating data sketches
    - Capabilities of sketches
    - How to understand sketching guarantees
    - How to choose a sketch
    - Where to look for more information

Researchers

- Statistical approach to sketching
- New developments
- Some open problems

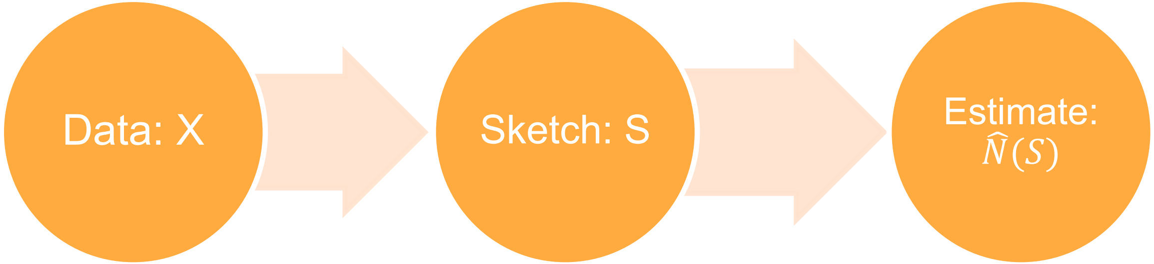
Approximate Distinct counting

Problem

Cardinality estimation

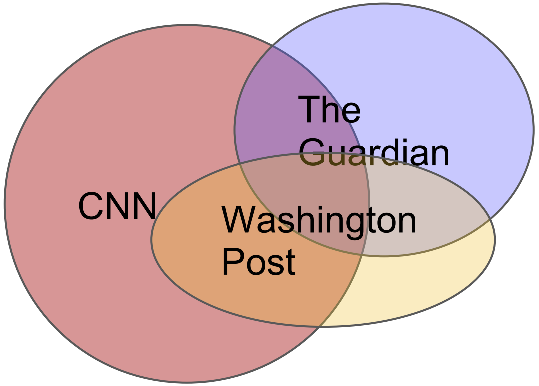
- Given a data stream X or multiset $x_1, x_2, \ldots, x_t$
- Compute the cardinality $N = |\{x_1, x_2, \ldots, x_t\}|$

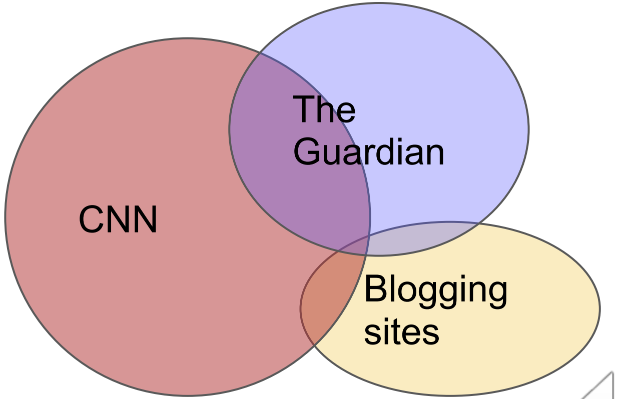
Approximate distinct counting sketch



Applications

- Advertisers
    - Reach estimation: How many distinct users saw an ad campaign?
    - Audience size: If I’m already publishing ads on CNN and The Guardian, should I spend money advertising on the Washington Post or a basket of blogging sites.
    - Demographic breakdowns





Distinct counting

Ad problem:

How many distinct users saw an ad / website / post?

Database query:

select count(distinct user)
from very_large_stream

Execution (Naive):

- Insert each user into a hash table
- Compute the number of keys in the table

Cost:

- N = 1 million,   64 bits / user
- Hash table size per ad: 16 MB
- 100K ads / day ⇒ 1.6 TB / day
- Over 1 month ⇒ 50 TB / month
- With demographic breakdowns, much higher costs

Distinct counting

Ad problem:

How many distinct users saw an ad / website / post?

Database query:

select count(distinct user)
from very_large_stream

Sketch Cost:

- Sketch size (4 bit HLL) m=2 KB
- Relative error: 1.5%
- **8000x** improvement in space
- 50 TB → 6 GB (single machine)

Main challenge:

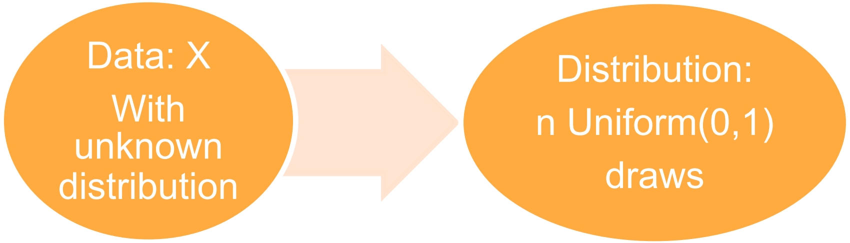
- Handling duplicates

Technique: Hash-based / Coordinated sampling

How distinct counting sketches work

Data: $x_1, x_2, \ldots, x_t$

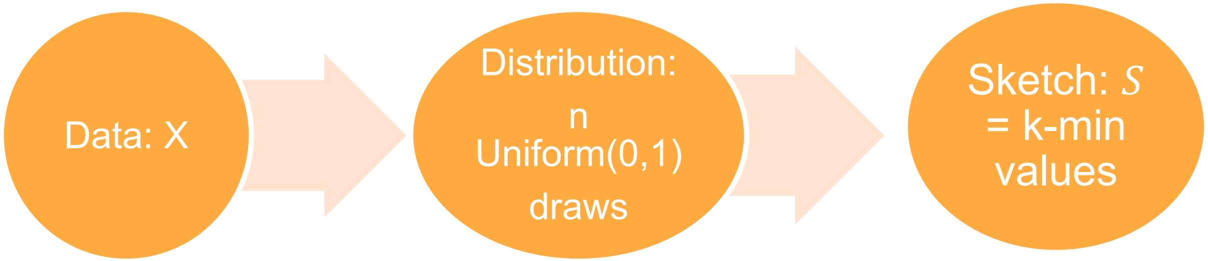
- Assume there is a universal hash function $x_i \rightarrow Z_i \sim$ Uniform(0,1)
  - Results in n distinct values
  - Distribution is known and depends only on the desired value n



How distinct counting sketches work

Data: $x_1, x_2, \dots, x_t$

- Assume there is a universal hash function $x_i \rightarrow Z_i \sim \text{Uniform}(0,1)$
  - Results in n distinct values
  - Distribution is known and depends only on the desired value n



- MinCount / kMV Sketch: Keep smallest k distinct values

General paradigm (for statistical data sketching)

1. Convert data into a random process with some known distribution
    - Parameters of the distribution are answers to the queries of interest
    - Random process can be updated and stored efficiently

2. Perform parameter estimation to extract answers of queries of interest

Estimating the count

Imagine counting the total trash on a beach.

Total trash =
(Trash rate) x
(length of beach)

$$Trash\ Rate = \frac{k}{Distance\ to\ find\ k\ pieces\ of\ trash}$$

Values Z_i = location along beach

vaidehi shah https://flic.kr/p/JBUH9y CC BY 2.0

Estimating the count

Imagine counting the total trash on a beach.

Total trash = (Trash rate) x (length of beach)

$$Trash\ Rate = \frac{k}{Distance\ to\ find\ k\ pieces\ of\ trash}$$

Values Z_i = location along beach

MinCount

Sketch: k smallest values $Z_{(1)} \leq Z_{(2)} \leq \ldots \leq Z_{(k)}$

Estimate: $\hat{N} = \dfrac{k-1}{Z_{(k)}}$

Properties:

- Know the exact distribution of the estimate
- Estimate is unbiased
- Error is $\sigma(\widehat{N}) \approx \frac{N}{\sqrt{k-2}}$
  - 10% error if k = 102
- Minimum Variance Unbiased Estimator

Advantages / disadvantages

Sketch is (essentially) a random sample.

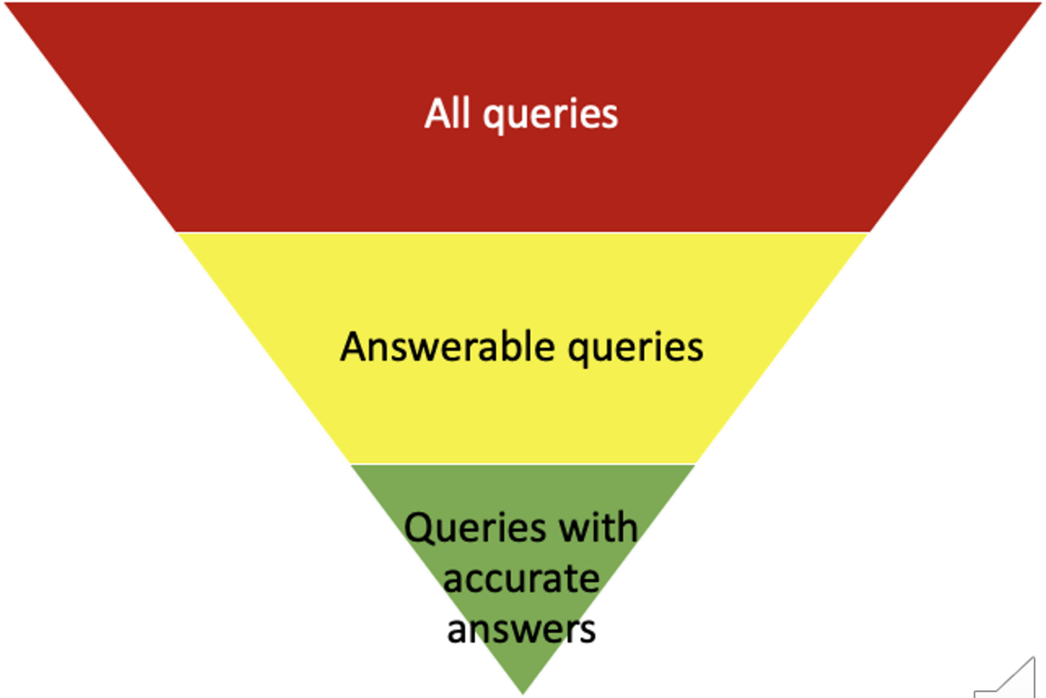
Advantages	Disadvantages
● Very flexible ● Supports all set operations including 　○ unions 　○ intersections 　○ set difference ● Filtering	● But is much larger than other distinct counting sketches (up to 15x larger for large cardinalities) ● Slower worst case updates

Same idea is the basis for Theta sketch and Tuple sketches in the Apache Datasketches library.

Flexibility vs efficiency

Sketches reduce space by restricting

- the queries that it can answer and
- hence, the amount of information it needs to store



Technique: Quantization

Girl with a Mandolin
Picasso, 1910

Sketch components

Summarization	Convert data into a (random) process that preserves the answers to a set of given queries
Encoding	Find representations that store the summarization in small space and allow for fast access.
Estimation	Extract answers and error estimates from the sketch

HyperLogLog

HyperLogLog makes 2 modifications to MinCount

- k smallest ⇒ smallest in k bins (fast worst case updates)
- Quantize values (space efficiency)

Continuous HLL (fast worst case updates)

- Sketch: $S \in \mathbb{R}^k$
- Hash item $x_i \rightarrow$ (Bin_i, Z_i)
  - $Bin_i \sim$ Uniform({1, 2, …., k})
  - $Z_i \sim$ Uniform(0,1)
- S_b = Minimum Z_i value assigned to bin b

Cardinality estimate:

$$Rate = \left(\frac{1}{k}\sum_{b=1}^{k} S_b\right)^{-1} = \frac{1}{Avg\ distance\ between\ items}$$

$$\hat{N} = k \cdot Rate = \frac{k^2}{\sum_{b=1}^{k} S_b}$$

HLL (+ quantization)

- Sketch: $S \in \mathbb{R}^k$
- Hash item $x_i \rightarrow (Bin_i, Z_i)$
  - $Bin_i \sim \text{Uniform}(\{1, 2, \ldots., k\})$
  - $Z_i \sim \text{Uniform}(0,1) \implies \tilde{Z}_i = ceiling(-\log_2 Z_i)$
- $\tilde{S}_b$ = Minimum $\tilde{Z}_i$ value assigned to bin b

HLL (+ discretization)

- Sketch: $S \in \mathbb{R}^k$
- Hash item $x_i \rightarrow (Bin_i, Z_i)$
  - $Bin_i \sim$ Uniform({1, 2, …., k})
  - $Z_i \sim$ Uniform(0,1)
- $\tilde{S}_b$ = Minimum Z_i value assigned to bin b
- $S_b = ceiling\left(-\log_2 \tilde{S}_b\right)$

Cardinality estimate:

Rescale back to (0,1)

Correct for quantization

$$Rate = \alpha \left(\frac{1}{k} \sum_{b=1}^{k} 2^{-S_b} \right)^{-1} = \frac{\alpha}{Avg\ distance\ between\ items}$$

$$\hat{N} = k \cdot Rate = \frac{\alpha k^2}{\sum_{b=1}^{k} 2^{-S_b}} \qquad \alpha \approx \frac{0.7213}{1+1.079/k}$$

HLL vs MinCount

	MinCount	HLL
Error (large cardinalities)	$\approx \frac{N}{\sqrt{k}}$	$\approx 1.04\frac{N}{\sqrt{k}}$
Size	32 or 64 bits per entry	4-6 bits per entry
Operations	Unions, Intersections, Set difference, Filtering	Unions

Navigating sketching literature

Fishing boats at sea
Monet, 1868

Distinct counting sketches

Subset of distinct counting sketches

- Probabilistic counting (FM85)
- Linear probabilistic counting (LPCA)
- MinCount / Bottom-k
- Theta / Tuple sketch
- Self-learning bitmap
- Multi-resolution bitmap
- α - stable distinct counting
- Optimal distinct counting
- HyperLogLog (HLL)
    - HLL++
    - Streaming HLL
    - Virtual HLL / CountMin-HLL
    - Other variations

What should you pick??

Distinct counting sketches

Subset of distinct counting sketches

- Probabilistic counting (FM85)
- Linear probabilistic counting (LPCA)
- MinCount / Bottom-k
- Theta / Tuple sketch
- Self-learning bitmap
- Multi-resolution bitmap
- α - stable distinct counting
- Optimal distinct counting
- HyperLogLog (HLL)
    - HLL++
    - Streaming HLL
    - Virtual HLL / CountMin-HLL
    - Other variations

What should you pick??

Distinct counting sketches

Subset of distinct counting sketches

- Probabilistic counting (FM85)
- Linear probabilistic counting (LPCA)
- MinCount / Bottom-k
- Theta / Tuple sketch
- Self-learning bitmap
- Multi-resolution bitmap
- α - stable distinct counting
- Optimal distinct counting
- HyperLogLog (HLL)
  - HLL++
  - Streaming HLL
  - Virtual HLL / CountMin-HLL
  - Other variations

What should you pick??

It depends.

The ”optimal algorithm” turns out to be highly inefficient even though it has optimal space complexity.

Often a disconnect between theoretical results and practice

HLL and Sampling based sketches (MinCount, Theta, etc.) are particularly well-rounded.

Many choices for implementing HLL

Good choices

- Sparse representation from HLL++ in Heule et al (2013)
- 4 bit bins using offset from Presto / Druid / Datasketches implementation
- Improved raw estimator from Ertl (2017)
- Error estimator from Ting (2019)
- Compression from Scheuermann and Mauve (2007), Lang (2017)
- Streaming HIP estimator + error from Cohen (2014) and Ting (2014)

Questionable choices

- Sacrifice ability to do unions for some efficiency gains using HLL-TailCut+

Key Property: Mergeability

Mergeability

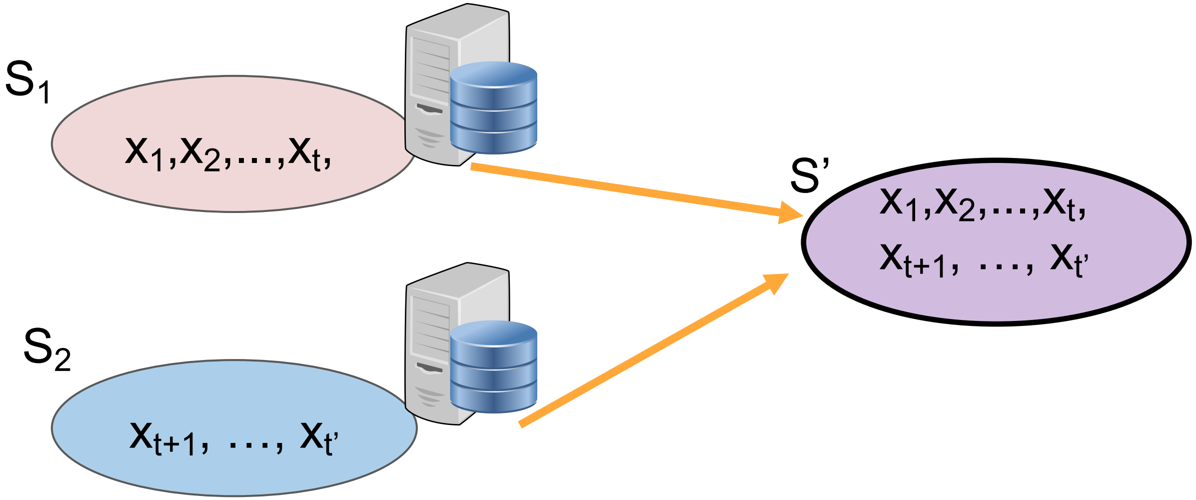
Given: A sketch construction algorithm S that yields estimates with error of scale σ.

Definition: S is *mergeable* if there is a function which takes sketches $S(D_1)$ and $S(D_2)$ yielding error of scale σ and generates a sketch S' whose estimate has error of scale σ as well.

Mergeability

Important for

- Distributed processing / reliability
- Further aggregation
    - E.g. over time or demographics



Real problem:
Distinct counting for
many sets

Advertising problem

Question: How many distinct users saw an ad *broken down by day, age, gender, country, and device type?*

- 100K ads
- 30+ days
- 5+ age buckets
- 2 genders
- 100+ countries
- 5+ device types

⇒ 30 * 5 * 2 * 100 * 5 = 150,000 counters / ad

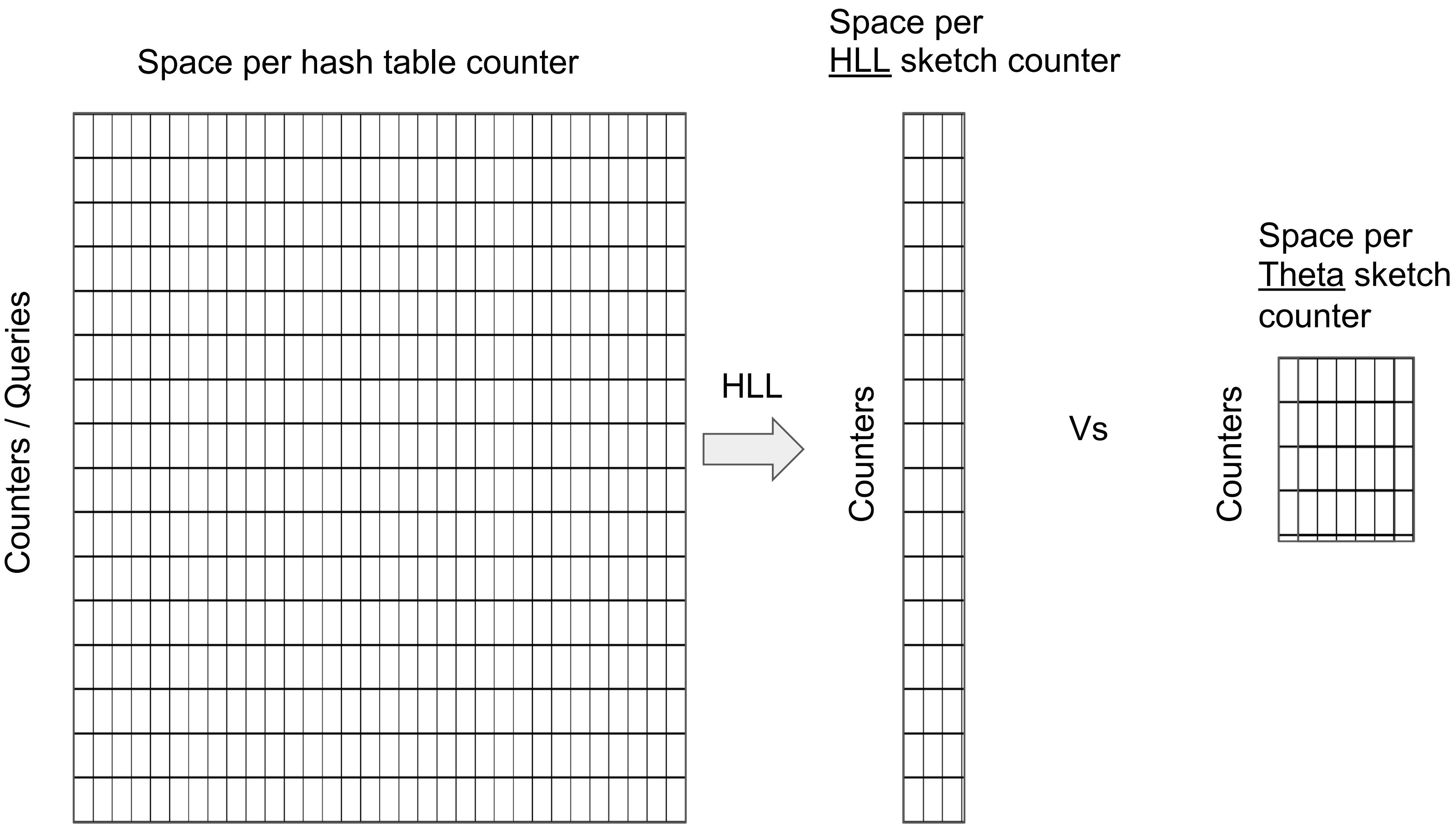
⇒ 15 billion counters ⇒ 30 TB with 2KB sketches

Solution 1: Intersections

- 100K ads
- 30+ days
- 5+ age buckets
- 2 genders
- 100+ countries
- 5+ device types

⇒ 30 + 5 + 2 + 100 + 5 = 142 counters / ad

⇒ 14.2 million counters (vs 15 billion)


Space per hash table counter
Counters / Queries
HLL
Space per
HLL sketch counter
Counters
Vs
Space per
Theta sketch
counter
Counters

Intersections using Inclusion-Exclusion

Intersection cardinality estimates can be computed using inclusion-exclusion:

$$|A \cap B| = |A| + |B| - |A \cup B|$$

- But they are often terrible
- Error tends to be proportional to the largest estimate, i.e. $|A \cup B|$
    - If the intersection has Jaccard similarity: $\frac{|A \cap B|}{|A \cup B|} = 0.1$
    - And sketch has cardinality estimates with error of 2%
    - The estimated intersection has error of approximately $2\% \cdot |A \cup B| = 20\% \cdot |A \cap B|$

- Even worse if taking multiple intersections

Sketches for Intersections

Sampling based sketches are

- much larger than HLL for single counts,
- but are often much better for estimating intersections

Examples:

- MinCount / Bottom-k
- Theta / Tuple sketch
  - In Apache Datasketches

Other (union only) solutions

- Sparse representations for low cardinalities
  - HLL++ (Heule 2013)
  - Partial solution that makes low cardinality counters smaller
- Counter-sharing
  - Virtual HLL (Xiao et al 2015)
  - Count-HLL (provably correct, Ting 2019)

Sampling

Sampling

Advantages

- Extremely flexible
  - Same sample can be used to answer many questions
- Easy to work with
  - Often requires no change or only adding a weight to downstream methods

Disadvantages

- Less space efficient than specialized sketches

Previous tutorial: Cormode and Duffield, KDD 2014

Subset sum problem

What are the total sales with a promotional discount? by product category?

Problem: compute the sum

$$V = \sum_{i \in \mathcal{I}} x_i$$

For any subset of indices $\mathcal{I}$

Sums

Despite simple form of the subset sum problem:

- A distinct count is a sum
- Intersections are subset sums
- Parametric statistical / ML models are (often) asymptotically sums

Problem: How to do better than uniform sampling?

Technique: Horvitz-Thompson

Horvitz-Thompson Estimator

A way to estimate sums for samples drawn ***without replacement***.

Denote:

- $Z_i = 1$ if x_i is in the sample and 0 otherwise
- $\pi_i = P(Z_i = 1)$

The Horvitz-Thompson (HT) estimator is

$$\hat{V} = \sum_{i \in \mathcal{I}} \frac{Z_i}{\pi_i} x_i \approx \sum_{i \in \mathcal{I}} x_i \qquad since \quad \mathbb{E}\frac{Z_i}{\pi_i} = 1.$$

Technique: Probability proportional to size (PPS) sampling

Probability proportional to size (PPS) sampling

Goal: Find probabilities $\pi_i = P(Z_i = 1)$ that give the HT-estimator low variance

A PPS sample samples has per item inclusion probabilities

$$\pi_i \propto x_i \quad or \quad \pi_i = 1$$

- π_i's scaled so the sample has expected size k_0
- Larger items are more likely to be included
- Non-zero contributions to the HT estimator have constant value: $\frac{x_i}{\pi_i} = c$
  - No variability from values in the sum, only from the number of values K

$$Var(S) = Var\ E(S|K) + \ E\ \cancel{Var(S|K)}$$

Sketches for sampling

Challenge:

- Inclusion probabilities π_i depend on unknown data x_i
- Inclusion probabilities π_i are not fixed, change with the stream length
- To ensure bounded size, items cannot be drawn independently
  - True inclusion probabilities are intractable to compute

Sampling sketches give ways to

- Draw PPS-like samples without replacement in bounded space
- Circumvent computing true inclusion probabilities
- Algorithms:
  - Priority sampling
  - VarOpt
- Combine multiple weightings: Multi-objective sampling (Cohen 2015)

Importance sampling

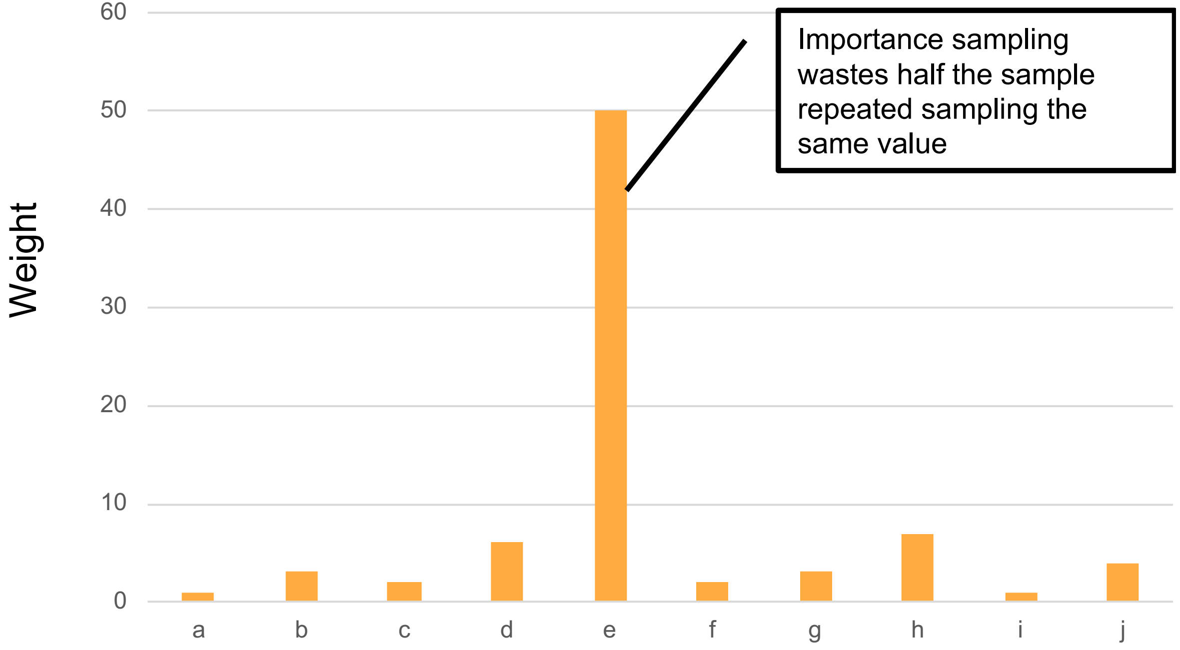
A way to estimate sums using samples drawn ***with replacement***.

- Given weights x_i
- Draw i^{th} point with probability $\pi_i \propto x_i$ but do so with replacement
- π_i's rescaled to sum to 1

Disadvantage: Duplicate items

Importance sampling

A way to estimate sums using samples drawn ***with replacement***.



Importance sampling vs HT + PPS

- HT + PPS is good when you
  - Have a fixed data set or
  - Have a discrete distribution

- Importance sampling is good when you
  - Perform Monte-Carlo estimation for
  - Continuous distributions
  - Or need to prove something where independence makes the proof easier

Break

Quantiles

Example problems

- Compute quality of service metrics
    - 99th percentile latency / battery life / etc.
- Robust metrics (median, interquartile range)
- Set anomaly detection thresholds

Usefulness of quantile sketches

- Saves an expensive operation: sorting a large array
- Extremely accurate
  - Errors $O\left(\frac{1}{\epsilon}\right)$ versus $O\left(\frac{1}{\epsilon^2}\right)$ for sampling based approaches.
- Example:
  - AB test / Confidence interval for a 90% quantile on 10 M data points requires accurate computation of 90.02% and 89.98% quantiles.

$$\epsilon \approx 0.0001$$

  - Sampling: $\frac{1}{\epsilon^2} = 100M$
  - Quantile sketch: $\frac{1}{\epsilon} = 10K$

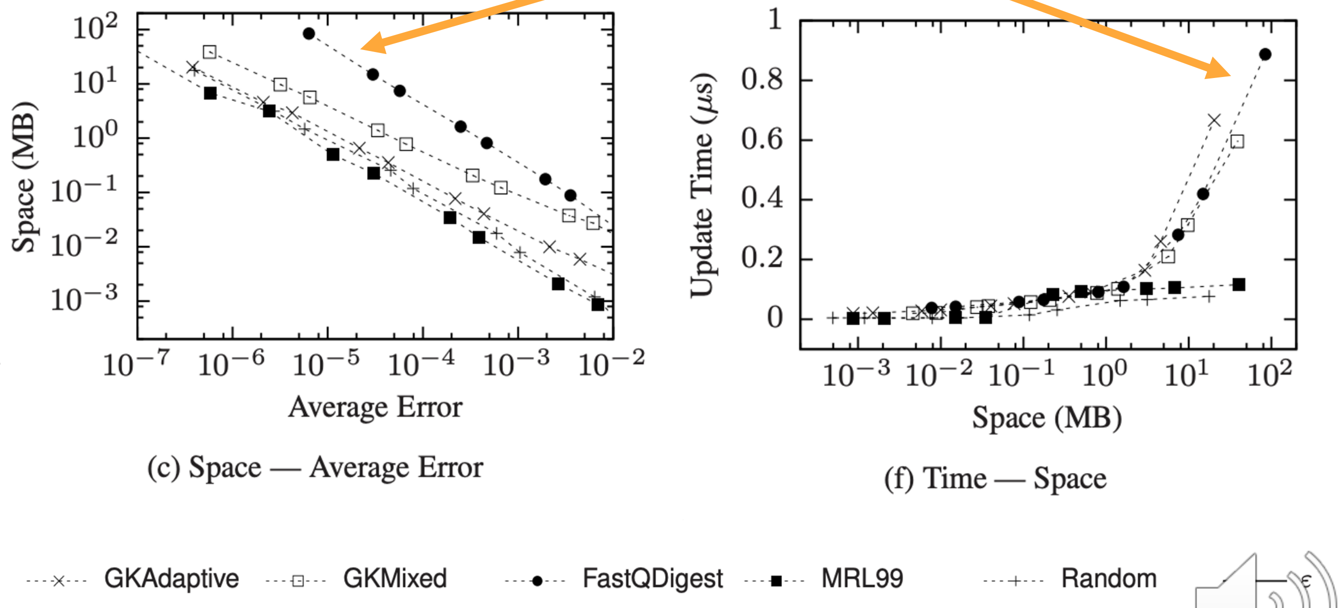
Many Quantile sketches, many implementations

Partial list of quantile sketches	Systems
Greenwald-Khanna (2001)	Spark
MRL / RANDOM (Manku et al 1999, Agarwal et al 2012)	
Q-digest (Shrivastava 2004)	Presto
T-digest (v1 Dunning 2013, v2 + Ertl 2019)	Dynatrace, Splunk, various
KLL (Karnin et al, 2016)	Apache Datasketches
DDSketch (Masson et al 2019)	Datadog
Moments sketch (Gan et a 2018)	Druid extension
Relative error streaming quantiles (Arxiv 2020)	

Navigating quantile sketches

Major, real world databases make suboptimal choices

Presto uses Q-digest



Quantiles over Data Streams: Experimental Comparisons, New Analyses, and Further Improvements
Luo, Wang, Yi, Cormode. VLDB 2016

Navigating quantile sketches

Challenges

- Theoretical space complexity hides constant factors
- Incomplete algorithm specifications
  - Q-digest has one of the best theoretical guarantees but
    - Original paper did not say much about fast implementations
    - Bad constant factors
- Heuristic approaches can work fairly well in practice
- Not all guarantees are comparable
  - Some guarantees depend on distributional assumptions
  - If data streams are i.i.d. draws, there are more efficient algorithms
- Empirical evaluation of robustness is hard
  - Worst cases depends on order of values in the stream, not just the distribution of values
  - Not obvious how to build realistic adversarial cases

Navigating quantile sketches

Comparison based sketches

- GK, KLL, MRL, Relative error streaming quantiles
- Strong error guarantees

Fixed universe sketches

- Q-digest, Dyadic count sketch
- Strong error guarantees but worse space usage

Others:

- T-digest, DDsketch, Moments sketch
- Weak to no guarantees or often requires strong assumptions on the distribution
- Can still work pretty well in practice

Strong error guarantees

Quantile problem:

- CDF: F
- Desired q-quantile: $x_q = F^{-1}(q)$

Guarantee:

- Guarantee is on CDF: $\sup_{x \approx x_q} \left| \hat{F}(x) - F(x) \right| < \epsilon$
- May ask for q-quantile but get $(q - \epsilon)$-quantile: $\hat{x}_q \in \left(x_{q-\epsilon}, x_{q+\epsilon} \right)$
- Not on the raw values: not $\hat{x}_q = x_q \pm \epsilon$

- No assumptions on input stream
- Probabilistic guarantees hold with probability $1 - \delta$
- Precise error bounds can be computed

Example weak guarantee: DDSketch

- DDSketch is essentially a histogram on log scale values
  - Cares about only one tail of the distribution
  - Keeps equal width bins on that side of distribution and collapses bins on the other side

- Guarantee: $\hat{x}_q = x_q(1 \pm \epsilon)$ if $x_1 \leq x_q \gamma^{m-1}$
  - m is the size of the sketch
  - Histogram bin widths are defined by $\gamma = \frac{(1+\epsilon)}{1-\epsilon}$

- Assumptions in guarantee:
  - Values are from a bounded interval: (0,c)
  - True quantile is within m bins of the max

Example weak guarantee: Moment sketch

- Assumptions:
  - Data is on bounded interval: (-c, c)
  - Has bounded density f

- Guarantee: $\|\hat{F} - F\| = O\left(\frac{f_{max}}{m}\right)$ where m = # stored moments

- Not for a specific, desired quantile
- Has distributional assumptions
- Only a rate, cannot provide error bound

Comparing sketches

Sketch	Guarantee	Space	Mergeable with space guarantee	Optimal space complexity
KLL (with GK)	Probabilistic	$O\left(\frac{1}{\epsilon}\log\log\delta^{-1}\right)$	No	Optimal randomized
KLL	Probabilistic	$O\left(\frac{1}{\epsilon}\log^2\log\delta^{-1}\right)$	Yes	
MRL	Deterministic	$O\left(\frac{1}{\epsilon}\log^2\epsilon n\right)$	Yes	
Greenwald-Khanna	Deterministic	$O\left(\frac{1}{\epsilon}\log\ \epsilon n\right)$	No	Optimal deterministic

Comparing sketches

Sketch	Guarantee	Space	Mergeable with space guarantee	Optimal space complexity
KLL (with GK)	Probabilistic	$O\left(\frac{1}{\epsilon}\log\log\delta^{-1}\right)$	No	Optimal randomized
KLL	Probabilistic	$O\left(\frac{1}{\epsilon}\log^2\log\delta^{-1}\right)$	Yes	
MRL	Deterministic	$O\left(\frac{1}{\epsilon}\log^2\epsilon n\right)$	Yes	
Greenwald-Khanna	Deterministic	$O\left(\frac{1}{\epsilon}\log\ \epsilon n\right)$	No	Optimal deterministic

Superiority of randomized sketches

Sketch	Guarantee	Space	Me sp gu	ace
KLL (with GK)	Probabilistic	$O\left(\frac{1}{\epsilon}\log\log\delta^{-1}\right)$	No	Optimal randomized
KLL	Probabilistic	$O\left(\frac{1}{\epsilon}\log^2\log\delta^{-1}\right)$	Yes	
MRL	Deterministic	$O\left(\frac{1}{\epsilon}\log^2\epsilon n\right)$	Yes	
Greenwald-Khanna	Deterministic	$O\left(\frac{1}{\epsilon}\log\ \epsilon n\right)$	No	Optimal deterministic

Probabilistic guarantees have weak dependence on failure probability

Deterministic guarantees depend on the stream length n

Compactors and compactor hierarchies

How randomized quantile sketches work

- Built by stacking “compactors” of fixed size
- Each compactor is a “better than uniform” sample
  - Sort the items in the compactor
  - Randomly select either the even indices or the odd indices and double the selected items’ weight
- If I ask how many are $\leq x$ where x is even, the compactor will return an exact answer while uniform sampling will return a random one

1	2	3	4	.	.	.	.	.	98	99	100

New developments: Improved tail quantiles

Problem: Often only tail quantiles are of interest (e.g. 99th percentiles)

- Heuristic targeting of tail quantiles
  - DDSketch
  - T-digest
- Theoretically sound, but not practical
  - Guarantees error $\epsilon q \ \ or \ \ \epsilon(1-q) \ \ vs \ \ \epsilon$ for regular quantile sketches
  - KLL sketch with error ϵq would be smaller for practical values of q
  - Zhang and Wang 2007
  - Q-digest variation: Cormode, Korn, Muthukrishnan, Srivastava 2006
- New:
  - Relative Error Streaming Quantiles (Cormode, Karnin, Liberty, Thaler, Vesely 2020)

Choosing a sketch

- KLL is clearly the best sketch if
  - Theoretical guarantees are needed for all inputs
  - Providing randomized guarantees are acceptable
  - Good practical performance is needed
- Heuristic approaches can still work
  - T-digest v2
  - DDsketch
- Recent developments may be better for tail quantiles

Top-k / Frequent Items / Heavy hitters

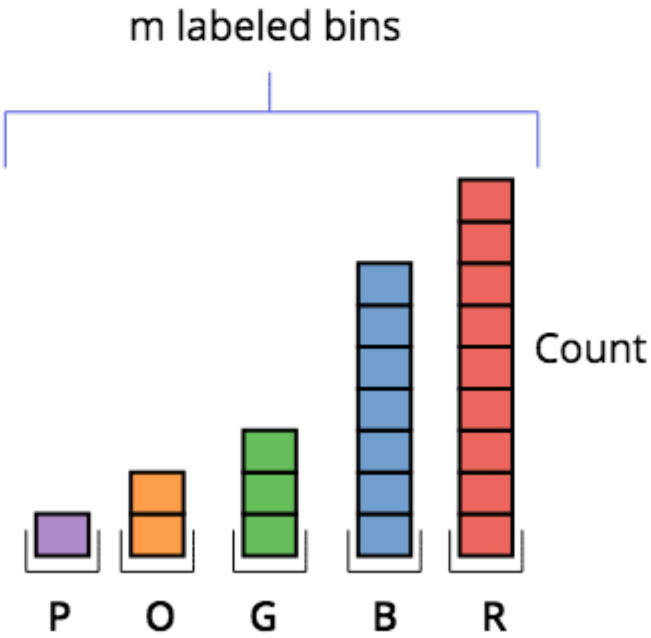
Problems

Examples:

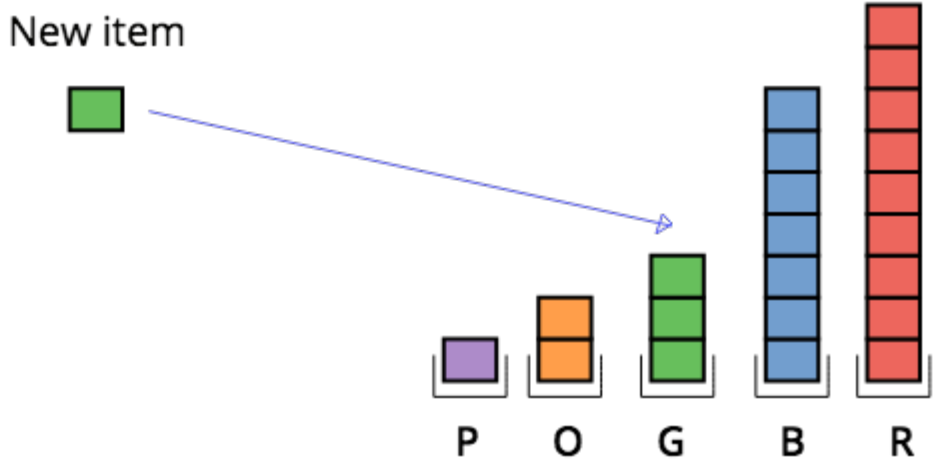
- Monitoring: Detect a DDoS attack on a destination
- Analytics: Find the top selling products or heaviest users
- Recommendations: What are the trending topics?

Problem: Given a stream of (key, increment) pairs, find the keys with the largest total sums

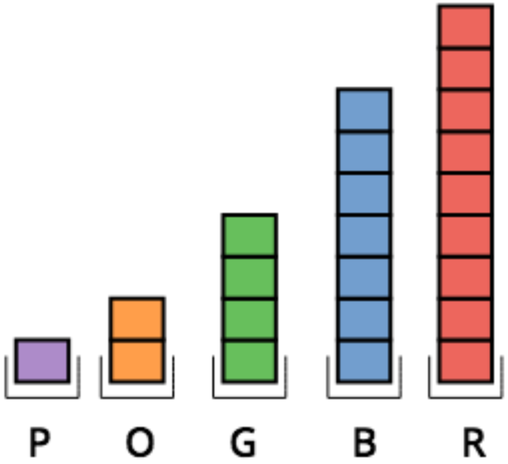
Space Saving



Space Saving

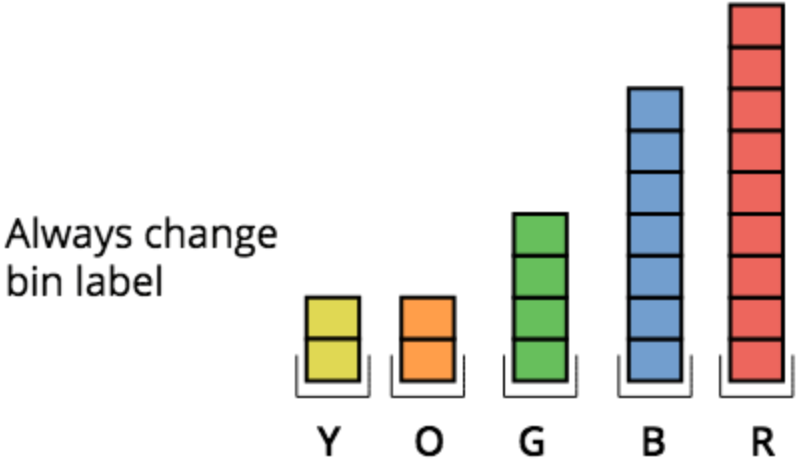


Space Saving



Space Saving

Space Saving



Properties

- Simple definition:
  - For a sketch of size m, an item is a heavy hitter if it appears > n/m times.

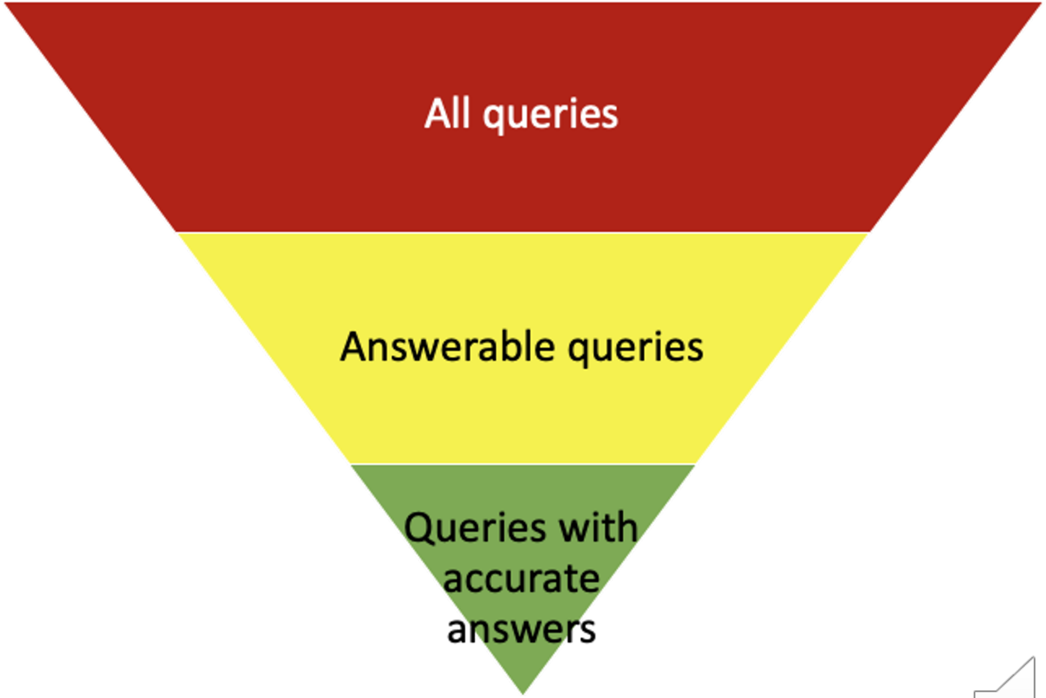
- Deterministic guarantee:
  - Every heavy hitter is included in the sketch

- Space-Saving / Misra-Gries and its variants are the current ”best” heavy hitter sketch
  - When there exist heavy hitters and
  - Only heavy hitters are of interest

Flexibility vs efficiency

Sketches reduce space by restricting

- the queries that it can answer and
- hence, the amount of information it needs to store



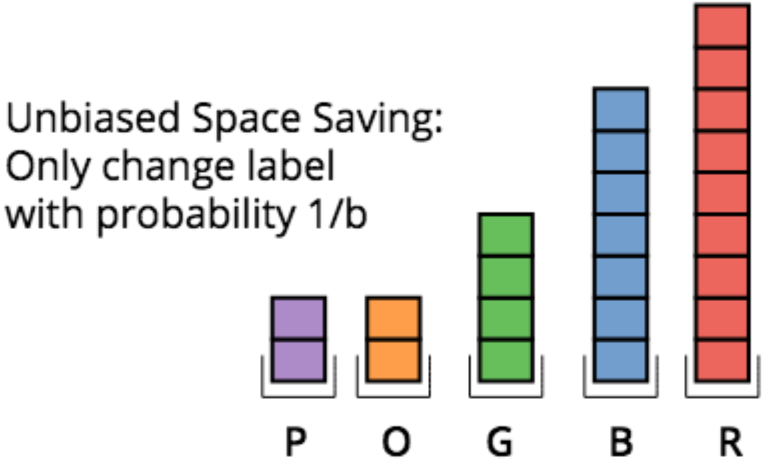
Power of randomization

- For any question other than what the heavy hitters are or there are no heavy hitters, then the sketch is useless!

- With a bit of randomization, you can extend the functionality and compute subset sums.

Unbiased Space Saving

Unbiased Space Saving



Properties

- Given an i.i.d. input stream,
  - The heavy hitters are recovered almost surely
  - The sketch asymptotically generates a PPS sample
    - with size proportional to each item’s total count
    - but without knowing the item counts a priori.
    - Good for event streams

- For non-i.i.d. streams
  - The sketch still generates a sample with unbiased weights

Takeaway: Consider more flexible sketches

- Can be better to choose one sketch to solve two problems than two sketches that are the best for their specific problems

Linear sketches

Linear sketch

- Linear transformation of the data
  - $S = M X$
  - M does not depend on X
- Automatically supports
  - Merging: $S_{new} = M(X_1 + X_2) = S_1 + S_2$
  - Deletions: $S_{new} = M(X_1 - X_2) = S_1 - S_2$

Counting sketches

Data stream of key, value pairs: $(k_1, x_1), (k_2, x_2), \ldots, (k_t, x_t)$

Goal: Compute sum grouped by key $v_\kappa = \sum_{k_i=\kappa} x_i \quad \forall keys\ \kappa$

Example

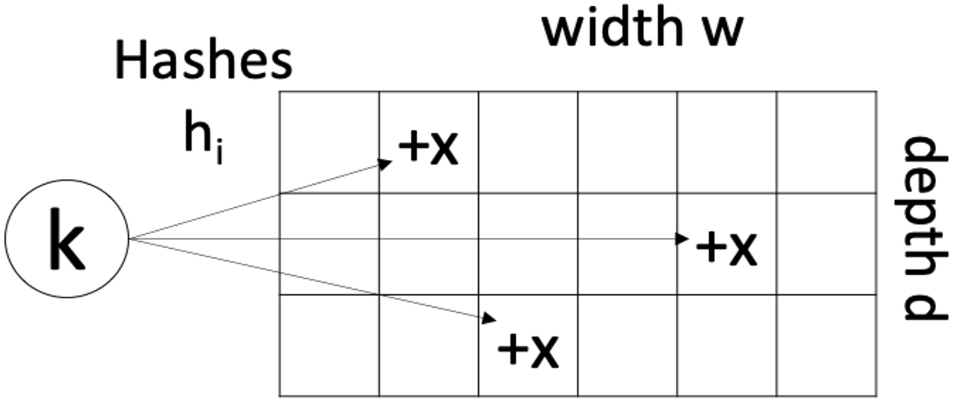
- Given an event stream of delivered ad ids,
- Compute the total number of impressions per ad
- Can be used as a component for other sketches that count
    - Quantile sketches
    - Heavy hitter sketches

CountMin

Sketch: d x w array of counters

Given a pair (k,x) from the stream, each key hashes to d counters and increments them by x

Sketch is a linear mapping of total count vector $\vec{v}$



Technique: Success Amplification and concentration inequalities

Replication

Each counter for key k contains

- True sum v_k
- IID error ϵ_i due to hash collisions
- Independent upper bound on v_k that can be used as an estimate

	$v_k+\varepsilon_1$				
				$v_k+\varepsilon_2$	
		$v_k+\varepsilon_3$			

CountMin estimation

Estimate total sum for key k by

$$\hat{V}_k = \min_i \{v_k + \epsilon_i\} = v_k + \min_i \{\epsilon_i\}$$

	$v_k+\varepsilon_1$				
				$v_k+\varepsilon_2$	
		$v_k+\varepsilon_3$			

Success amplification

- One estimate has failure probability

$$P(\epsilon_i > c) = \rho$$

- Minimum of d estimates has exponentially smaller failure probability

$$P\left(\min_{i=1,\dots,d} \epsilon_i > \mathrm{c}\right) = \rho^d$$

- Many analyses for sketching are based on similar probabilistic inequalities (Markov’s, Azuma-Hoeffding, Bernstein, etc)

Technique: Optimal statistical estimation

Advantages of statistical estimation techniques

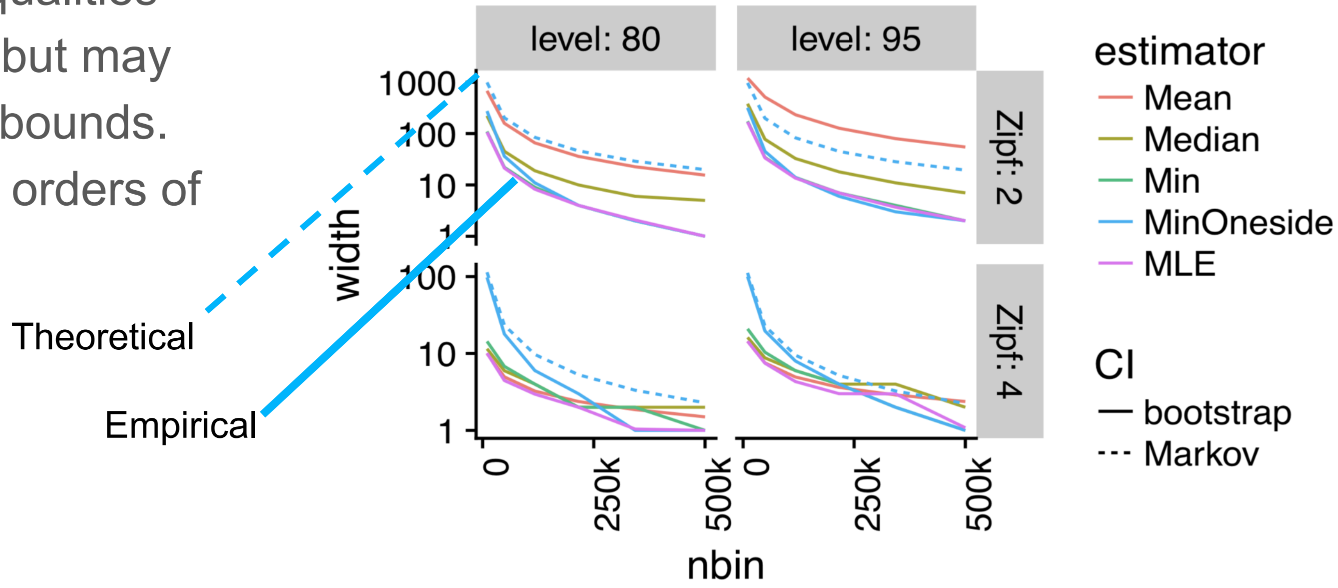
Statistical techniques can provide

- General techniques that can work on a variety of problems e.g. MLE
- Which often automatically have good properties
  - Asymptotic efficiency / Minimum variance estimators
  - Consistency / unbiasedness
  - Tight error estimates
- Results are useful for finite sample behavior
  - Space complexity ignores large leading constants
  - Asymptotics often are interested in that leading constant and only ignore lower over terms

New developments:
Connecting theory and practice

Concentration inequalities prove correctness but may not provide useful bounds.

- Bounds can be orders of magnitude off



Improved error and estimation

CountMin sketch entries all contain identically distributed error.

- True model: $S_{\mathrm{i}} = v_i + \epsilon_{ij}$
- Basically known error distribution $\epsilon_{ij} \sim F$ since there a many replicates.

⇒ Statistical techniques yield improved estimates and tight error bounds

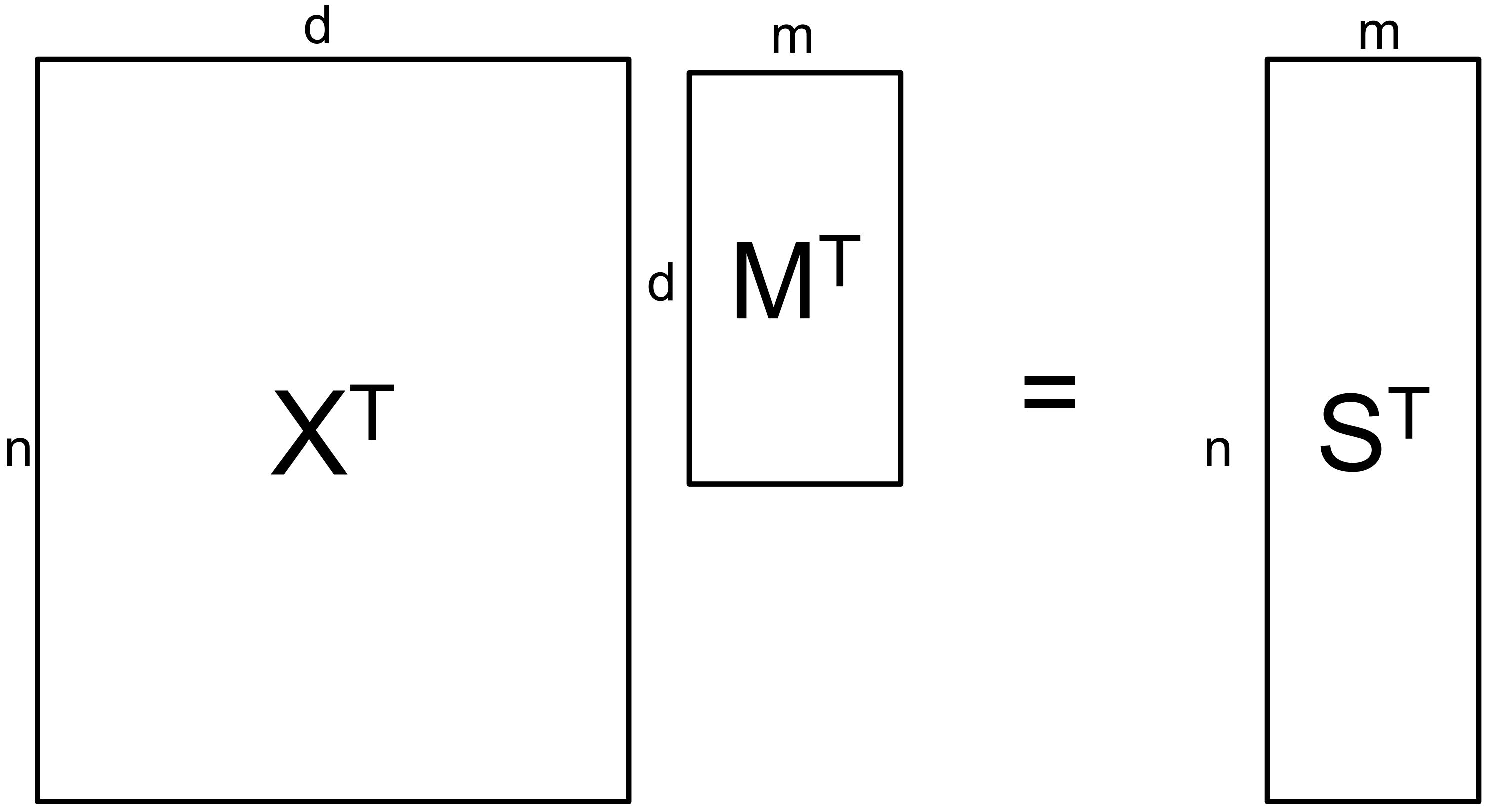
ε_{11}	$v_k+\varepsilon_{12}$		…		ε_{16}
ε_{21}	ε_{22}		…	$v_k+\varepsilon_{25}$	ε_{26}
ε_{31}	ε_{32}	$v_k+\varepsilon_{33}$	…		ε_{36}

Implementing sketches

- Often requires
  - Choosing which sketch to use
  - How to size the sketch and choose parameters to achieve a desired error
    - Time consuming to do empirically, can't always be fully trusted
    - Preferable to use theory if it returns tight error bounds
    - Tight error bounds makes it feasible to optimize for the best sketch parameters
- Ting (2019). Count-Min: Optimal Estimation and Tight Error Bounds using Empirical Error Distributions. KDD

Technique:
Random Projections

Random projection



Random projection

- $S(X) = \ M\ X$
- Take M to be matrix with random entries such that
$$\mathbb{E}\ M^T M = I$$
- Then for an $n' \times d$ matrix V,
$$\mathrm{V}^{\mathrm{T}}\mathrm{X} \approx S^T(V)S(X)$$
- Typically $M_{ij} \sim Normal\left(0, \frac{1}{\mathrm{m}}\right)$

Johnson-Lindestrauss Theorem

- Given a $d \times n$ data set X of n points
- The JLT gives that a Gaussian random projection simultaneously preserves all pairwise distances

$$(1-\epsilon)\left\|X_i - X_j\right\|^2 \leq \left\|S_i - S_j\right\|^2 \leq (1+\epsilon)\left\|X_i - X_j\right\|^2$$

- JLT also implies inner products are preserved
- Condition: need $m = \Omega(\epsilon^{-2} \log d)$ dimensions in the random projection

Applications

- Other counting sketches: AGMS / Count
    - Join size estimation for query optimization
    - Count based features in ML models
        - Historical click through rates
- Dimensionality reduction
- Fast, iterative numerical linear algebra solvers

Advanced methods

Sampling for Statistical / ML models

Ways to generate weights for data point

- Leverage score sampling
- Local Case-control sampling
- Influence based
  - Many estimators that are loss minimizers / likelihood maximizers (M-estimators) are theoretically analyzed as a sum

$$\hat{\theta} = \theta + \frac{1}{n}\sum_i \psi_\theta(X_i) + o_p\left(\frac{1}{\sqrt{n}}\right)$$

  - $\psi_\theta(\cdot)$ is called the influence function
  - Contributions to error are a sum over influences $\psi_\theta(X_i)$ $\Rightarrow$ Use PPS sampling
  - Asymptotically optimal if you can compute influence exactly

Matrix approximations

- Frequent directions (Liberty 2013)
  - Approximate SVD: Find top right singular vectors in a stream
  - Bears resemblance to Misra-Gries

- Nystrom approximation
  - For approximating low-rank kernel matrices
  - Used for Gaussian processes, Kernel methods, spectral graph based methods

Coresets

Any set of weighted points $\mathrm{C} = (\mathrm{W}, \tilde{X})$ which can be used in place of the original data X to obtain an accurate approximation.

- Typically satisfies: $|cost(C, \theta) - cost(X, \theta)| \leq \epsilon \cdot cost(X, \theta)$
- More general than sampling as it can be generated through optimization or sampling
- Example uses: SVM, Bayesian methods, clustering

Graph problems

- Graphs pose unique challenges since
  - They can easy grow very large
  - Node / edges cannot be picked independently
- Example problems:
  - Minimum spanning tree
  - Maximum weight matchings
  - Graph sparsification

Privacy and sketching

Uses

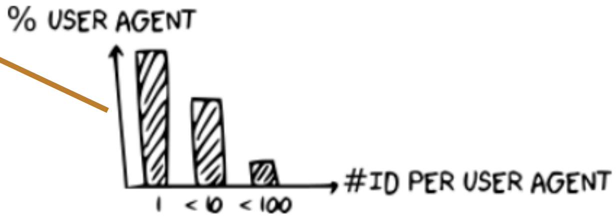
- Identifying privacy risks

- In privacy preserving data collection

Identifying privacy risks

- KHyperLogLog
- Example
    - Sensitive data set with user ids removed but contains User Agent (UA) strings
    - If a UA string is unique, then it may be joined to a (non-sensitive) data set with UA string
        - ⇒ Potential privacy violation

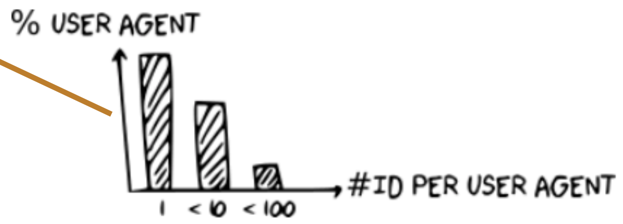
Many UA strings are associated to a unique ID => privacy violation



KHLL

- Very simple sketch that composes a sampling based method with a distinct counting sketch
  - Use a k-minimum values sketch to sample a set of User Agent strings
  - Use an HLL sketch to estimate the number of distinct users associated with these sampled UA strings.
- Produces this histogram identifying privacy risks

Many UA strings are associated to a unique ID => privacy violation



Differential privacy

- Definition: An algorithm $\mathcal{A}$ is ϵ-differentially private if for any datasets D_1, D_2 differing in only one data point and for any set S,

$$\frac{P(\mathcal{A}(D_1) \in S)}{P(\mathcal{A}(D_2) \in S)} \leq \exp(\epsilon)$$

- Colloquially, even if you knew everything about everyone in the database except the single person you want to snoop on, you would at best be $\exp(\epsilon) - 1 \approx \epsilon$ more sure about that person's true value.

- Advantage of DP is that it provides provable privacy guarantees

Relevance of sketches

Data: (key, value) pairs

- Reduce data transferred and costs
  - Differential privacy mechanisms rely on injecting noise
  - Sparse vectors of (key, value) pairs become not sparse since unobserved keys get noise too
  - E.g. website visits
  - Increased communication costs
- Reduces added noise since there are fewer entries to add noise to
- Privatized or changing domains (keys)
- Sketches themselves can introduce some noise-like behavior

Privacy in use

- Google developed the KHLL sketch
- Apple uses a Private Count Mean Sketch
    - Other data sketching techniques (Fast Hadamard Transform) are used to further reduce the communication costs from 1 row in a sketch down to 1 bit.

Summary

- Sketches
  - Distinct counting, Quantiles, Frequent Items, Sampling
  - Linear sketches and random projections
  - “Advanced” sketches
- How they work
  - Techniques for constructing sketches and obtaining estimates
  - Statistical perspective. Sketch = efficiently encoded random process
  - Use of statistical techniques to improve sketches
- Principles
  - Trading off flexibility with efficiency
  - All queries vs. Inserting queries vs. Interesting answers
  - Constants matter!
  - Reduction of complex problems to simple sketches
- Guarantees
- Privacy and sketches

Some resources

- Book: Small Summaries for Big Data. Cormode and Yi (2020)
  - http://dimacs.rutgers.edu/~graham/ssbd.html

- Documentation and sketches at https://datasketches.apache.org/
- Problems at https://sublinear.info/
- Edith Cohen's papers organized by topic at http://www.cohenwang.com/edith/publications.html
- Past tutorials on sampling and sketching
  - KDD 2014: Cormode and Duffield: https://nickduffield.net/download/papers/Tutorial_KDD_2014.pdf

- Book: *Sketching as a Tool for Numerical Linear Algebra*. Woodruff (2014)