



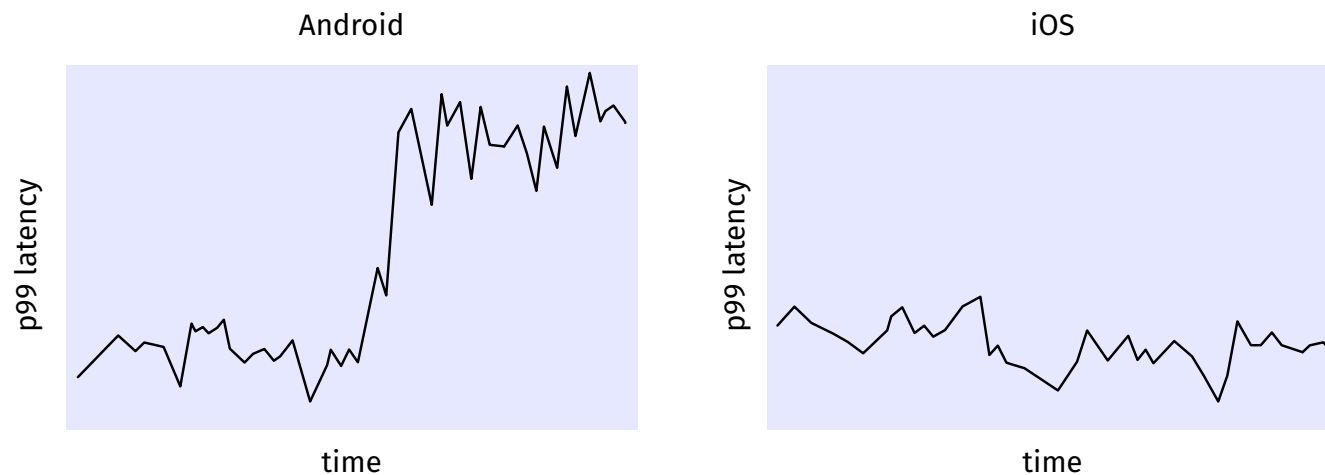
Moment-Based Quantile Sketches for Efficient Aggregation Queries

Edward Gan, Jialin Ding, Kai Sheng Tai, Vatsal Sharan, Peter Bailis

Stanford University

Motivation: Monitoring production data streams

Billions of events / day of mobile app telemetry data



Query for 99-th **percentile**

Group By Operating system

Where Location = USA

Quantile Query

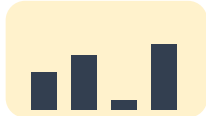
Spike in response latency, need to issue queries:

Percentiles are targeted: single metric, for specific sub-populations

Goal: Enabling fast quantile queries at scale



Baseline: Scan and sort billions of rows, multi-second latencies



Data Summaries

+

$$\mu_i = \frac{1}{n} \sum_{x \in X} x^i$$

Statistics

+

$$\theta' = \theta - \frac{\nabla F(\theta)}{\nabla^2 F(\theta)}$$

Optimization

= Scalable Queries

Systems make use of summaries to scale



Summaries represent a dataset using sublinear space (e.g. histogram)

Quantile estimates can be extracted from a *quantile summary*

Commonly used to avoid sorting large datasets

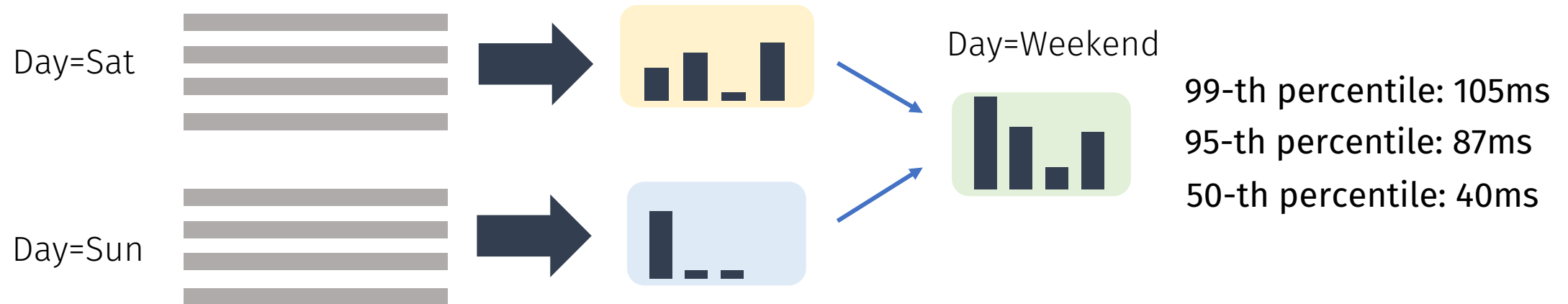


Pre-aggregating summaries reduces latency

Systems can pre-aggregate summaries for populations ahead of time



Data associated with day of week



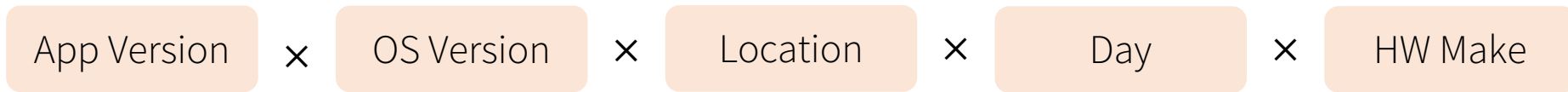
*Mergeable summaries*¹ can be combined without loss of accuracy

Improved query response time

1: [Agarwal et al, PODS '12]

Challenge: aggregations bottlenecked by merge

Many attributes means potentially more pre-aggregated subpopulations



5 columns x 20 distinct values each = **3.2M combinations**

Queries bottlenecked when merging pre-aggregated summaries

Greenwald Khanna Sketch: updatable equi-depth histogram
GK Performance: **3 μ s x 1 million merges = 3 seconds**

How can we optimize quantile summaries for aggregation?

Talk Outline

1. Setting: Quantile roll-ups at scale
2. Challenge: merging pre-aggregated summaries
- 3. Summarizing data using statistics (moments sketch)**
4. Improving sketch performance
5. Results: benchmark + integrated into data systems

Efficient data summaries using statistics

How can we optimize quantile summaries for aggregation?

Use statistics to summarize sub-populations (indexing)



Aggregate statistics using arithmetic (query time)

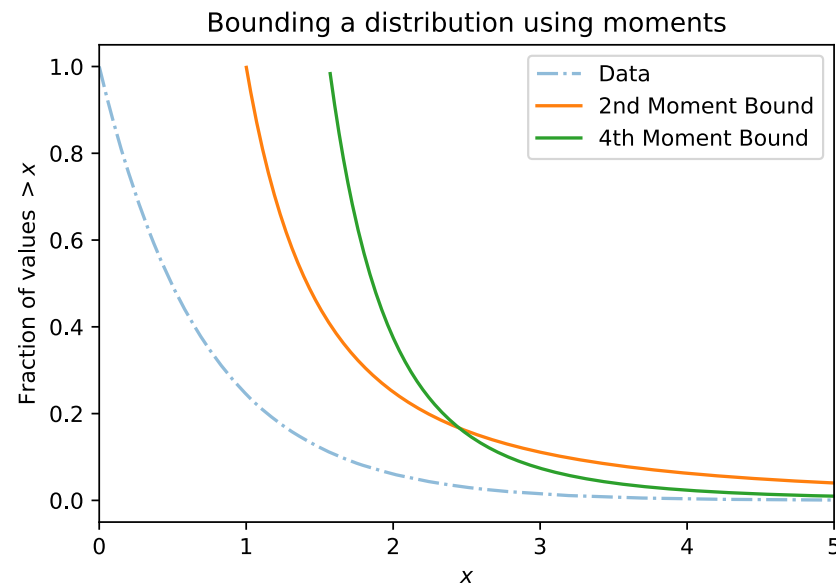


Moments: statistics that capture distribution shape

Moments: averages of powers of the data values.

i^{th} moment: $\mu_i = \frac{1}{n} \sum_{x \in X} x^i$ The first moment is the mean.

Intuition: Averages bound the number of “large” values



$\frac{1}{n} \sum x^2 = 1$ limits size of the tail

$\frac{1}{n} \sum x^4 = 6$ further limits size of tail

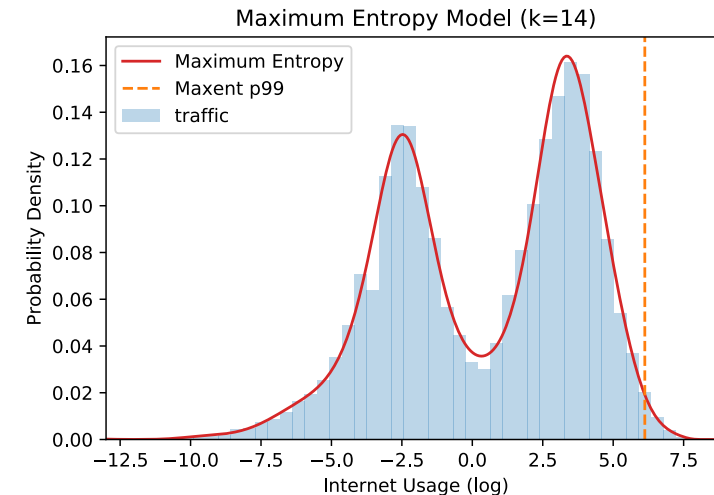
Given k moments,
distribution known to within $O(1/k)$,
Can estimate quantiles

Quantile estimates from moments

Method-of-moments¹: technique for estimating distribution parameters given moments. Used in statistics, econometrics, physics

Given k moments, we solve for the unique distribution that:

1. Matches all k moments
2. Minimizes unwarranted assumptions about the data (maximizes entropy)



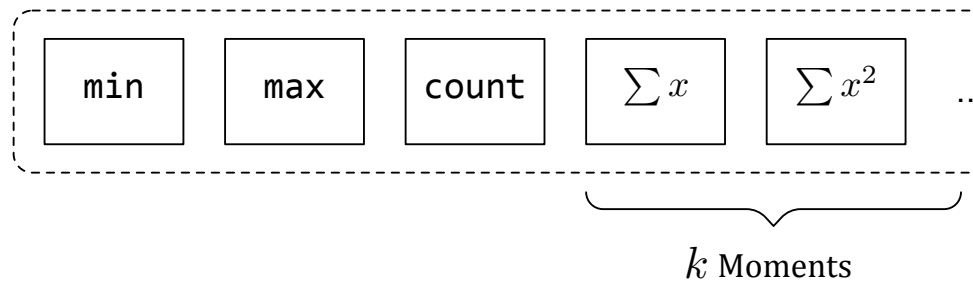
With 14 moments, can estimate quantiles with 1% error on real data

1: [Wasserman, 2004], 2: [Jaynes, Phys. Rev. 1957]

Moments for fast quantile queries

Our contribution: method of moments can be used for *compact, efficiently mergeable, and accurate* sketches

The **moments sketch**: an array tracking the min, max, count, and k sums of powers ($2 \leq k \leq 15$)

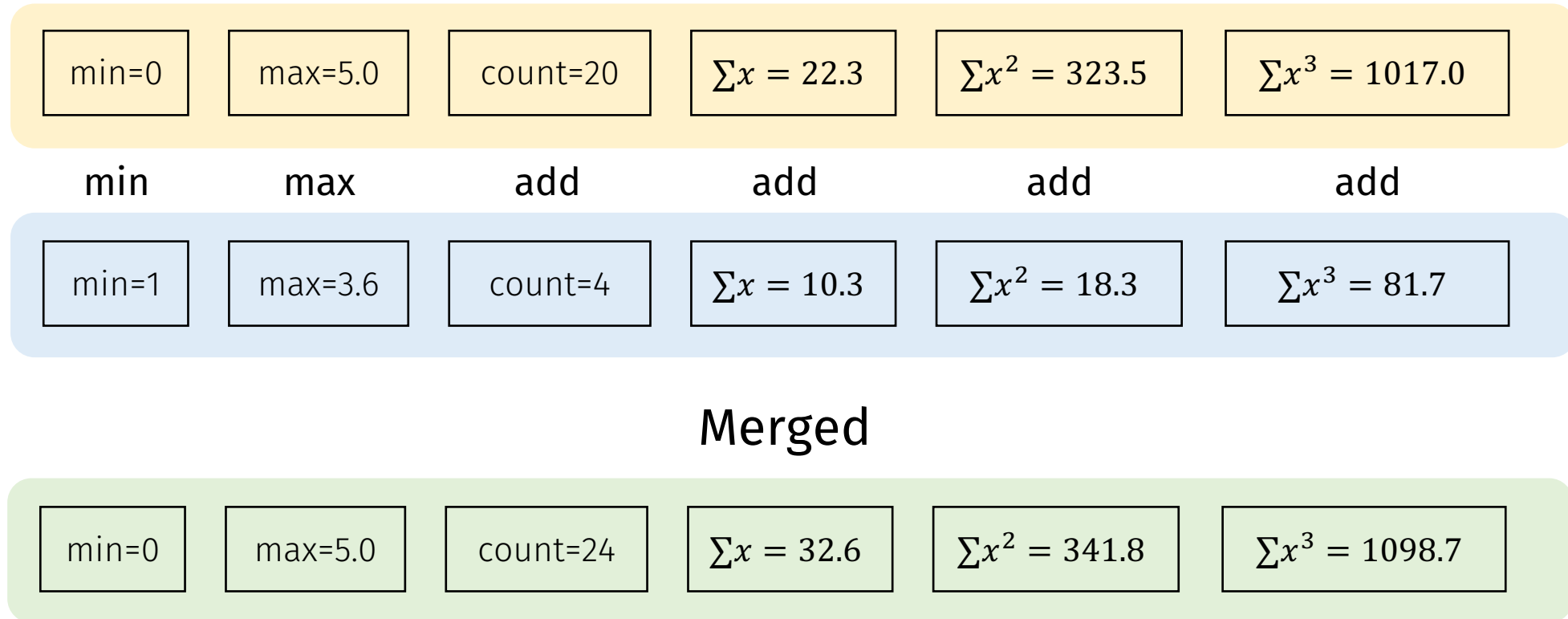


Goals: low-memory footprint, fast to aggregate, fast to compute quantiles

Talk Outline

1. Setting: Quantile roll-ups at scale
2. Challenge: merging pre-aggregated summaries
3. Summarizing data using statistics (moments sketch)
- 4. Improving sketch performance**
5. Results: benchmark + integrated into data systems

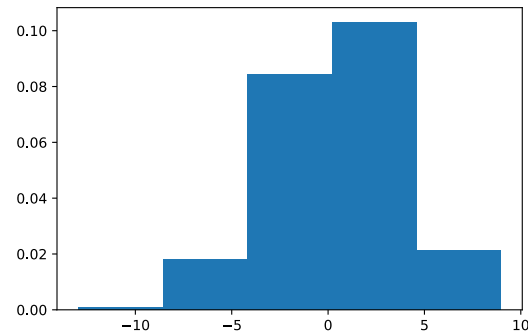
Moments sketch has low aggregation overhead



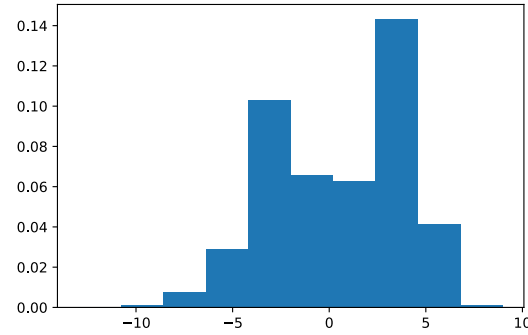
$k + 3$ primitive floating point operations
< 200 bytes in practice, $k < 20$

Method-of-moments is now a bottleneck

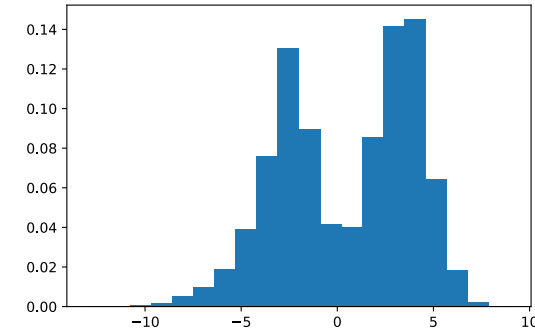
Solving quantiles from moments requires iterative optimization



Iteration 1



Iteration 2



Iteration 3

Increasingly precise agreement with known moments

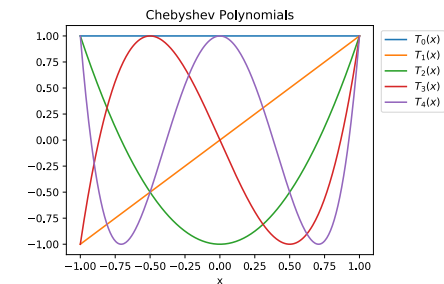
Challenge: Off the shelf solver routines too slow for interactive use

Convex optimization packages (cvxpy): 300ms to solve

Improvement: Accelerating method-of-moments

Practical improvements to a specialized optimization routine bring solve time down to 1ms

- Improved numeric stability using Chebyshev polynomials
- Fast integration using polynomial approximation
- Selection of most informative moments
- Cascades for short-circuiting percentile estimation

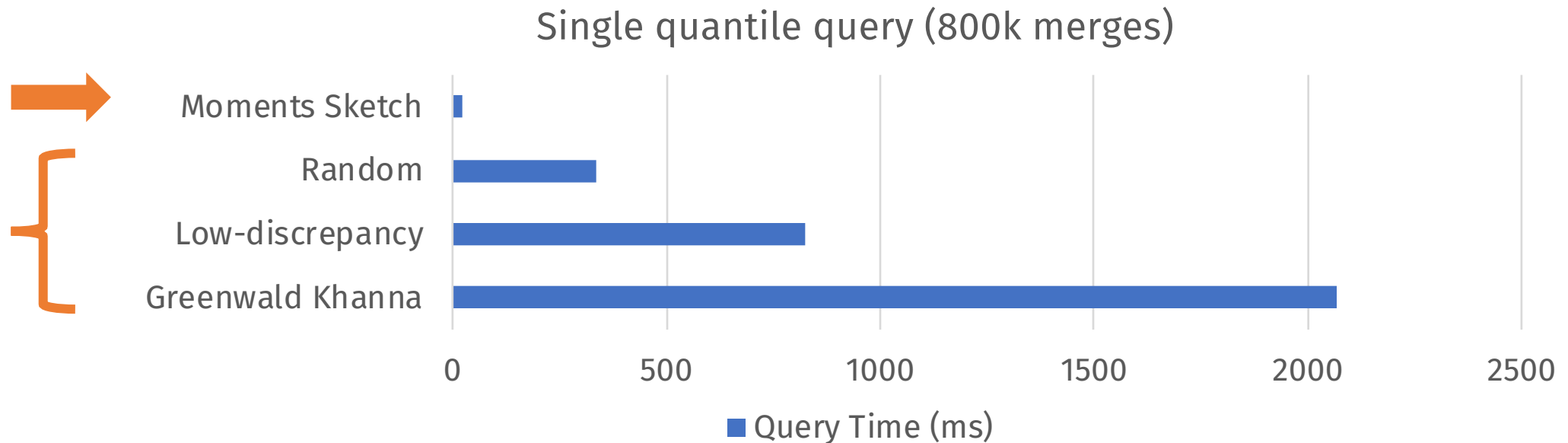


See paper (VLDB 2018) for details!

Talk Outline

1. Setting: Quantile roll-ups at scale
2. Challenge: merging pre-aggregated summaries
3. Summarizing data using statistics (moments sketch)
4. Improving sketch performance
- 5. Results: benchmark + integrated into data systems**

Moments sketch enables fast roll-up queries



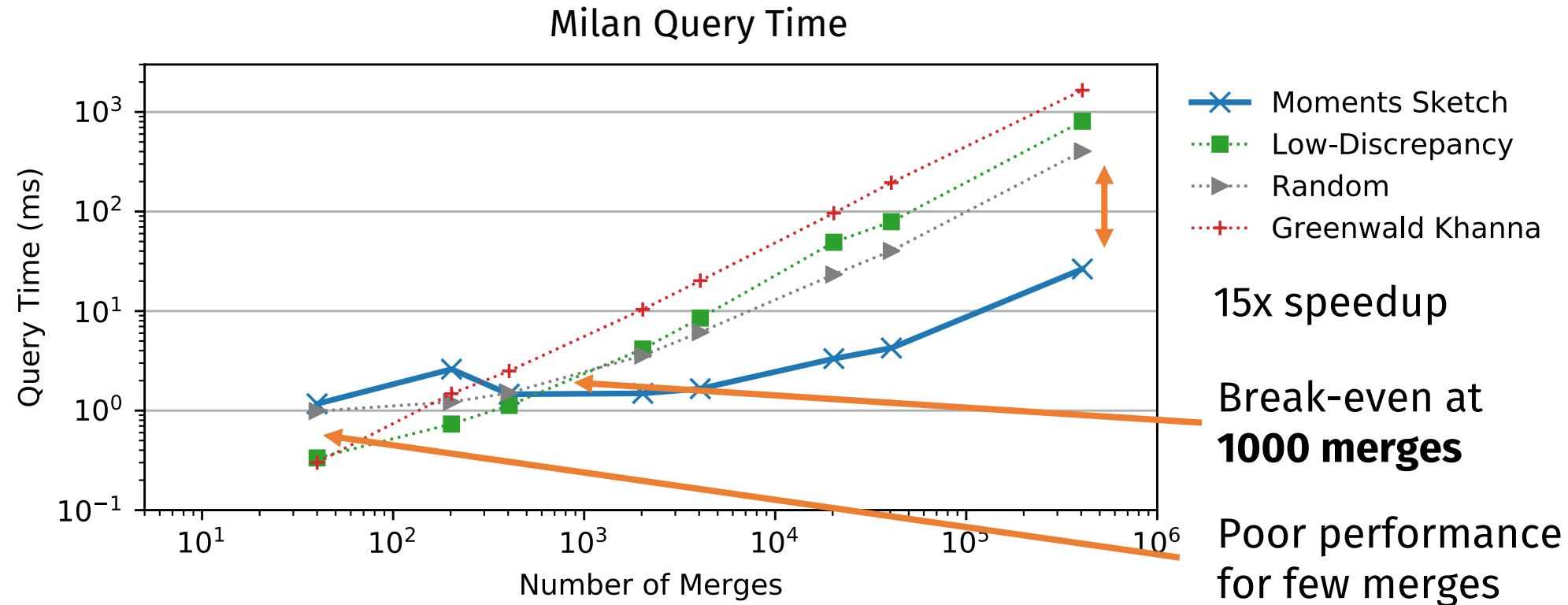
Milan: Single quantile query on cellular internet usage measurements

Moments Sketch: **23ms** total query time on 800k merges

Comparably accurate mergeable summaries **15x slower**

Random: [Wang et al, SIGMOD'13], Low-discrepancy: [Agarwal et al, TODS'13], Greenwald Khanna: [Greenwald et al, SIGMOD'01]

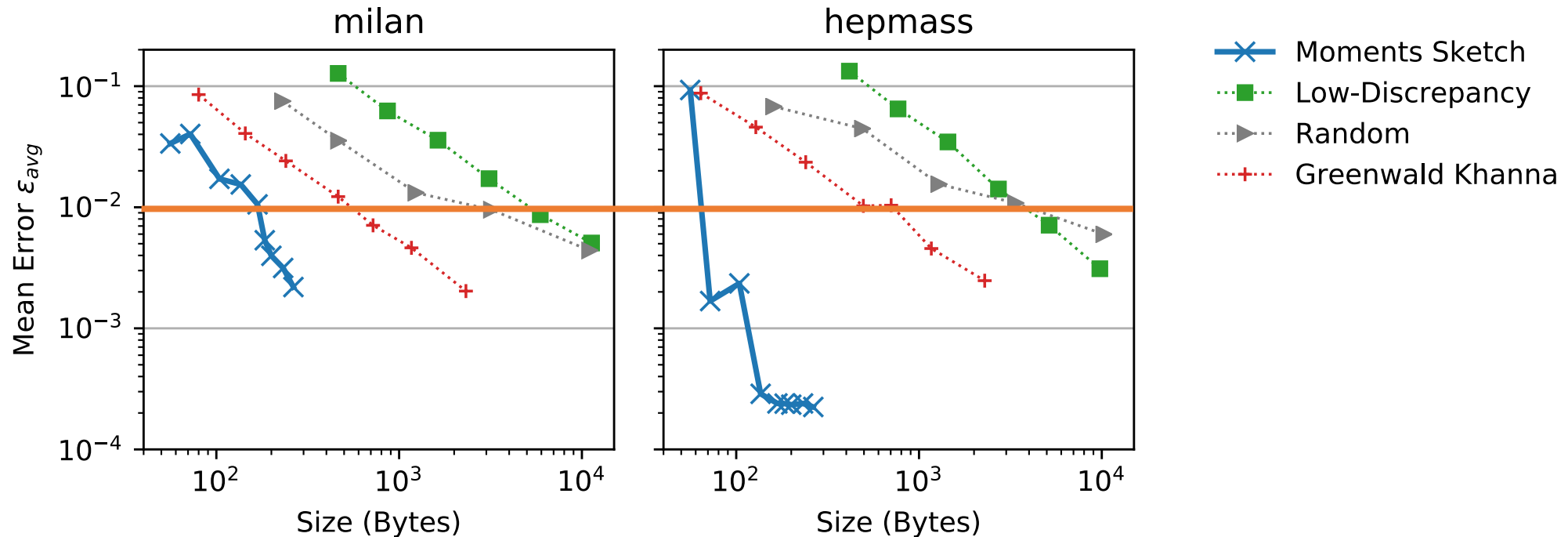
Performance scales with number of merges



Varying the number of merges in a single query

Moments sketch has fixed query overhead: method-of-moments solver

Accurate estimates with low space overhead

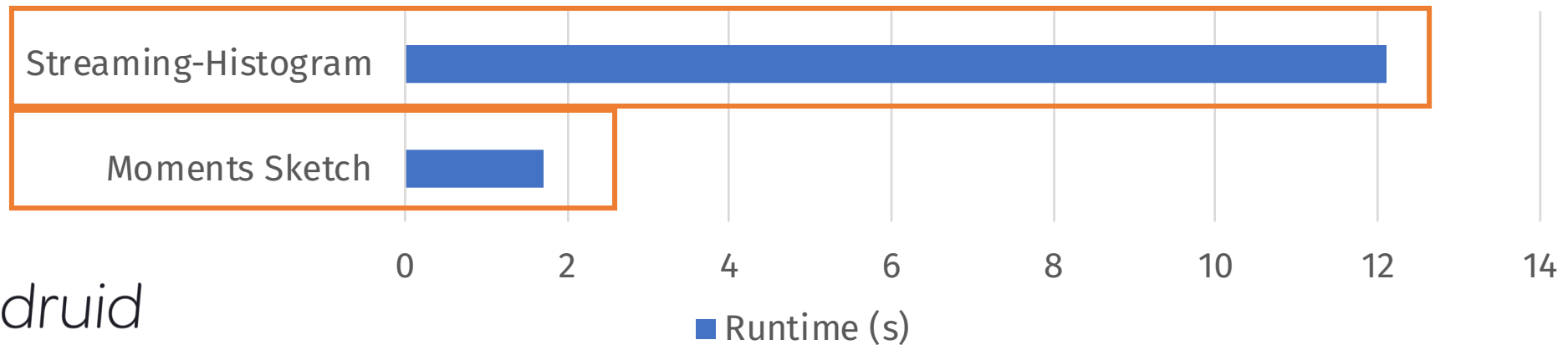


Accuracy of sketch varies with space used (bytes): error decreases

Moments sketch achieves 1% error with less space than other summaries

Performance translates to real-world systems

Druid Query (10M merges)



Single quantile query over 10M summaries from the Milan data

Default streaming-histogram¹ sketch performance is poor

Integrated moments sketch as a *user defined aggregate* in Druid:
Better accuracy, **7x faster end-to-end queries**

1: [Ben-Haim et al, JMLR'10]

Moments Sketch in MacroBase

MacroBase¹ uses quantile queries to search for explanations:

Find attribute values which have
97th percentile latency > 100ms

App Ver 7.0: p97 = 125ms

App Ver 6.5: p97 = 60ms

USA, Android: p97 = 250ms

...

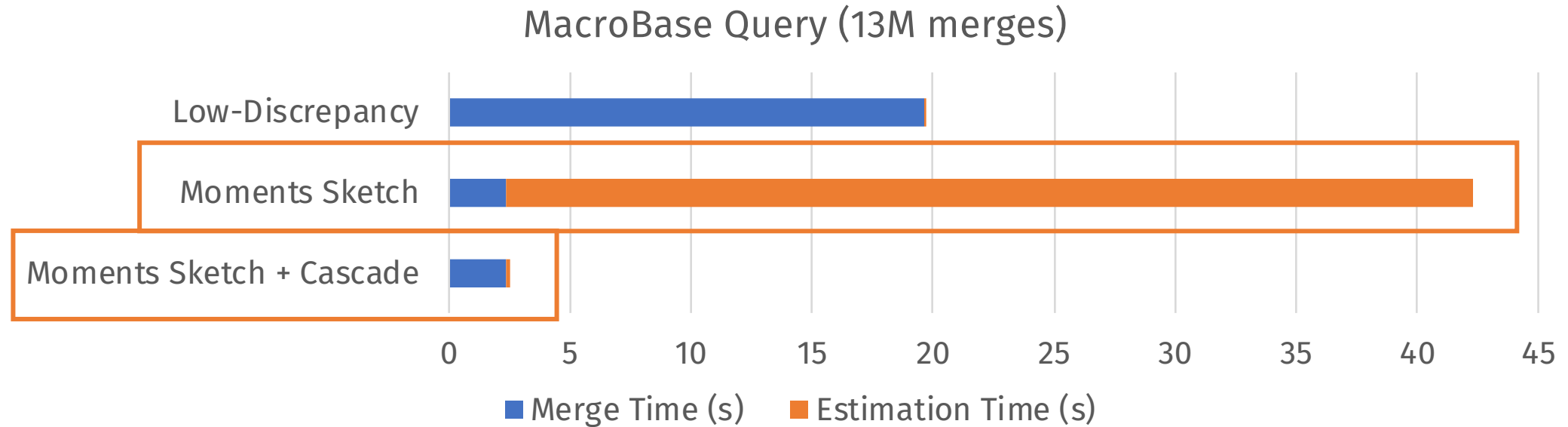
Using sketches lets us ingest pre-aggregated data:
less network bandwidth vs raw event logs



Challenge: solve for quantiles on *many* attribute combinations

1: [Bailis et al, SIGMOD'17]

Performance translates to real-world systems



Percentile(latency, 97) > 100ms

Naïve moments sketch usage expensive due to repeated search queries

Cascades for short circuit evaluation reduce query time by **17x**

Conclusion: Efficient sketches for quantile queries

Setting:

Quantile queries bottlenecked by slow summary data structures

Quantiles via Moments Sketch:

Statistical moments enable compact + efficient sketches

Method-of-moments + Entropy to recover distribution

Results:

15x faster query times, 7x integrated with Druid

Blog: dawn.cs.stanford.edu/2018/08/29/moments/

Code: github.com/stanford-futuredata/msketch

Edward Gan, edgan8.github.io