Summarizing and Mining *Skewed* Data Streams

Graham Cormode
cormode@bell-labs.com

Flip Korn, S. Muthukrishnan, Divesh Srivastava

Data Streams

Many large sources of data are generated as streams of updates:

- IP Network traffic data
- Text: email/IM/SMS/weblogs
- Scientific/monitoring data

Must analyze this data which is high speed (tens of thousands to millions of updates/second) and massive (gigabytes to terabytes per day)

Data Stream Analysis

Analysis of data streams consists of two parts:

- Summarization
  - Fast memory is much smaller than data size, so need a (guaranteed) concise synopsis
  - Data is distributed, so need to combine synopses
- Mining
  - Extract information about streams from synopsis
  - Examples: Heavy hitters/frequent items, quantiles, changes/difference, clustering/trending, etc.

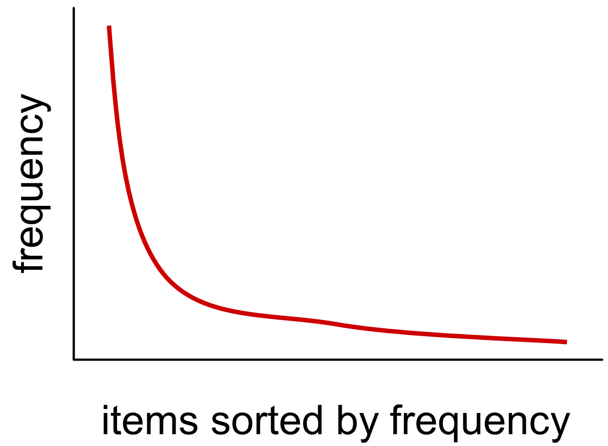
Skew In Data

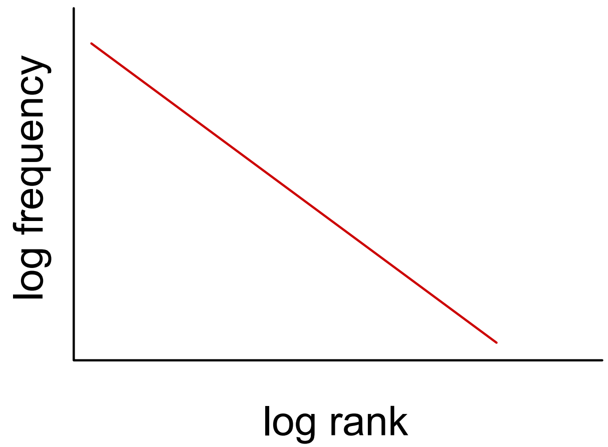
Data is rarely uniform in practice, typically skewed

A few items are frequent, then a long tail of infrequent items

Such skew is prevalent in network data, word frequency, paper citations, city sizes, etc.

One concept, many names: Zipf distribution, Pareto distribution, Power-laws, multifractals, etc.





Outline

- Better bounds for summarization/mining tasks by incorporating skewness into analysis
  - **Count-Min sketch and Zipf distribution**
- New mining tasks motivated by skewness in data
  - Biased Quantiles

Zipf Distribution (Pareto)

Items drawn from a universe of size U

Draw N items, frequency of i'th most frequent is

$$f_i \approx N i^{-z}$$

Proportionality constant depends on U, z, not N

z indicates skewness:

- z =0: Uniform distribution
- z < 0.5: light skew/no skew
- $0.5 \leq z < 1$: moderate skew
- $1 \leq z$: (highly) skewed

most real data in this range

Typical Skews

Data Source	Zipf skewness, z
Web page popularity	0.7 — 0.8
FTP Transmission size	0.9 — 1.1
Word use in English text	1.1 — 1.3
Depth of website exploration	1.4 — 1.6

Our contributions

A simple synopsis used to approximately answer:

- Point queries (PQ) — given item i, return how many times i occurred in the stream, f_i
- Second Frequency moment (F_2) — compute sum of squares of frequencies of all items

The basis of many mining tasks: histograms, anomaly detection, quantiles, heavy hitters

Asymptotic improvement over prior methods: for error bound ε, space is $o(1/\varepsilon)$ for $z>1$ previously, cost was $O(1/\varepsilon^2)$ for F_2, $O(1/\varepsilon)$ for PQ

Point Estimation

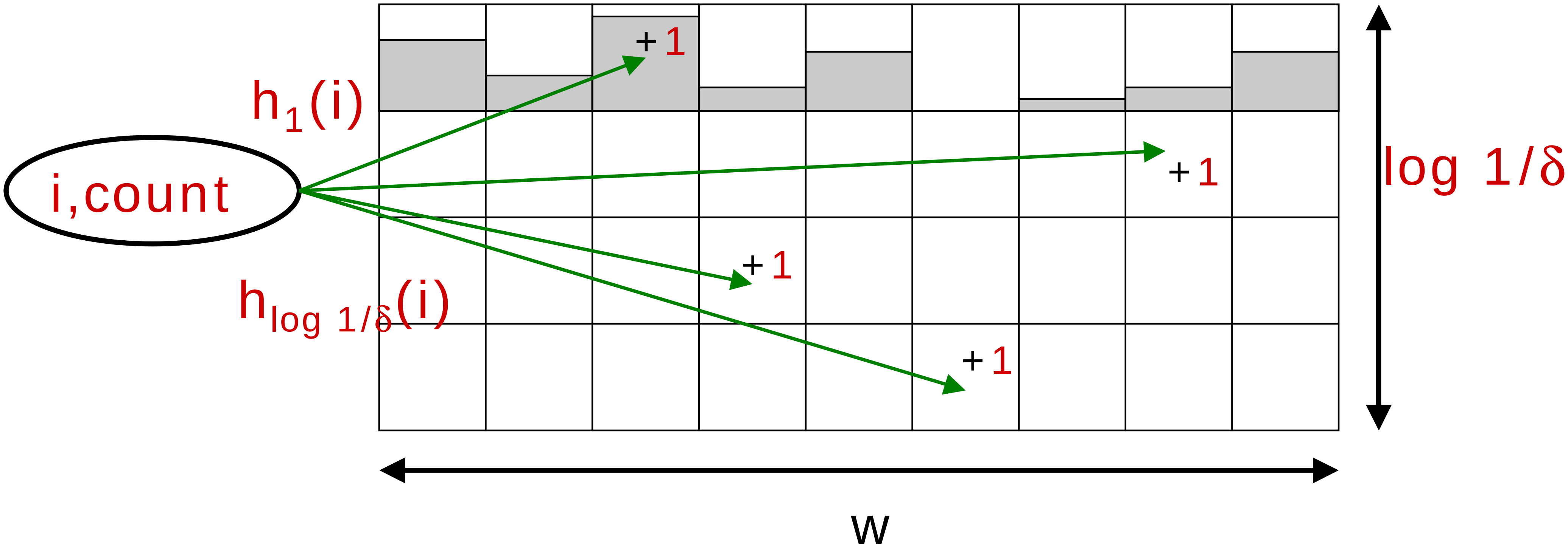
Use the Count-Min Sketch structure, introduced in [CM04] to answer point queries with error $<\varepsilon N$ with probability at least $1-\delta$

Tighter analysis here for skewed data, plus new analysis for F_2.

Ingredients:

- Universal hash fns
  $h_1..h_{\log 1/\delta}$ {items}→ {1..w}
- Array of counters CM[1..w, 1..log 1/δ]

Update Algorithm



Count-Min Sketch

Analysis for Point Queries

Split error into:

- Collisions with w/3 largest items
- Collisions with the remaining items

With constant probability (2/3), no large items collide with the queried point.

Expected error $= \frac{1}{w} \sum_{x=w/3+1}^{U} f_x \leq \frac{N}{w} (\frac{w}{3})^{1-z} \leq \frac{\varepsilon N}{3}$

Applying Zipf tail bounds and setting $w = 3\varepsilon^{-1/z}$.

Markov Inequality: $\Pr[\text{error} > \varepsilon N] < 1/3$.

Take Min of estimates: $\Pr[\text{error} > \varepsilon N] < 3^{-\log 1/\delta} < \delta$

Application to top-k items

Can find f_i with $(1\pm\varepsilon)$ relative error for i<k (ie, the top-k most frequent items).

Applying similar analysis and tail bounds gives:

$$\frac{Nk^{1-z}}{w} = \frac{\varepsilon N k^{-z}}{2}$$

and so $w = O(k/\varepsilon)$ for any $z>1$.

Improves the $O(k/\varepsilon^2)$ bound due to [CCFC02]

We only require $z>1$, do not need value of z.

Second Frequency Moment

Second Frequency Moment, $F_2 = \sum_i f_i^2$

Two techniques to make estimate from CM sketch:

- CM^+: $\min_j \sum_{k=1}^{w} CM[j,k]^2$
  — min of F_2 of rows in sketch
- CM^-: $\text{median}_j \sum_{k=1}^{w/2} (CM[j,2k] - CM[j,2k-1])^2$
  — median of F_2 of differences of adjacent entries in the sketch

We compare bounds for both methods.

CM+ Analysis

With constant probability, the largest $w^{1/2}$ items all fall in different buckets. For z>1:

$$
\begin{aligned}
\mathsf{E}(X_j) &= \sum_{i=1}^{U} f_i^2 + \sum_{i=1}^{U} \sum_{j=1, j\neq i}^{U} f_i f_j \Pr[h(i)=h(j)] - F_2 \\
&\leq F_2 + \frac{1}{w}\left(\sum_{i=1}^{U} \sum_{j=1, j\neq i}^{U} f_i f_j - \sum_{i=1}^{m} \sum_{j=1, j\neq i}^{m} f_i f_j\right) - F_2 \\
&\leq \frac{1}{w}\left(2 \sum_{i=1}^{m} f_i \sum_{j=m+1}^{U} f_j + \left(\sum_{i=m+1}^{U} f_i\right)^2\right) \\
&\leq \frac{2}{w}\left(\sum_{i=1}^{U} f_i \sum_{j=m+1}^{U} f_j\right) \leq \frac{2N^2 c_z m^{1-z}}{w(z-1)} \leq \frac{2F_2 c_z (2z-1)}{c_z^2 (z-1)} w^{\frac{-(1+z)}{2}}
\end{aligned}
$$

CM^+ Analysis

Simplifying, we set the expected error = $\frac{1}{2}\varepsilon F_2$.

This gives $w = O(\varepsilon^{-2/(1+z)})$.

Applying Markov inequality shows error is at most εF_2 with constant probability.

Taking the minimum of the log $1/\delta$ repetitions reduces failure probability to δ.

Total space cost = $O(\varepsilon^{-2/(1+z)} \log 1/\delta)$, provided $z>1$

CM$^-$ Analysis

For $z>1/2$, again constant probability that the largest $w^{1/2}$ items all fall in different buckets.

We show that:

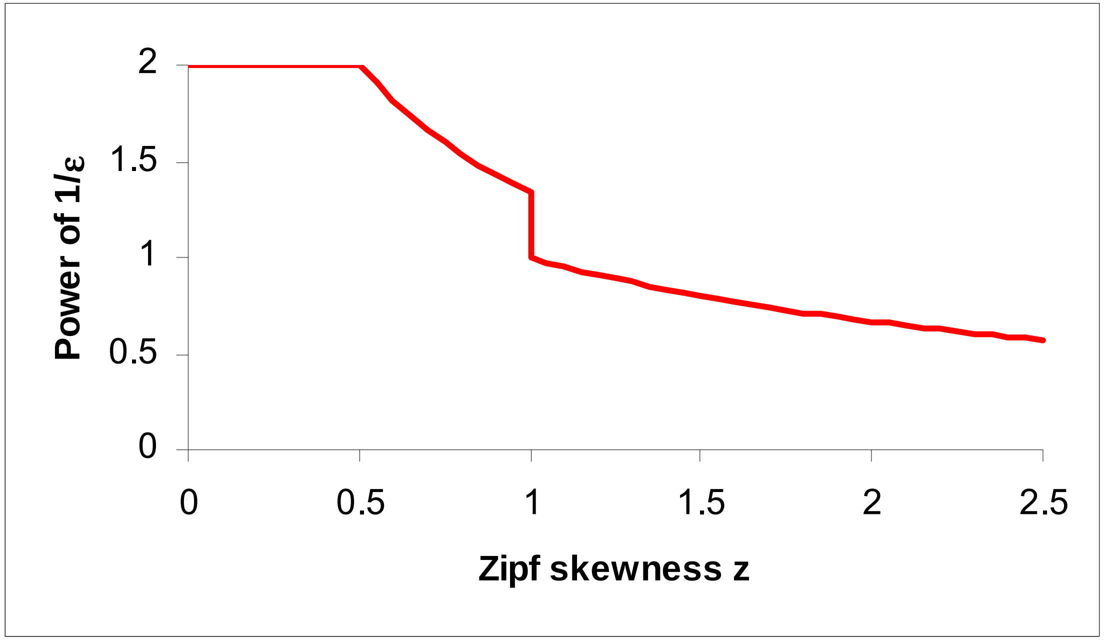
- Expectation of each CM$^-$ estimate is F_2
- Variance $\leq 8F_2^2\, w^{-(1-2z)/2}$

Setting Var $= \varepsilon^2 F_2^2$ and applying Chebyshev bound gives constant probability of $< \varepsilon F_2$ error.

Taking the median amplifies this to δ probability

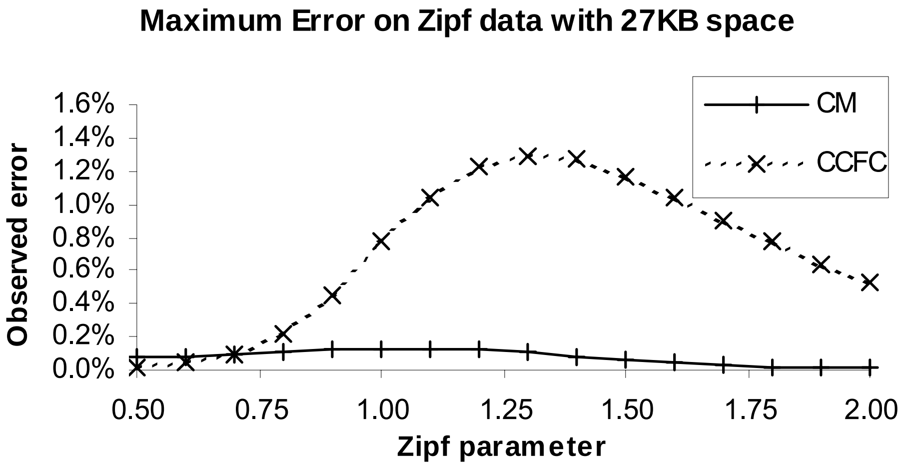
Total cost space $= O(\varepsilon^{-4/(1+2z)} \log 1/\delta)$, if $z>½$

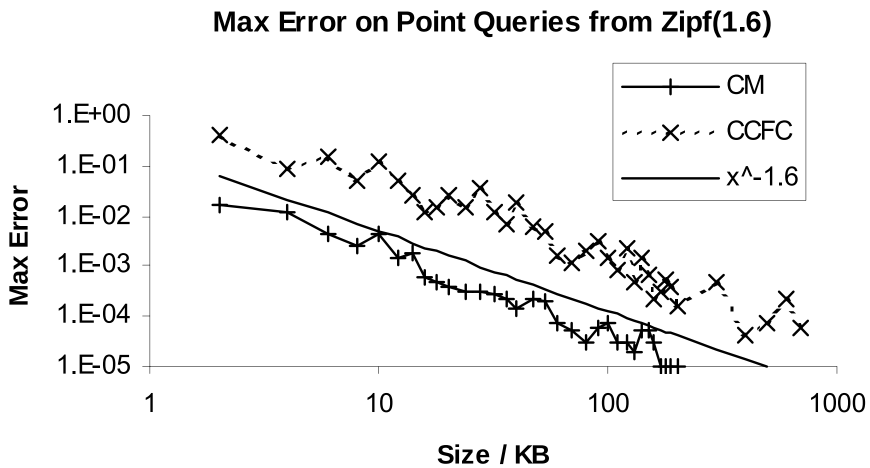
F_2 Estimation Summary



Skewness	Space Cost	Method
$z \leq 1/2$	$(1/\varepsilon)^2$	CM^-
$1/2 < z \leq 1$	$(1/\varepsilon)^{4/(1+2z)}$	CM^-
$1 < z$	$(1/\varepsilon)^{2/1+z}$	CM^+

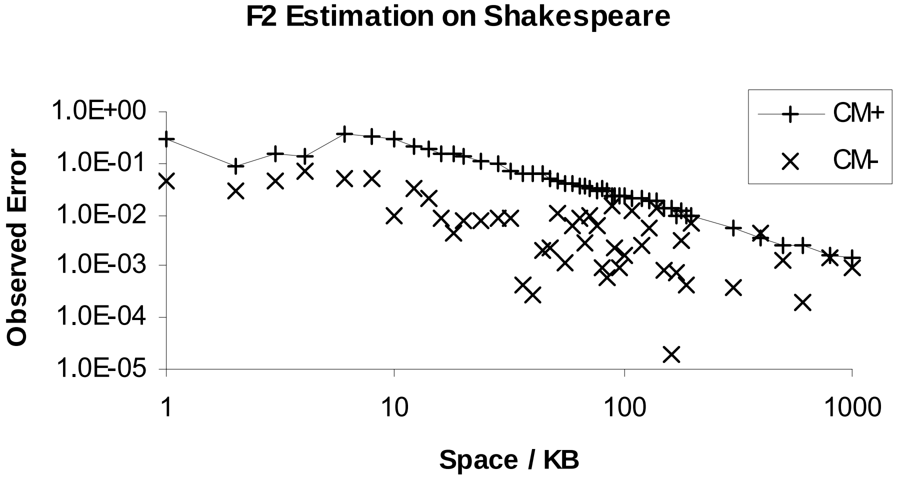
Experiments: Point Queries

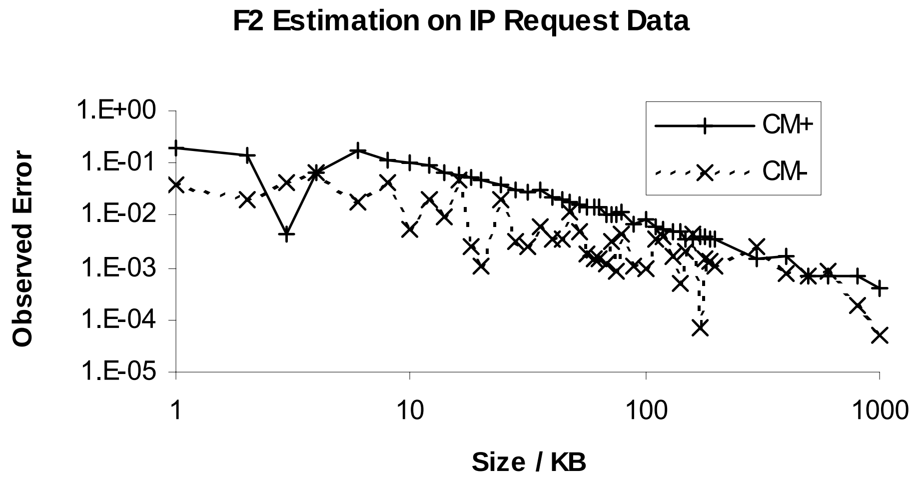




- On synthetic data, significantly outperforms worst error from comparable method [CCFC02]
- Error decays as space increases, as predicted

Experiments: F_2 Estimation





- Experiments on complete works of Shakespeare (5MB, $z \approx 1.2$) and IP traffic data (20MB, $z \approx 1.3$)
- CM^- seems to do better in practice on real data.

Experiments: Timing

Easily process 2-3million new items / second on standard desktop PC.

Queries are also fast

- point queries ≈ 1μs
- F_2 queries ≈ 100μs

Alternative methods are at least 40-50% slower.

Outline

- Better bounds for summarization/mining tasks by incorporating skewness into analysis
  – Count-Min sketch and Zipf distribution
- New mining tasks motivated by skewness in data
  – **Biased Quantiles**

Quantiles

Quantiles summarize data distribution concisely.

Given N items, the ϕ–quantile is the item with *rank* ϕN in the sorted order.

Eg. The median is the 0.5-quantile, the minimum is the 0-quantile.

Equidepth histograms put bucket boundaries on regular quantile values, eg 0.1, 0.2...0.9

Quantiles are a robust and rich summary: median is less affected by outliers than mean

Quantiles over Data Streams

Data stream consists of N items in arbitrary order.

Models many data sources eg network traffic, each packet is one item.

Requires linear space to compute quantiles exactly in one pass, $\Omega(N^{1/p})$ in p passes.

ε-approximate computation in sub-linear space

- Φ-quantile: item with rank between $(\Phi-\varepsilon)N$ and $(\Phi+\varepsilon)N$
- [GK01]: insertions only, space $O(1/\varepsilon \log(\varepsilon N))$
- [CM04]: insertions and deletions, space $O(1/\varepsilon \log 1/\delta)$

Biased Quantiles

- IP network traffic is very skewed
  - Long tails of great interest
  - Eg: 0.9, 0.95, 0.99-quantiles of TCP round trip times
- Issue: uniform error guarantees
  - $\varepsilon = 0.05$: okay for median, but not 0.99-quantile
  - $\varepsilon = 0.001$: okay for both, but needs too much space
- Goal: support *relative* error guarantees in small space
  - Low-biased quantiles: ϕ-quantiles in ranks $\phi(1\pm\varepsilon)N$
  - High-biased quantiles: $(1-\phi)$-quantiles in ranks $(1-(1\pm\varepsilon)\phi)N$

Prior Work

Sampling approach given by Gupta and Zane [GZ03] in context of a different problem:

- Keep $O(1/\varepsilon)$ samplers at different sample rates, each keeping a sample of $O(1/\varepsilon^2)$ items
- Total space: $O(1/\varepsilon^3)$, probabilistic algorithm

Uses too much space in practice.

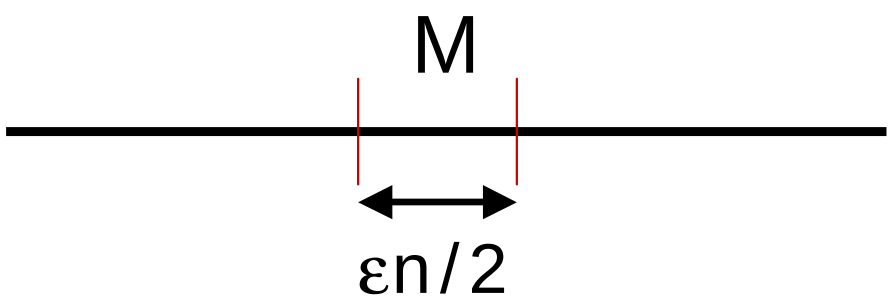
Is it possible to do better? Without randomization?

Intuition

Example shows intuition behind our approach.

Low-biased quantiles: give error $\varepsilon\phi$ on ϕ-quantiles

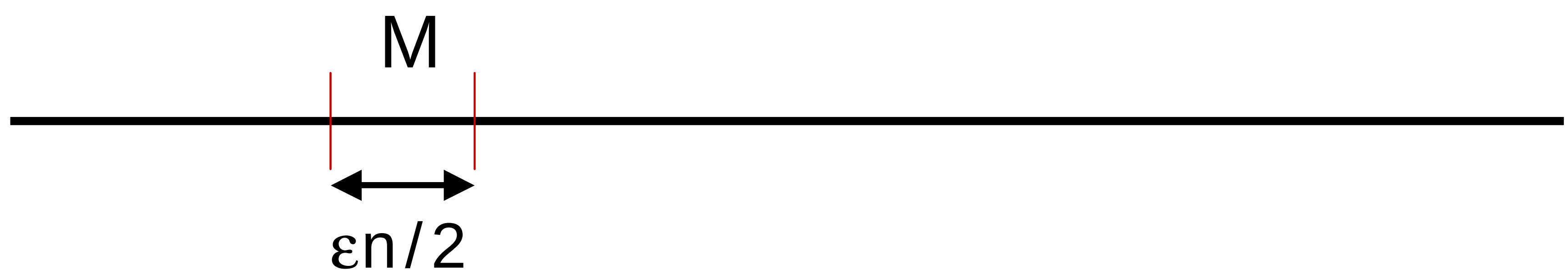
- Set ε=10%. Suppose we know approximate median of n items is M — so absolute error is $\varepsilon n/2$



- Then there are n inserts, all above M
- M is now the first quartile, so we need error $\varepsilon N/4$

Intuition

How can error bounds be maintained?



– Total number of items is now N=2n, so required absolute error bound is for M is still εn/2

Error bound never shrinks too fast, so we can hope to guarantee relative errors.

Challenge is to guarantee accuracy in small space

Space for Biased Quantiles

- Any solution to the Biased Quantiles problem must use space at least $\Omega(1/\varepsilon \log(\varepsilon N))$
- Shown by a counting argument, there are $\Omega(1/\varepsilon \log(\varepsilon N))$ possible different answers based on choice of ϕ
- For uniform quantiles, corresponding lower bound is $\Omega(1/\varepsilon)$ — biased quantiles problem is strictly harder in terms of space needed.

Our Approach

A deterministic algorithm that guarantees relative error for low-biased or high-biased quantiles

Three main routines:

- Insert(v) — inserts a new item, v
- Compress — periodically prune data structure
- Output(ϕ) — output item with rank $(1\pm\varepsilon)\phi N$

Similar structure to Greenwald-Khanna algorithm [GK01] for uniform quantiles ($\phi\pm\varepsilon$), but need new implementation and analysis.

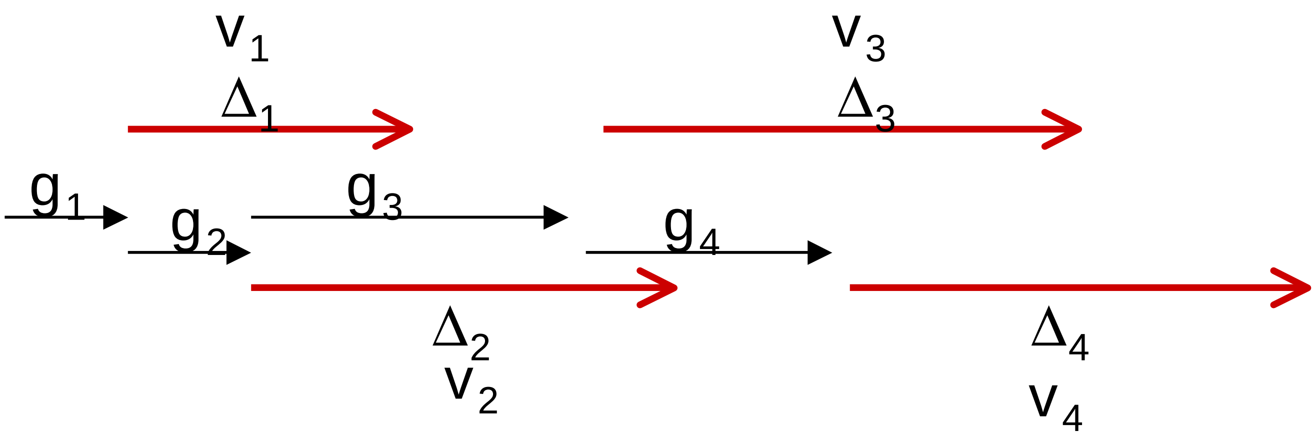
Data Structure

Store tuples $t_i = (v_i, g_i, \Delta_i)$ sorted by v_i

- v_i is an item from the stream
- $g_i = r_{min}(v_i) - r_{min}(v_{i-1})$
- $\Delta_i = r_{max}(v_i) - r_{min}(v_i)$

Define $r_i = \sum_{j=1}^{i-1} g_j$

We will guarantee that the true rank of v_i is between $r_i + g_i$ and $r_i + g_i + \Delta_i$



Biased Quantiles Invariant

In order to guarantee accurate answers, we maintain at all times for all i:

$$\underbrace{g_i + \Delta_i}_{\substack{\text{“uncertainty”} \\ \text{in rank of } v_i}} \leq \max\{\underbrace{2\varepsilon r_i}_{\substack{2\varepsilon \text{ times lower bound} \\ \text{on rank of } v_i}}, 1\}$$

Intuitively, if the uncertainty in rank is proportional to ε times a lower bound on rank, this should give required accuracy

Output Routine

Output(ϕ):

$r_0 := 0$;

for i := 1 to s do

    $r_i := r_{i-1} + g_{i-1}$;

    if $(r_i + g_i + \Delta_i > (\phi n + \epsilon \phi n))$

        print(v_{i-1}); break;

Compute r_i

Output *previous* item, v_{i-1}

max rank of v_i

Upper bound on allowed rank

Claim: Output(ϕ) correctly outputs ε–approximate ϕ-biased quantile

Proof

i is the smallest index such that

$$r_i + g_i + \Delta_i > \phi n + \varepsilon\phi n \quad (*)$$

So $r_{i-1} + g_{i-1} + \Delta_{i-1} \leq (1 + \varepsilon)\phi\, n.$ [+]

Using the invariant on (*), $(1 + 2\varepsilon)r_i > (1+\varepsilon)\phi n$ and (rearranging) $r_i > (1-\varepsilon)\phi n.$ [-]

Since $r_i = r_{i-1} + g_{i-1}$, we combine [-] and [+]:

[-] $(1-\varepsilon)\phi n < r_{i-1} + g_{i-1}$

$$\leq (\text{true rank of } v_{i-1}) \leq$$

$$r_{i-1} + g_{i-1} + \Delta_{i-1} \leq (1+\varepsilon)\phi n \text{ [+]}$$

Inserting a new item

We must show update operations maintain bounds on the rank of v_i and the BQ invariant

To insert a new item, we find smallest i such that $v < v_i$

- Set $g = 1$ (rank of v is at least 1 more than v_{i-1})
- Set $\Delta = \max\{2\varepsilon\, r_i, 1\}-1$ (uncertainty in rank at most one less than $\Delta_i \leq \max\{2\varepsilon\, r_i, 1\}$)
- Insert (v, g, Δ) before t_i in data structure

Easy to see that Insert maintains the BQ invariant

Compressing the Data Structure

Insert(v) causes data structure to grow by one tuple per update. Periodically we can Compress the data structure by pruning unneeded tuples.

Merge tuples $t_i = (v_i, g_i, \Delta_i)$ and $t_{i+1} = (v_{i+1}, g_{i+1}, \Delta_{i+1})$ together to get $(v_{i+1}, g_i + g_{i+1}, \Delta_{i+1})$.

$\Rightarrow$ Correct semantics of g and Δ

Only merge if $g_i + g_{i+1} + \Delta_{i+1} \leq \max\{2\varepsilon r_i, 1\}$

$\Rightarrow$ Biased Quantiles Invariant is preserved

k-biased Quantiles

Alternate version: sometimes we only care about, eg, $\phi = \frac{1}{2}, \frac{1}{4}, \ldots \frac{1}{2}^k$

Can reduce the space requirement by weakening the Biased Quantiles invariant:

k-BQ invariant:

$$g_i + \Delta_i \leq 2\varepsilon \max\{r_i, \phi^k n, \varepsilon/2\}$$

Implementations were based on the algorithm using this invariant.

Experimental Study

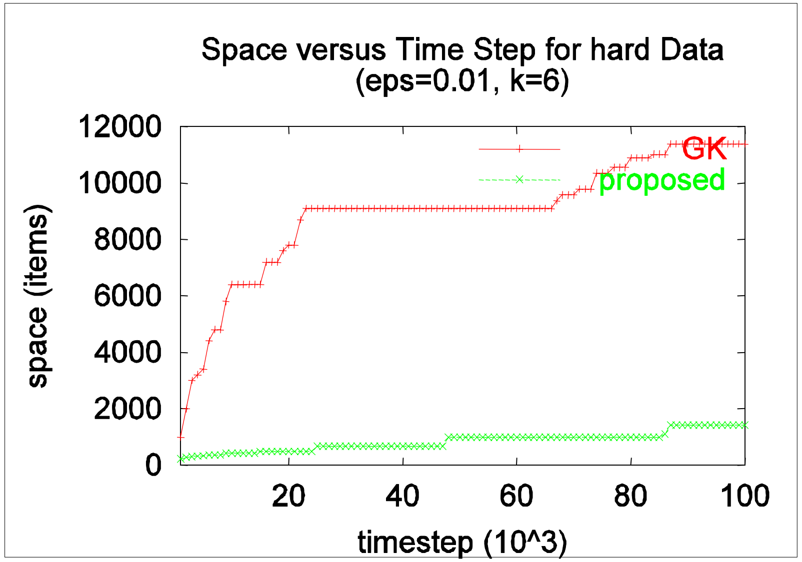
The k-biased quantiles algorithm was implemented in the Gigascope data stream system.

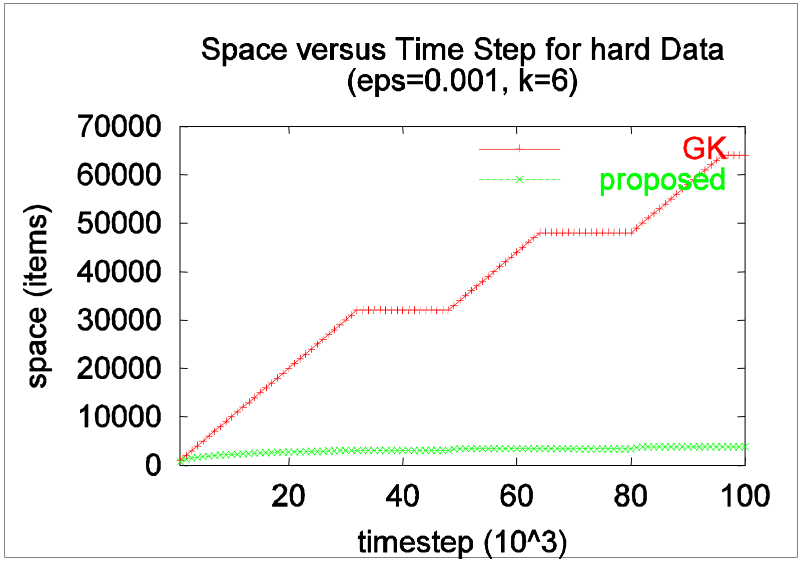
Ran on a mixture of real (155Mbs live traffic streams) and synthetic (1Gbs generated traffic) data.

Experimented to study:

- Space Cost
- Observed accuracy for queries
- Update Time Cost

Experiments: Space Cost

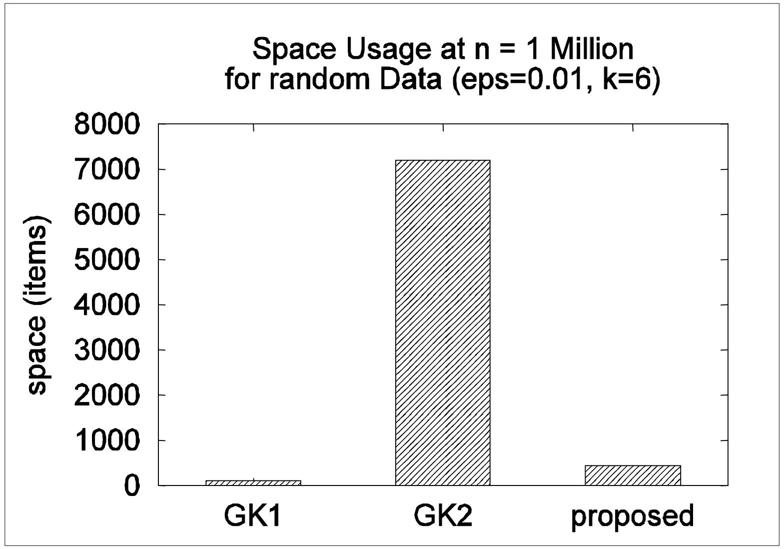


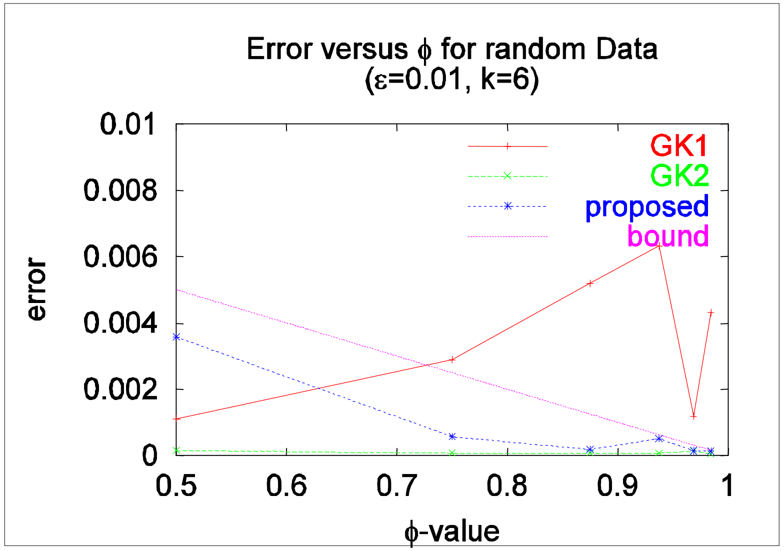


k-biased quantiles, vs. GK with ε = eps ϕ^k

$\Rightarrow$ Space usage scales roughly as $k/\varepsilon \log^c \varepsilon N$ on real data, but grows more quickly in worst case.

Experiments: Accuracy





GK1: ε = eps

GK2: ε = eps ϕ^k

Good tradeoff between space and error on real data

Experiments: Time Cost

Overhead per packet was about 5 – 10μs

Few packet drops (<1%) at Gigabit ethernet speed.

Choice of data structure to implement the list of tuples was an important factor.

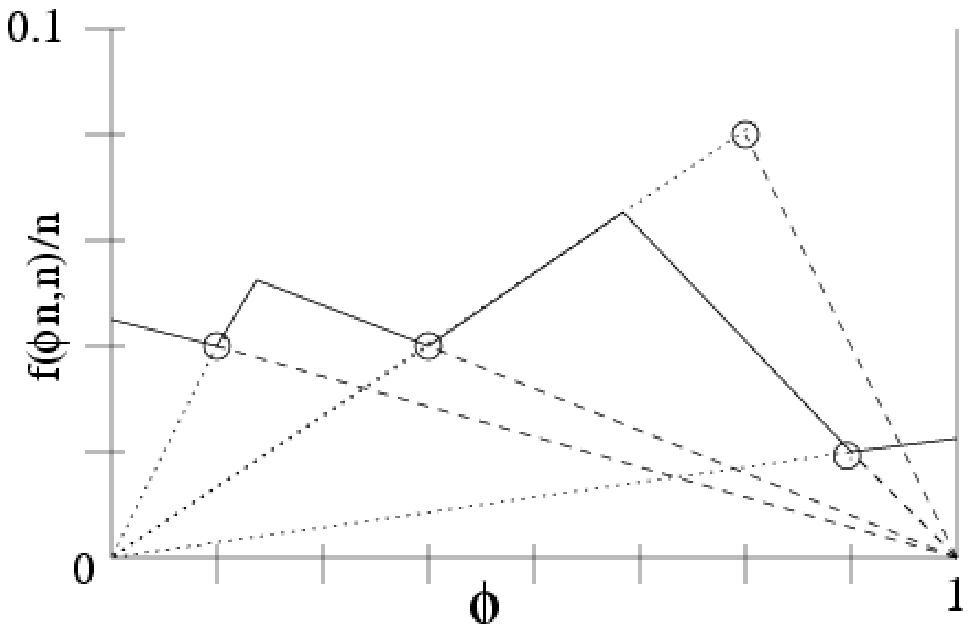
- running compress periodically is blocking operation. Instead, do a partial compression per update
- "cursor" + sorted list (5μs / packet) does better than balanced tree structure (22μs / packet)

Extension: Targeted Quantiles

Further generalization: before the data stream, we are given a set T of (ϕ,ε) pairs.

We must be able to answer ϕ-quantile queries over data streams with error $\pm\varepsilon n$.

From T, generate new invariant f(r,n) to maintain:



In paper, we show that maintaining $g_i + \Delta_i \leq f(r_i,n)$ guarantees targeted quantiles with required accuracy.

Deletions

For uniform quantile guarantees, can handle item deletions in probabilistic setting (with CM sketch)

But, provably need linear space for biased quantiles (with a strong “adversary”), even probabilistically

Sliding window also requires large space.

Conclusions

Skew is prevalent in many realistic situations

- By taking account of the skew inherent in most realistic data sources, can considerably improve results for summarizing and mining tasks.
- New problems eg Biased Quantiles give a non-uniform way to study skewed data.

Many other tasks can benefit from incorporating skew either into the problem, or into the analysis of the solution.

Extensions

Applying skewed data mining to other structured domains: hierarchical domains, graph data etc.

Work in progress: new algorithm for Biased Quantiles with provable space bounds, extension to multi-dimensional data etc.