Data Summaries for Scalable High-Cardinality Analytics

Edward Gan

Stanford University

Outline and structure

Introduction

A. Motivating Context: ASP systems

B. Challenge: Query performance for high-cardinality analytics

Three new summaries to reduce performance bottlenecks:

1. Cooperative Summaries

2. Moments Sketches

3. TKDC

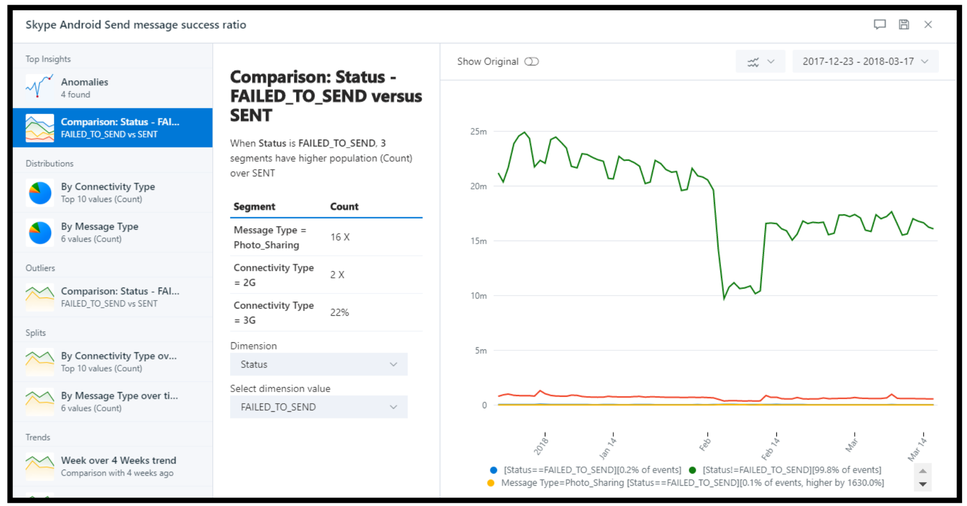
Discussion and Conclusion

Illustrative example: monitoring Microsoft Skype

Billions of application requests daily

Engineers issue queries through UIs to diagnose & explore trends

Azure **Monitoring Dashboard**



Example Skype Diagnostic Query:

```
Compute the 99th percentile latency for requests in Canada
from 9:00pm March 1 to 10:30pm March 11.
```

Processing analytics queries using direct scans is expensive

```
Compute the 99th percentile latency for requests in
Canada from 9:00pm March 1 to 10:30pm March 11.
```

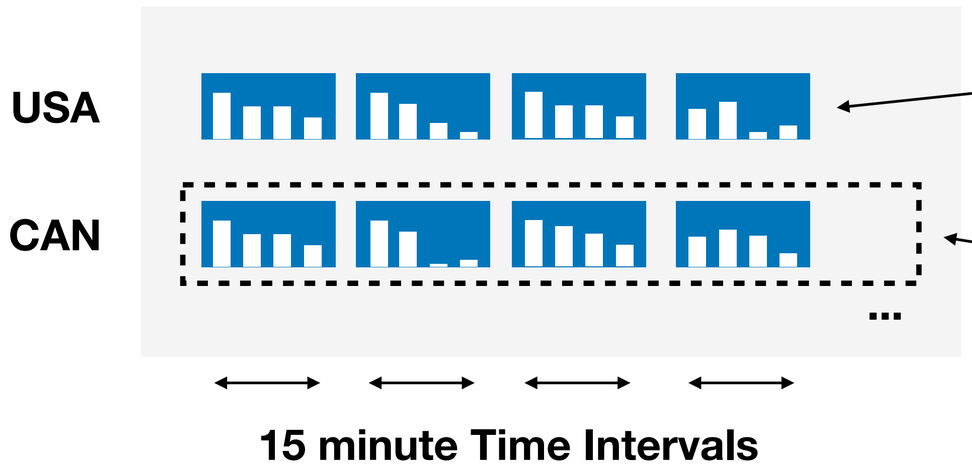
Goal: < 1s query response for interactive exploration

- Hadoop / Spark / Presto require aggregating and sorting billions of requests: minutes of CPU time
- Indexes, sub-sampling still require disk scans over many rows

Precomputing data summaries reduces query time overhead

```
Compute the 99th percentile latency for requests in
Canada from 9:00pm March 1 to 10:30pm March 11.
```

Goal: < 1s query response for interactive exploration

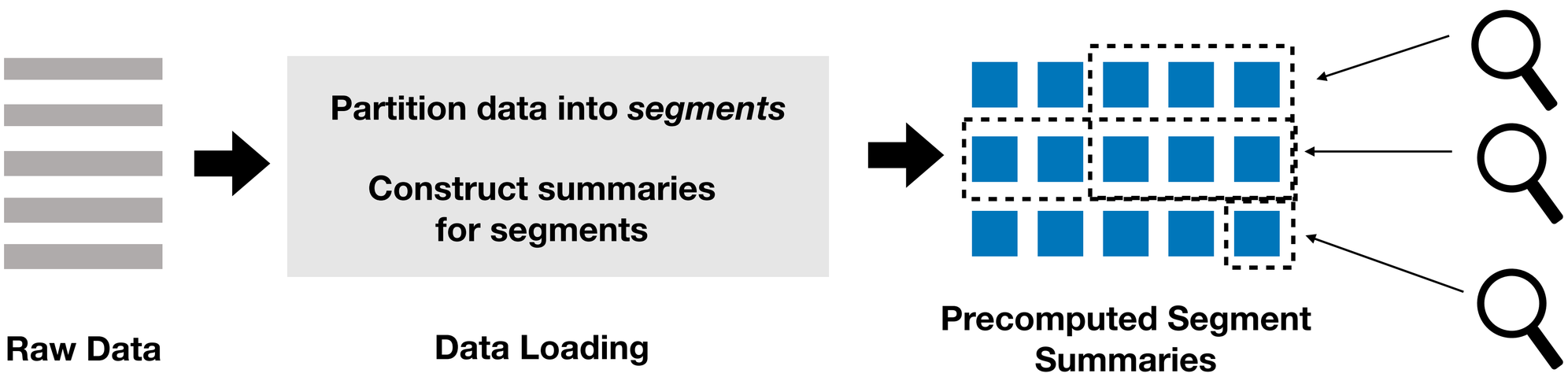


Precomputed in-memory summaries (e.g., histograms, sketches, samples) partitioned by query dimensions

Efficiently estimate quantiles for *flexible locations and time ranges* by combining summaries

Background: Approximate Summary Precomputation (ASP) Systems

Includes many open source, industry systems: druid Apache Kylin pinot



Summaries cached in-memory for maximum performance
Limited space for summaries introduces approximation error

Outline and structure

Introduction

A. Motivating Context: ASP systems

B. Challenge: Query performance for high-cardinality analytics

Three new summaries to reduce performance bottlenecks:

1. Cooperative Summaries
2. Moments Sketches
3. TKDC

Discussion and Conclusion

Segment counts are growing due to high-cardinality dimensions

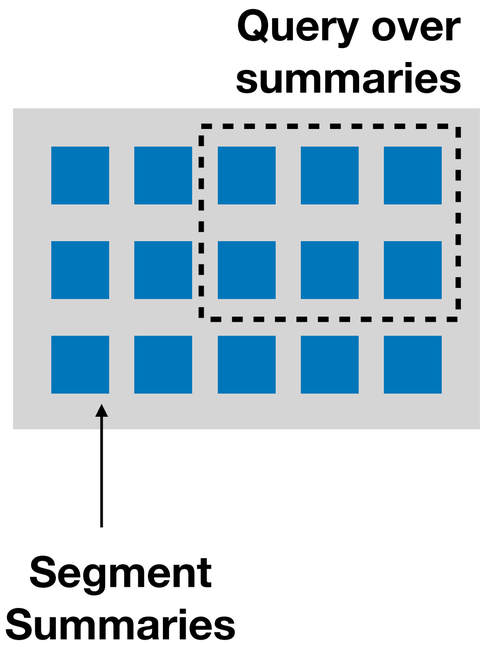
Users want to examine, diagnose trends with precise context

ASP systems precompute summaries for segments on **all queryable dimension values,** leading to explosion in segments

- 2,016 five-minute intervals in a week
- 24,000 Android device Types
- 29 tenants x 167 app versions x 472 providers: 2 million combinations

Challenge: ASP systems have trouble scaling to large numbers of segments

More segments, more summaries, more bottlenecks

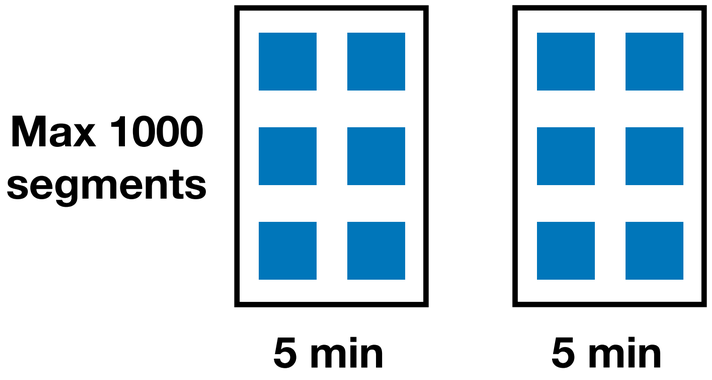


Performance bottlenecks along three axes:

1. Space: limited total memory to store summaries
2. Accuracy: limited by small summary sizes
3. Query Time: limited time to process summaries

Summary performance limits the usability of production ASP systems

Quantile Queries on Skype monitoring team:



- Data partitioned into 5-minute intervals and user-specified dimensions: locations, etc... Summarized using t-digests*
- t-digests provide limited accuracy under space constraints & are slow to combine

To maintain performance, team imposes a strict *cardinality* limit:
1000 segments per time interval. Many user complaints.

**[Dunning & Ertl. 2019]*

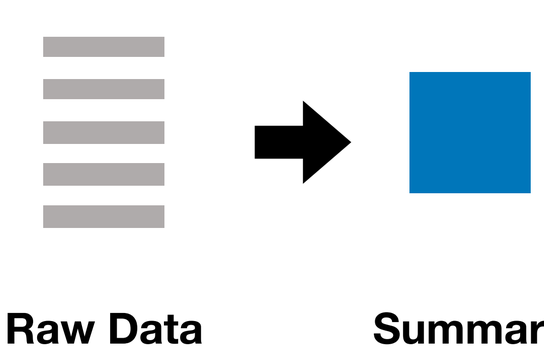
Thesis:
Summaries for high-cardinality aggregation

How can we support high-cardinality aggregations over precomputed summaries without sacrificing query performance?

Novel summaries tailored for high-cardinality workloads can provide **end to end improvements** in accuracy, space usage, and runtime.

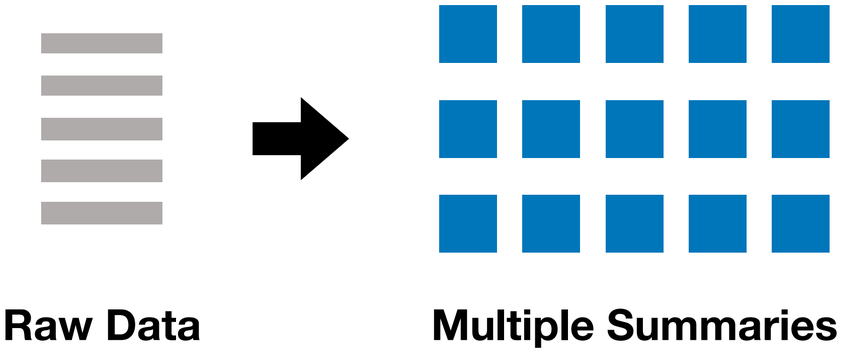
High-cardinality summary design

Traditional Summaries



Single summaries constructed and queried individually (e.g., independent streams)

Our Thesis



Summaries constructed and queried as a collection enables new optimizations

Main Contributions: Three summaries for three query types

	Query Type	Accuracy	Time	Space
1) Cooperative Summaries [Gan, Bailis, Charikar. VLDB Under Revision]	Frequencies & Quantiles	✔		✔
2) Moments Sketches [Gan, Ding, et al. VLDB 2018]	Quantiles		✔	✔
3) TKDC [Gan, Bailis. SIGMOD 2017]	Kernel Density Classification	✔	✔	

Outline and structure

Introduction

A. Motivating Context: ASP systems

B. Challenge: Query performance for high-cardinality analytics

Three new summaries to reduce performance bottlenecks:

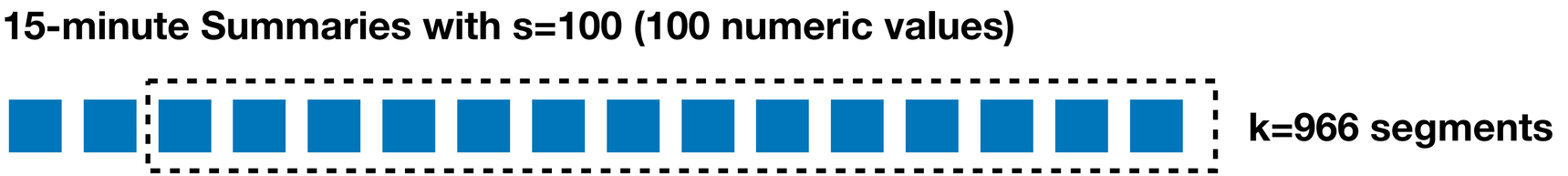
1. Cooperative Summaries: Improved accuracy for quantiles and frequencies
2. Moments Sketches: Improved runtime for quantiles
3. [Briefly] TKDC: Fast and accurate kernel density classification

Discussion and Conclusion

Existing mergeable summaries provide fixed accuracy at high cardinalities

```
Compute the 99th percentile latency for app requests
from 9:00pm March 1 to 10:30pm March 11.
```

Limited Space: SOTA summaries using space s can provide error O(1/s)



Mergeable Summaries*: error preserved when combining summaries

Query error is 1/100 = 1%

*[Agarwal, Cormode et al. TODS 2013]

Merging summaries is much less accurate than using larger summaries

15-minute Mergeable Summaries with s=100

k=966 segments

Query error 1/100 = 1%

Flexible but inaccurate

Ideal: one large summary with s=100*k

Query error $1/(sk) \approx .001\,\%$

Inflexible but accurate

Cooperative Summaries: Minimize query error when aggregating many summaries

15-minute Mergeable Summaries with s=100

k=966 segments

Query error 1/100 = 1%

Flexible but inaccurate

Ideal: one large summary with s=100*k

Query error $1/(sk) \approx .001\,\%$

Inflexible but accurate

Cooperative with s=100

k summaries, total space sk

Query error $\approx .005\,\%$

Flexible and accurate

Cooperative Summaries support quantile and frequency queries

Quantile Queries

Query Result

Compute the 99th percentile latency for app requests from 9:00pm March 1 to 10:30pm March 11. → 475 ms

Item Frequency Queries

How frequently does ip address 13.23.103.12 occur for requests in Canada? → 13023 counts

Cooperative Summaries consist of item counts

A cooperative summary consists of s item -> count mappings that approximate a segment item distribution with error $O(1/s)$

Item Frequencies

```
{
 a: 1330,
 b: 1725,
 c: 403,
 ...
}
```

Quantiles

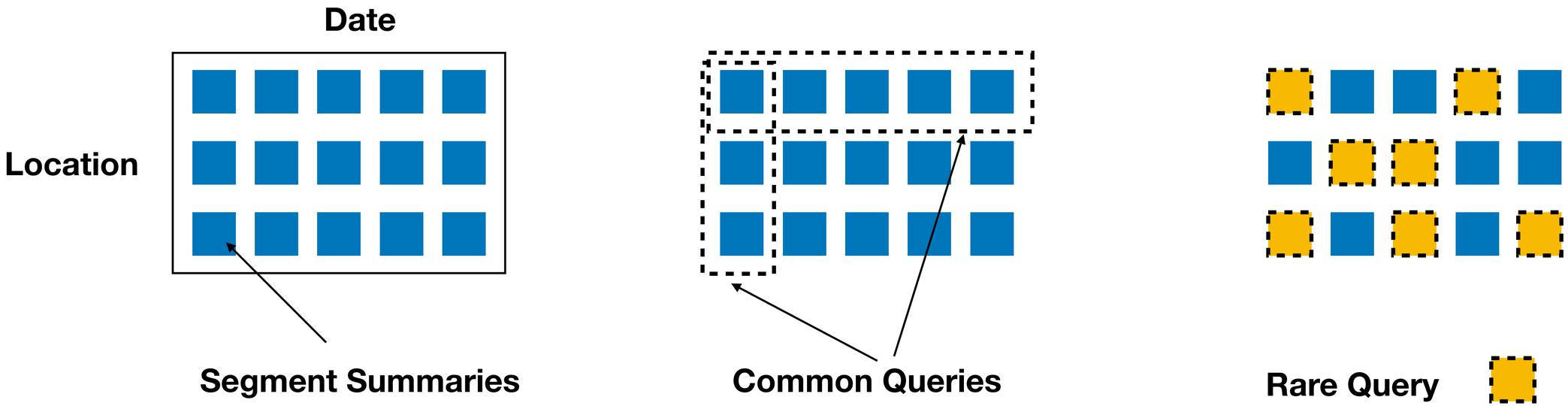
```
{
 5: 10,
 17: 31,
 31: 21,
 ...
}
```

Queries operate over summary counts as a proxy for the data in a segment.

Comparable in speed to merging existing summaries.

How to choose items and counts to maximize query accuracy?

Key Idea: optimizations can target common segment selection patterns

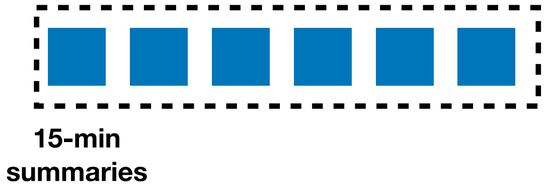


Two common selection patterns: intervals and data cubes

1. Interval Queries *[Basat et al. VLDB 2018], ...*

Selects segments in a contiguous 1-d range

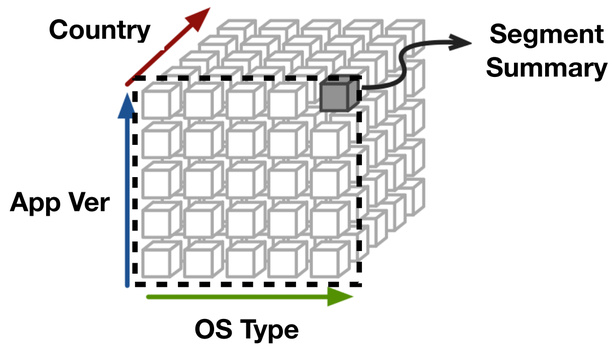
```
where lower <= t <= upper
```



2. Data Cube Queries *[Gray et al. DMKD 1997]*

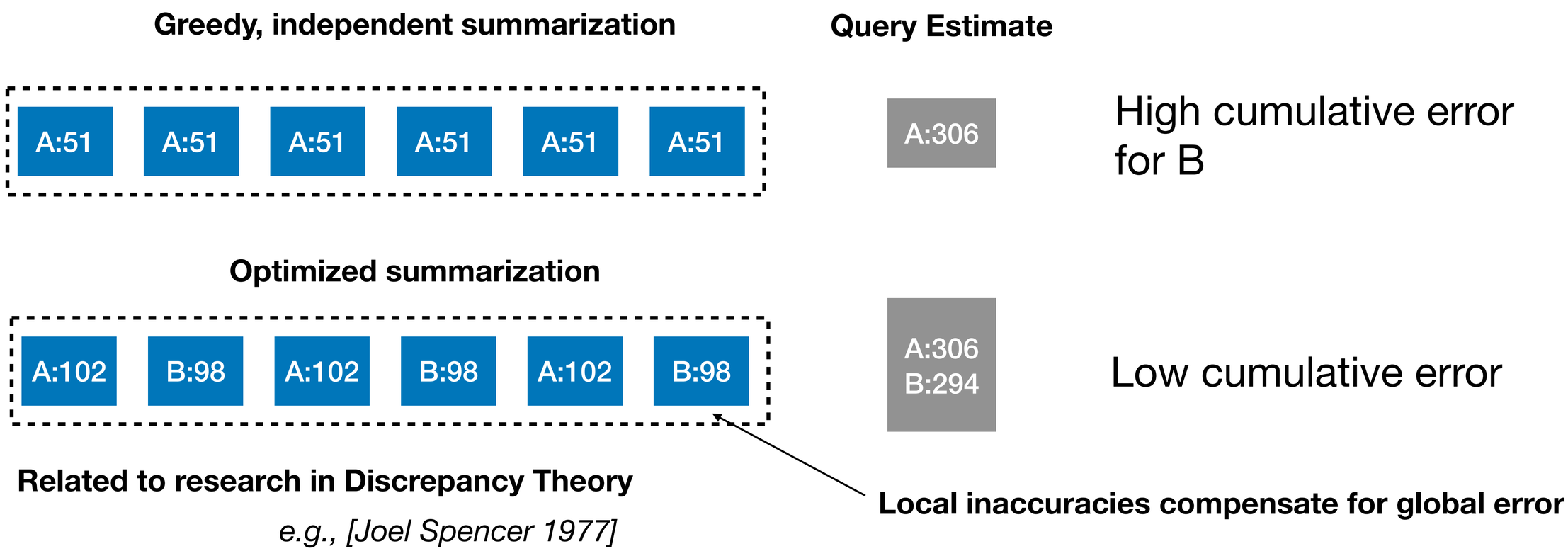
Select segments sharing dimension values

```
where d1 = v1 AND d2 = v2 ...
```



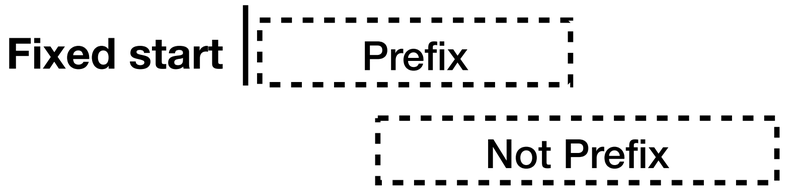
Known segment selection structures allows error optimization

Frequencies over 6 consecutive segments with 51% A, 49% B, Space limit s=1



For interval queries, Cooperative summaries exploit prefixes

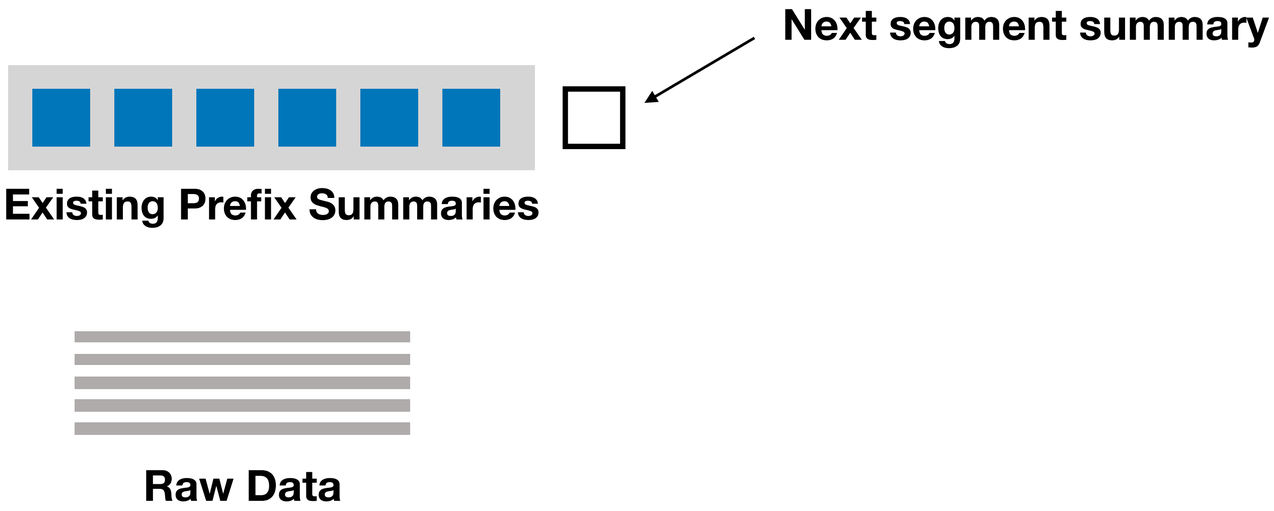
Prefixes are intervals with fixed start points



Prefixes allow Cooperative summaries to *incrementally* bound error

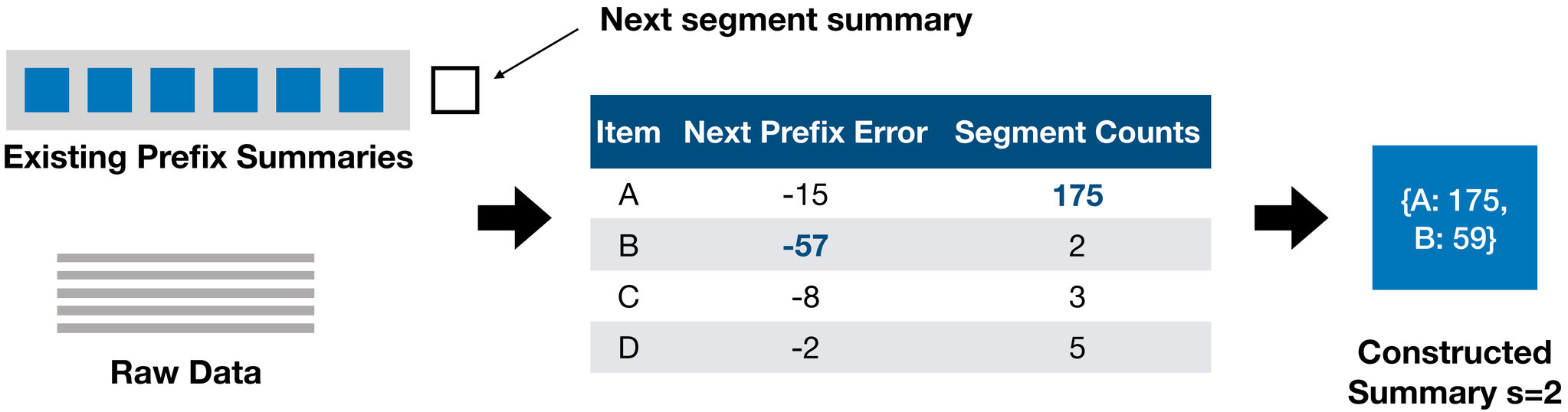
Bounding prefix errors also bounds error for **all intervals** using a reduction

Cooperative summaries use a greedy strategy to select items to include



1. Need to approximate the next single segment with error $O(1/s)$
2. Need to minimize accumulated error over prefixes

Cooperative summaries use a greedy strategy to select items to include



1. Need to approximate the a single segment with error $O(1/s)$

2. Need to minimize accumulated error over prefixes

Theory bounds on interval query error

Reduced error when aggregating intervals of k summaries of size s

Cooperative Frequencies: $|\epsilon| \leq O\left(\frac{\log(sk)}{sk}\right)$

Cooperative Quantiles: $|\epsilon| \leq O\left(1/(s\sqrt{k})\right)$

Proofs bound growth of a loss $L_k \approx \exp(|\epsilon_k|)$ for prefixes of length k, see paper for details

Empirical error $\approx O\left(\frac{1}{sk}\right)$

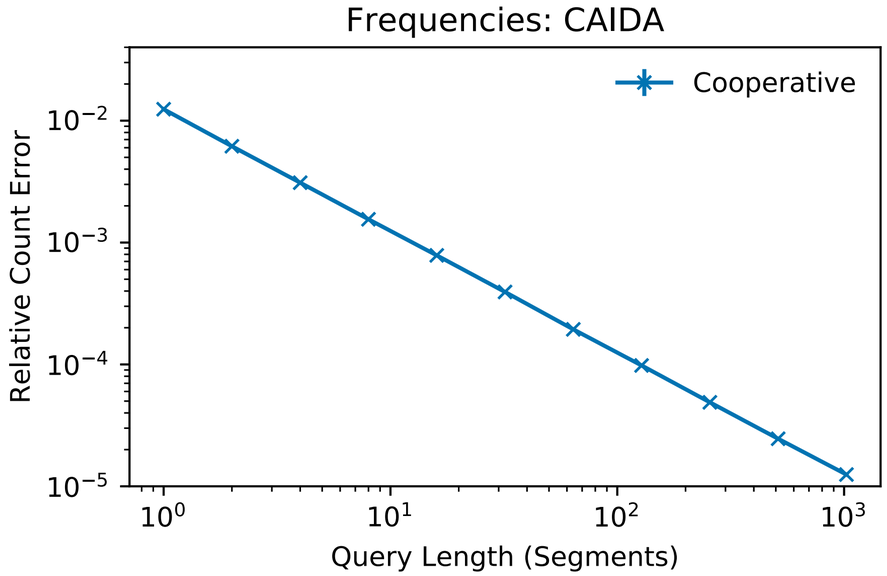
Mergeable summaries: $|\epsilon| \leq O(1/s)$

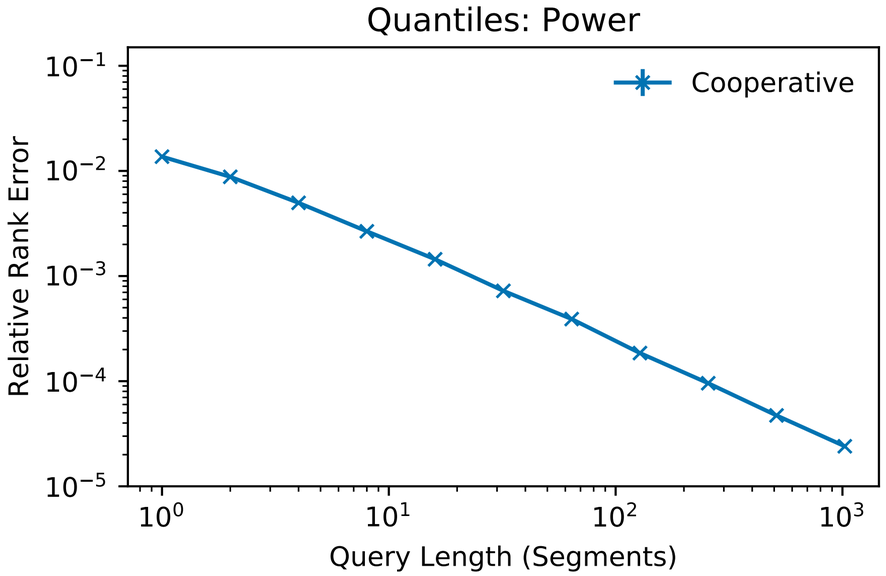
Hierarchical techniques*: $|\epsilon| \leq O\left(\log^2 k/(sk)\right)$

**[Basat et al. VLDB 2018]*

Results: Intervals

Query error over intervals of different lengths k

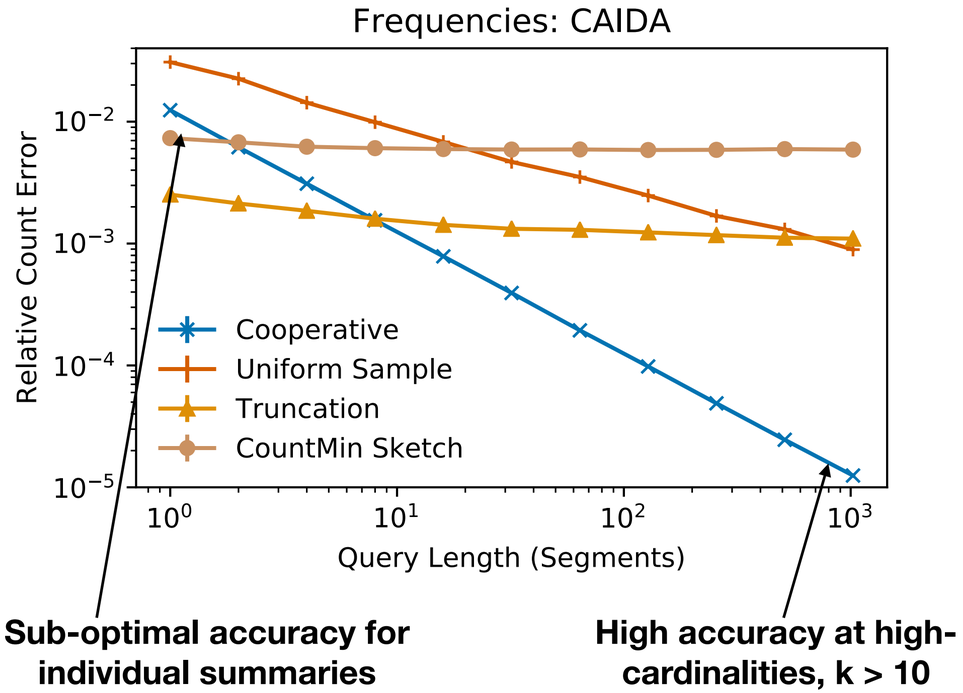


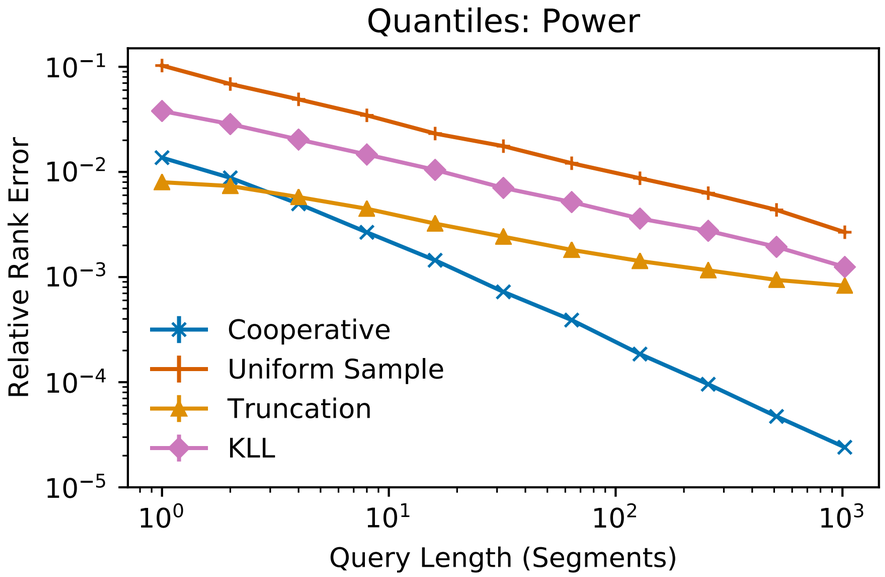


Query error falls $\approx 1/k$ as multiple summaries aggregated

Results: Intervals

Query error over intervals of different lengths





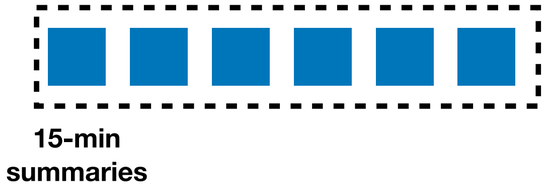
Count-Min Sketch [Cormode et al. 2005] KLL [Karnin, Lang, Liberty. 2016]

Cooperative summaries focus on interval and data cube selections

1. Interval Queries

Selects segments in a contiguous 1-d range

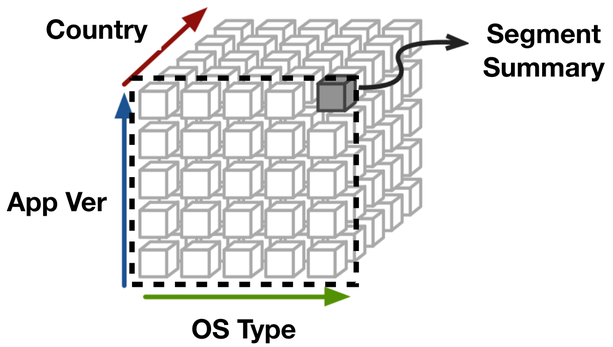
```
where lower <= t <= upper
```



2. Data Cube Queries

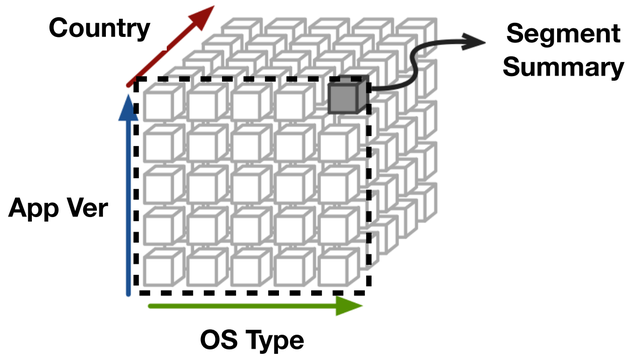
Select segments sharing dimension values

```
where d1 = v1 AND d2 = v2 ...
```



Weighted random samples reduce error for generic workloads

Data Cubes do not have single dimension to incrementally bound errors: have to consider queries on multiple axes



Probability Proportional to Size* (PPS) samples select s items and provide *randomized* frequency or quantile estimates with error at most O(1/s).

Accumulating t unbiased and random estimators has error $\sum_{i=1}^{k} \epsilon_i \leq O\left(\frac{\epsilon}{\sqrt{k}} \log 1/\delta\right)$

**[Cohen, Cormode, Duffield. VLDB 2011]*

PPS summary error depends on space and bias allocation

Probability Proportional to Size (PPS) provide *randomized* error at most O(1/s).

1. How to allocate space between multiple cube segments?*

**known for uniform samples, STRAT: [Chaudhuri et al. TODS 2007]*

Can allow some *bias* in frequency estimates to reduce variance, but consistent bias will accumulate over multiple summaries.

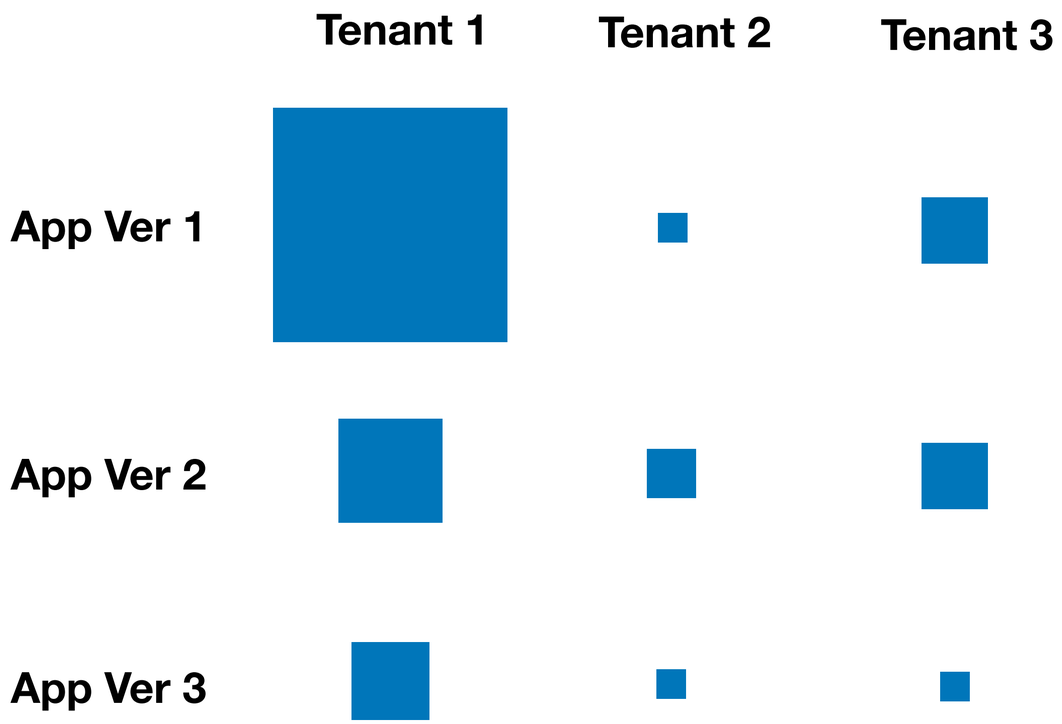
2. How to allocate bias-variance tradeoff over multiple cube segments?

Optimization: Data cubes have skewed dimension and segment sizes

Dimension value distributions from 10 million rows of a Microsoft request log:

Dimension	Top Value Count	Median Value Count	Min Value Count
Tenant	6.0 million	190	1
App Ver	4.8 million	40	1
Time Zone	2.2 million	13,000	2
...			

Some segments have disproportionate impact on queries that include them



Cooperative summaries minimize average error over expected workload

Per-query mean square error:

$$E\left[\epsilon'^{2}_{Q}\right] \leq |Q|^{-2}\left(\left(\sum_{\mathcal{D}_i \in Q} b_i\right)^2 + \sum_{\mathcal{D}_i \in Q} \frac{1}{4}\left(\frac{n_i\,[b_i]^2}{s_i^2}\right)\right)$$

bias term for summary

adjusted total data in segment

space allocated to summary

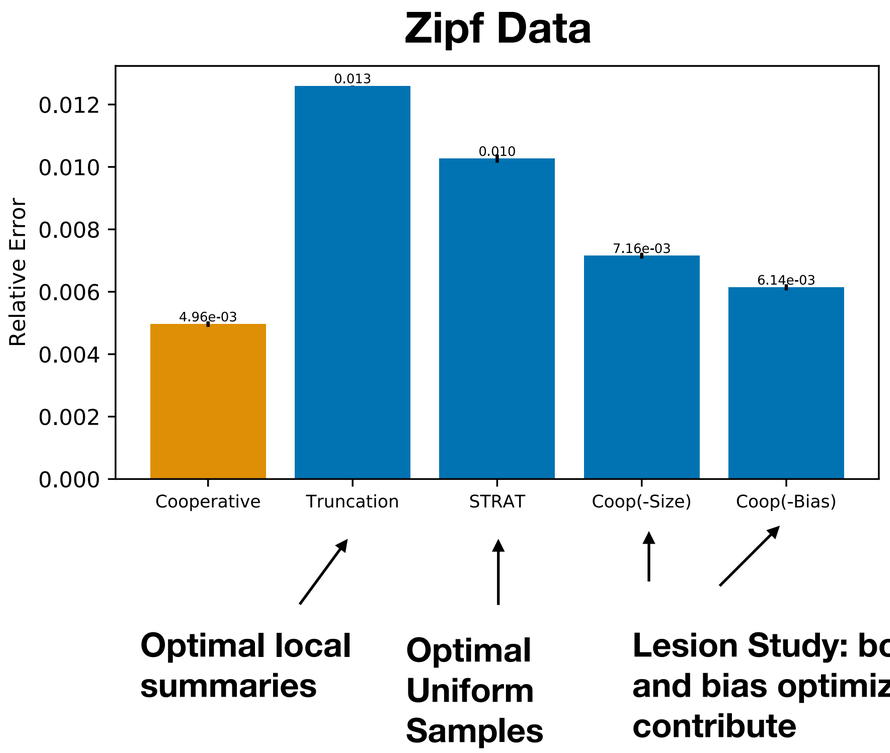
Optimize for workload error: $E_{Q \sim W}[E[\epsilon'^{2}_{Q}]]$

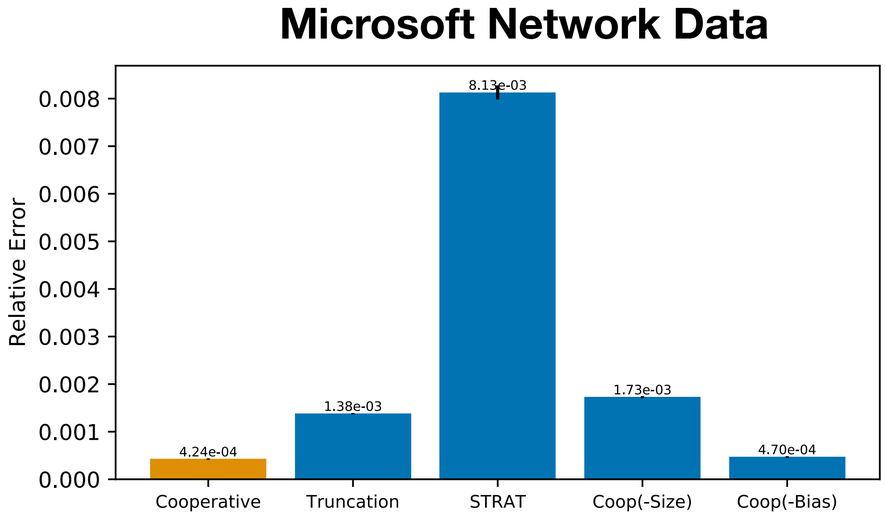
averaged over expected queries

After convex optimization, more populated segments can be allocated more space, make more aggressive bias trade-offs

Results: Cubes

Average frequency query error over random data cube queries





STRAT: [Chaudhuri et al. TODS 2007]

Outline and structure

Introduction

A. Motivating Context: ASP systems

B. Challenge: Query performance for high-cardinality analytics

Three new summaries to reduce performance bottlenecks:

1. Cooperative Summaries: Improved accuracy for quantiles and frequencies
2. Moments Sketches: Improved runtime for quantiles
3. [Briefly] TKDC: Fast and accurate kernel density classification

Discussion and Conclusion

At extremely high cardinality, existing quantile summaries are too slow

Quantile estimation over 1 million segment summaries:

k=1 million segments

Summary	Query Time
Mergeable [Agarwal et al. 2013]	4.5s
Cooperative	10s
Streaming [Karnin et al. FOCS 2017] over 1 billion records	2 minutes

Existing quantile summaries manipulate expensive sorted lists

Lower bound: accumulating segment sums and counts takes 20ms!

Frequency summaries are less expensive to merge [Anderson et al. IMC 2017]

Moments Sketch Goal: quantile summary with minimal merge time

Query Time has fixed and scaling costs $t_{query} = t_{estimation} + t_{merge} \cdot k$

Efficient query times for large k can only be achieved with low-overhead *merges*

Summary	Query Time
Mergeable	4.5s
Cooperative	10s
Streaming estimate over 1 billion records	2 minutes
Moments Sketch	**35 ms**

Moments Sketch addresses two types of query time overhead

Query Time has fixed and scaling costs $t_{query} = t_{estimation} + t_{merge} \cdot k$

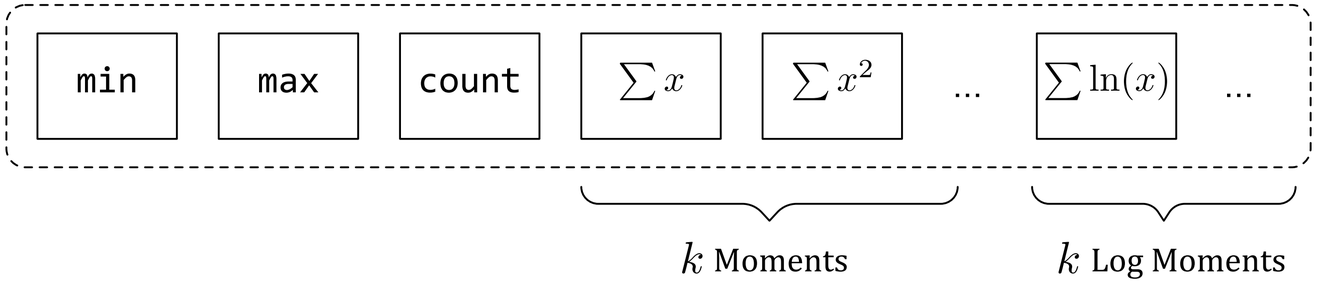
➡ 1. Data structures to minimize merge time t_{merge}

2. Optimized solvers to minimize estimation time $t_{estimation}$

Moments Sketch: moments as an efficiently mergeable summary

- Key Observation: scalar statistics can serve as a hyper-efficient sketch
- Moments and log-moments (e.g., mean, standard deviation) commonly used in statistics to estimate distribution shape
- Updates and merges: nanoseconds for floating point addition

Moments Sketch: Stores k moments and log-moments



Maximum entropy can estimate real-world distributions from moments

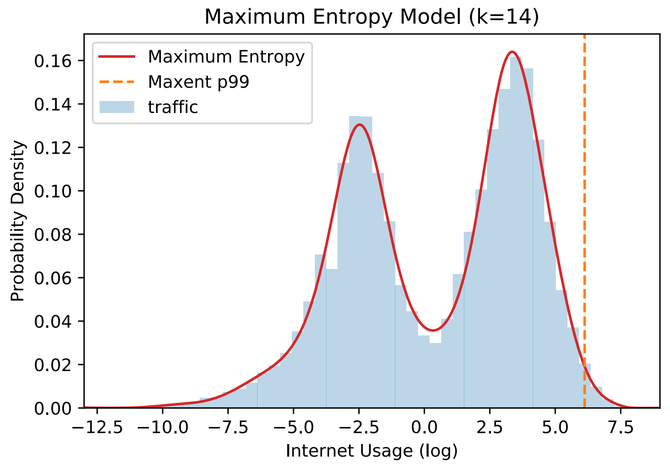
Maximum entropy principle: In the absence of additional information, the best estimate maximizes entropy *[Jaynes, Phys. Rev. 1957]*

$$\underset{f \in \mathcal{F}[x_{\min}, x_{\max}]}{\text{maximize}} \quad H[f] \longleftarrow \textbf{Entropy of distribution estimate}$$

$$\text{subject to} \int_{x_{\min}}^{x_{\max}} x^i f(x)\, dx = \mu_i, \qquad i \in \{1, \ldots, k_1\}$$

$$\int_{x_{\min}}^{x_{\max}} \log^i(x) f(x)\, dx = \nu_i, \qquad i \in \{1, \ldots, k_2\}$$

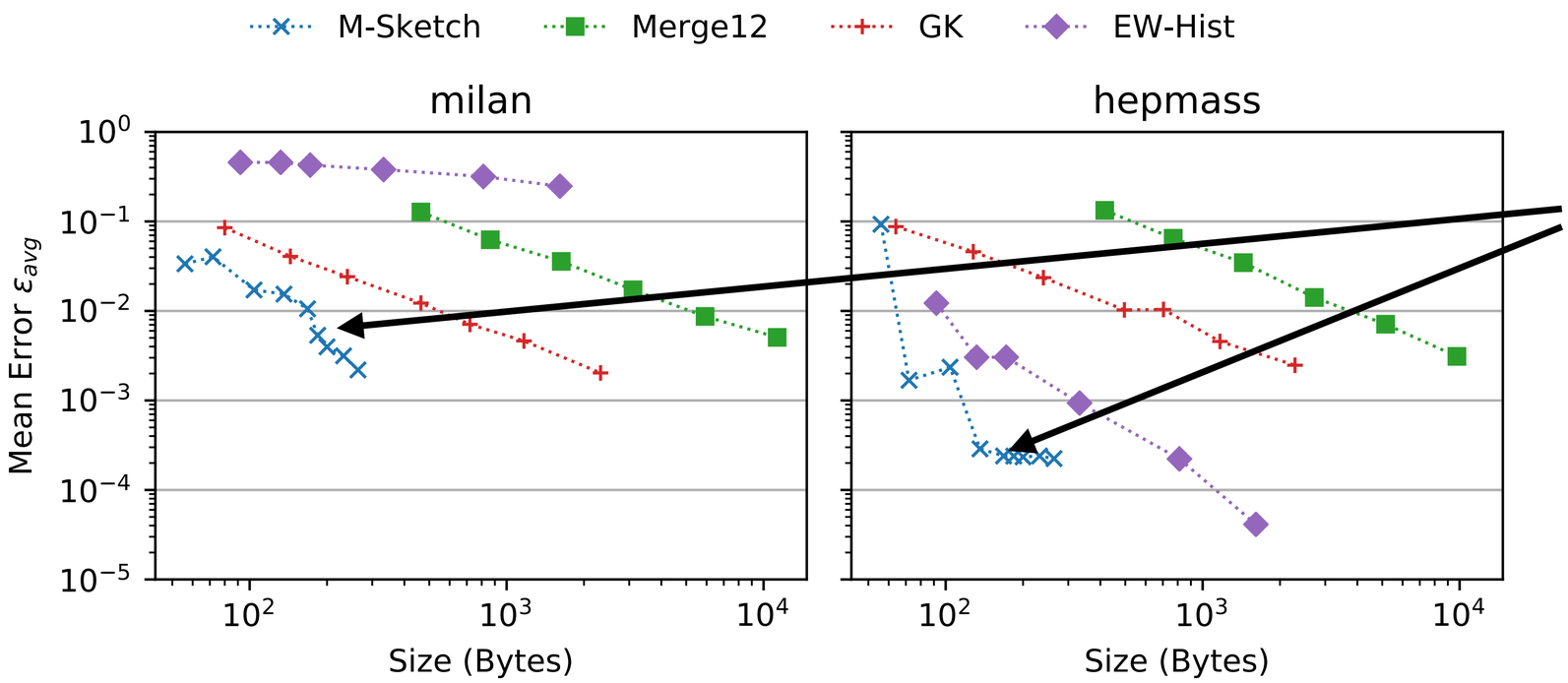
Error Upper bound: $\epsilon_{\text{avg}} \leq O\left(\frac{\hat{f}_{\max}}{k}\right)$



Observation: Maximum entropy fits real-world distributions, better than bounds

Results: Accuracy

Average single-segment query error for summaries of different sizes



Moments sketch achieves <1% error with k=10 moments, less space than existing summaries

Limitation: maximum accuracy defined by floating point precision past k=17

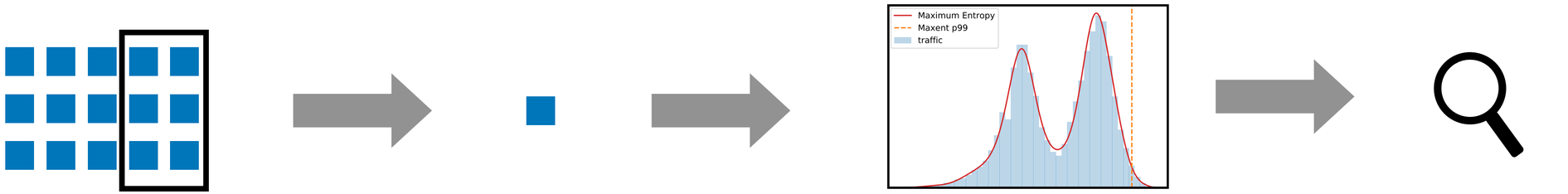
Merge12: Low-discrepancy summary [Agarwal et al. 2013], GK: [Greenwald & Khanna. 2001], EW-Hist: Mergeable Equal-Width histogram

Moments Sketch addresses two types of query time overhead

Query Time has fixed and scaling costs $t_{query} = t_{estimation} + t_{merge} \cdot k$

1. Data structures to minimize merge time
2. Optimized solvers to minimize estimation time

Maximum entropy fixed estimation costs are relatively high



Segment Summaries | **Merge Summaries** | **Entropy Optimization** | **Maximum Entropy Solution** | **Estimated Quantile**

Existing solvers are slow or unstable*: ECOS takes 300ms

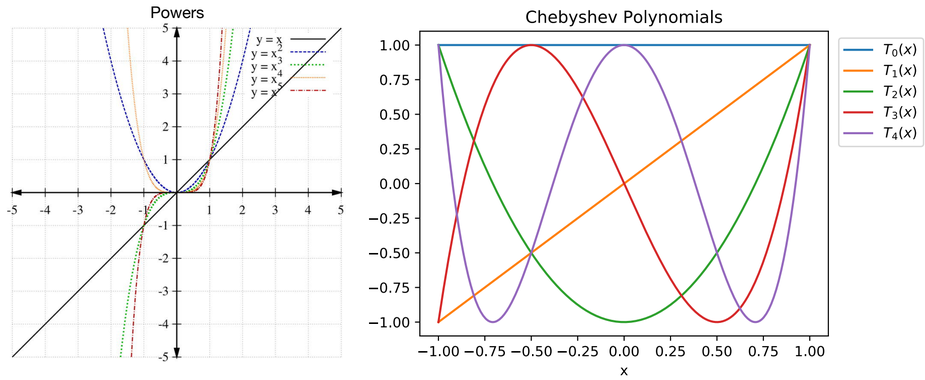
Merging the millions of moments sketches takes 30ms

**[Mead & Papanicolaou. J. Math. Phys. 1984], [Bandyopadhyay et al. Phys. Rev. 2005], ...*

Improving the maximum entropy solver to reduce query time

Components of a stable and efficient maximum entropy solver:

- Convert moments to Chebyshev basis for stability (pre-conditioning)
- Polynomial approximations for fast integration
- Dropping less-informative moments
- Early stopping using Markov bounds

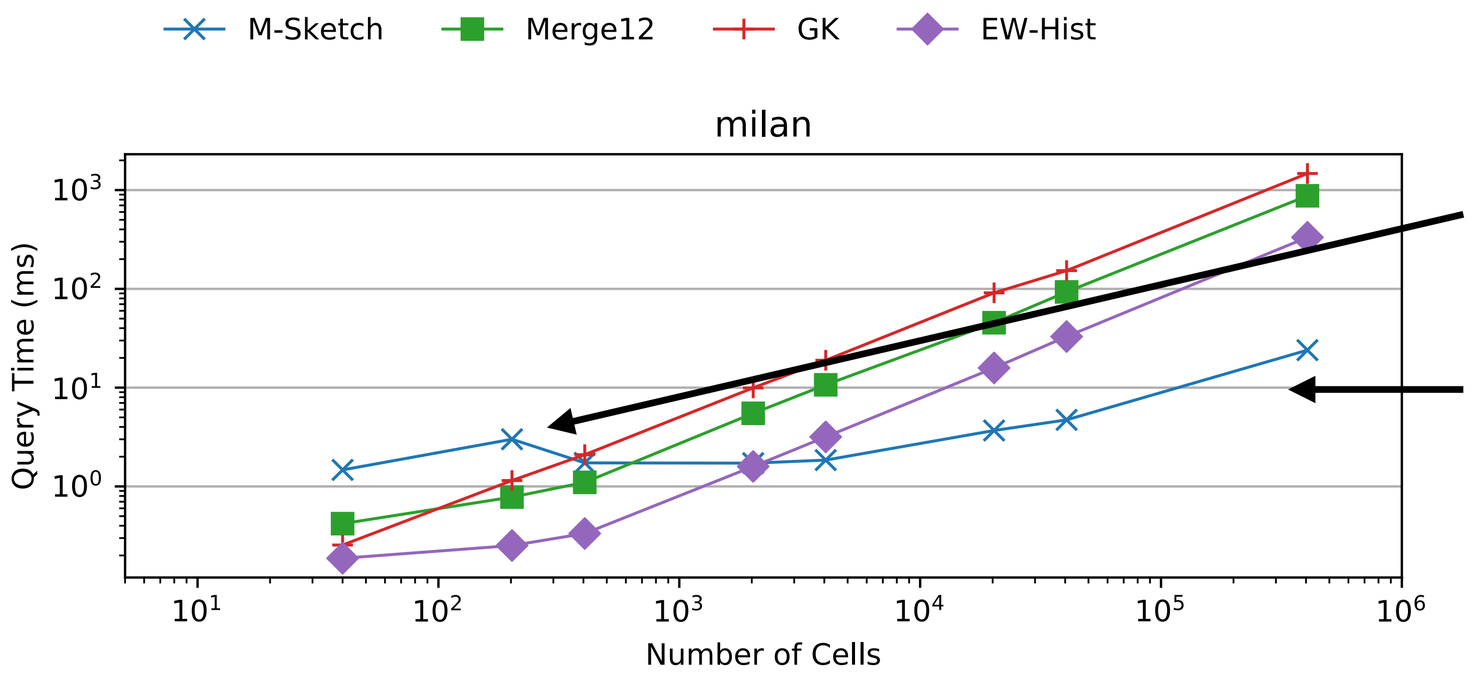


Solve time down to < 5ms

See paper for details

Results: Query Time

Average query error when aggregating multiple segments



Fixed solver time dominates when aggregating < 300 summaries

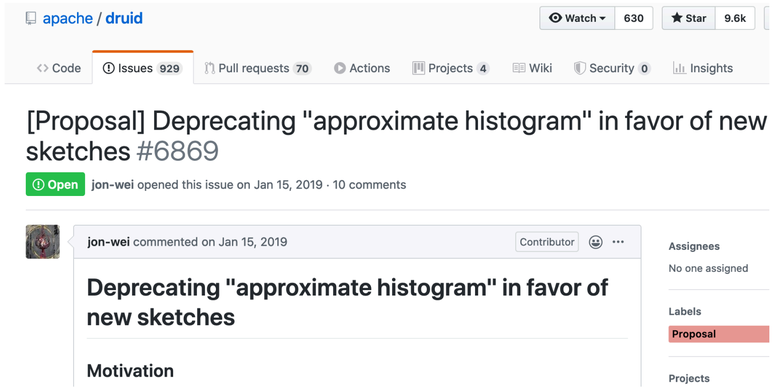
Low merge costs scale well when aggregating many summaries

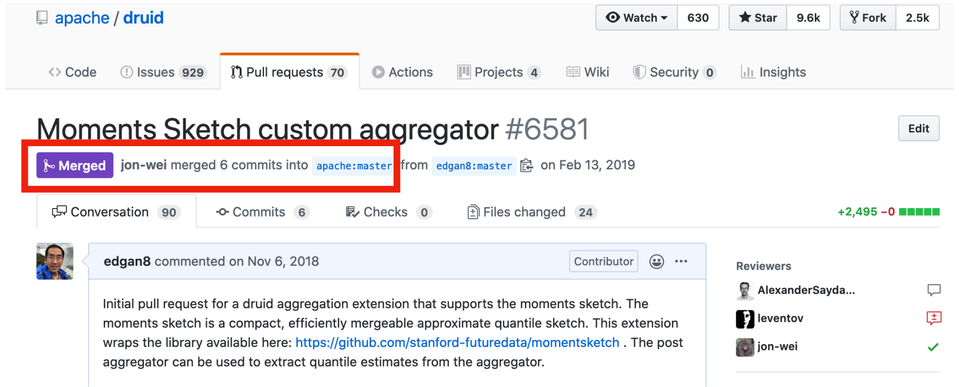
Merge12: Low-discrepancy summary [Agarwal et al. 2013], GK: [Greenwald & Khanna. 2001], EW-Hist: Mergeable Equal-Width histogram

Results: integration into real-world ASP systems

Integrated with

druid

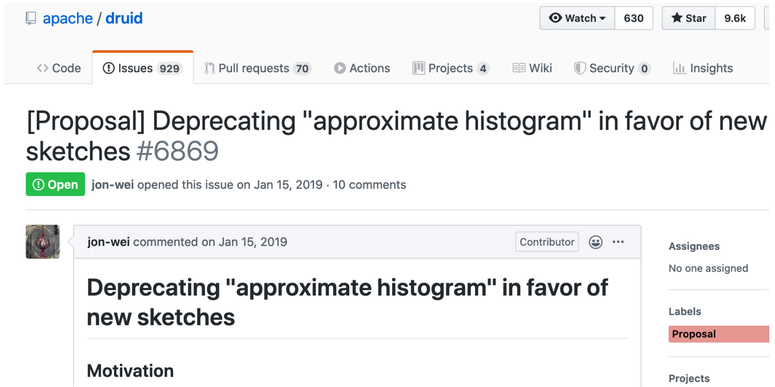


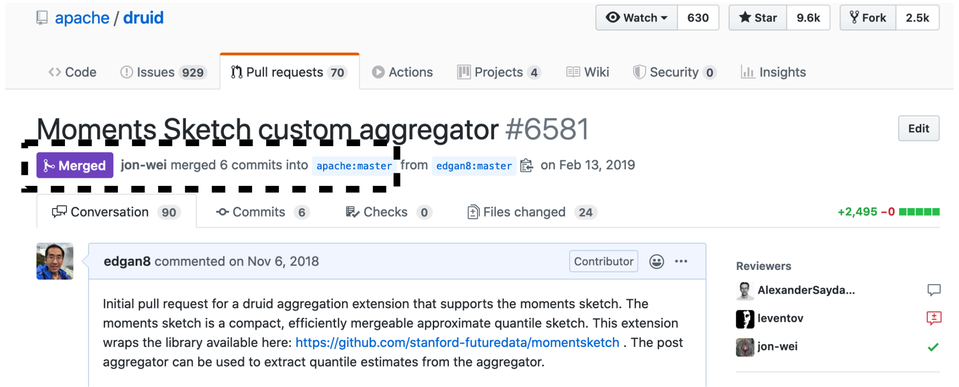


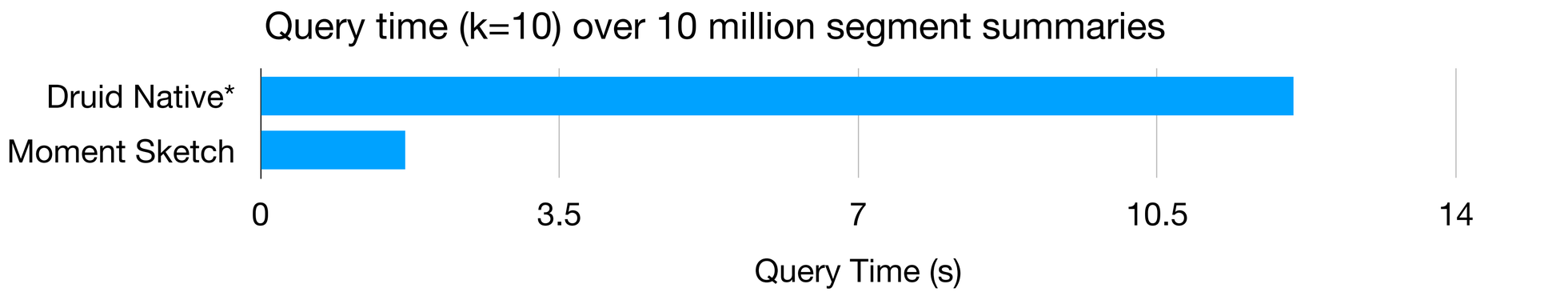
Results: integration into real-world ASP systems

Integrated with

druid







**Streaming Histogram sketch configured for 1% error from [Ben-Haim & Tom-Tov. JMLR 2010]*

Outline and structure

Introduction

A. Motivating Context: ASP systems

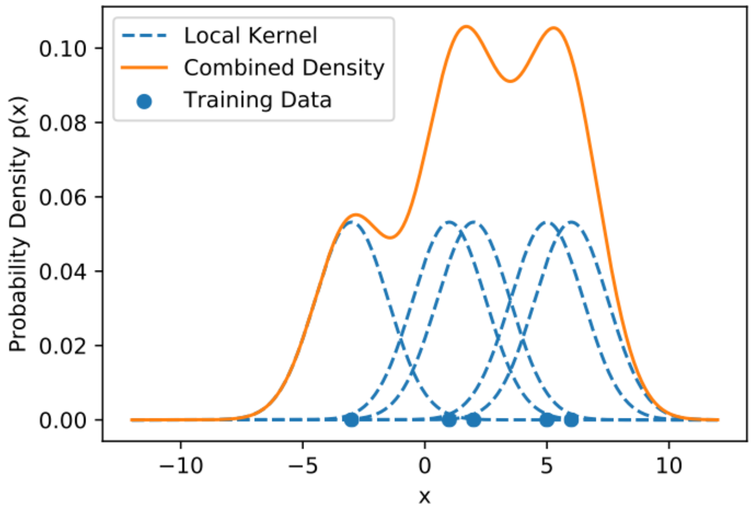
B. Challenge: Query performance for high-cardinality analytics

Three new summaries to reduce performance bottlenecks:

1. Cooperative Summaries: Improved accuracy for quantiles and frequencies
2. Moments Sketches: Improved runtime for quantiles
3. [Briefly] TKDC: Fast and accurate kernel density classification

Discussion and Conclusion

Kernel density estimates are a powerful but expensive distribution model



Kernel Density Estimates over $\mathbb{R}^d$

$$f(x_q) = \sum_{x_i \in \text{Data}} K(x_q - x_i)$$

Accumulates kernels from all points in dataset

Commonly used for **classification** (KDC): useful for clustering, outlier detection

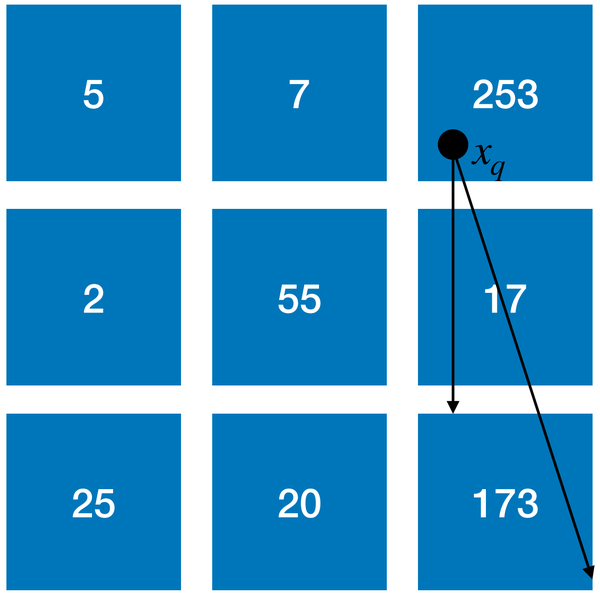
$$f_c(x_q) = \begin{cases} \text{Hi} & f(x_q) \geq t \\ \text{Low} & f(x_q) < t \end{cases}$$

Slow: 2 hours to compute on all 1 million points on a 2.9Ghz cpu

Precomputed region summaries can approximate kernel densities

Can summarize regions using counts and bounding boxes

[Wand & Jones. 1995, Gray & Moore. ICDM 2003]



$$f(x_q) = \sum_{x_i \in \text{Data}} K(x_q - x_i) \approx \sum_{S \in \text{Segments}} K(x_q - x_S) \cdot n_S$$

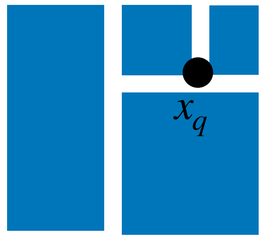
Approximate and bound $f(x_q)$ using counts and distance to regions

High-cardinality: Need exponentially many segments in high dimensions

TKDC: optimized hierarchical regions reduce size of aggregations

TKDC summaries are **hierarchical** regions that are accumulated to prioritize **precise classification** with minimal number of segments

Some classification queries need more precision than others

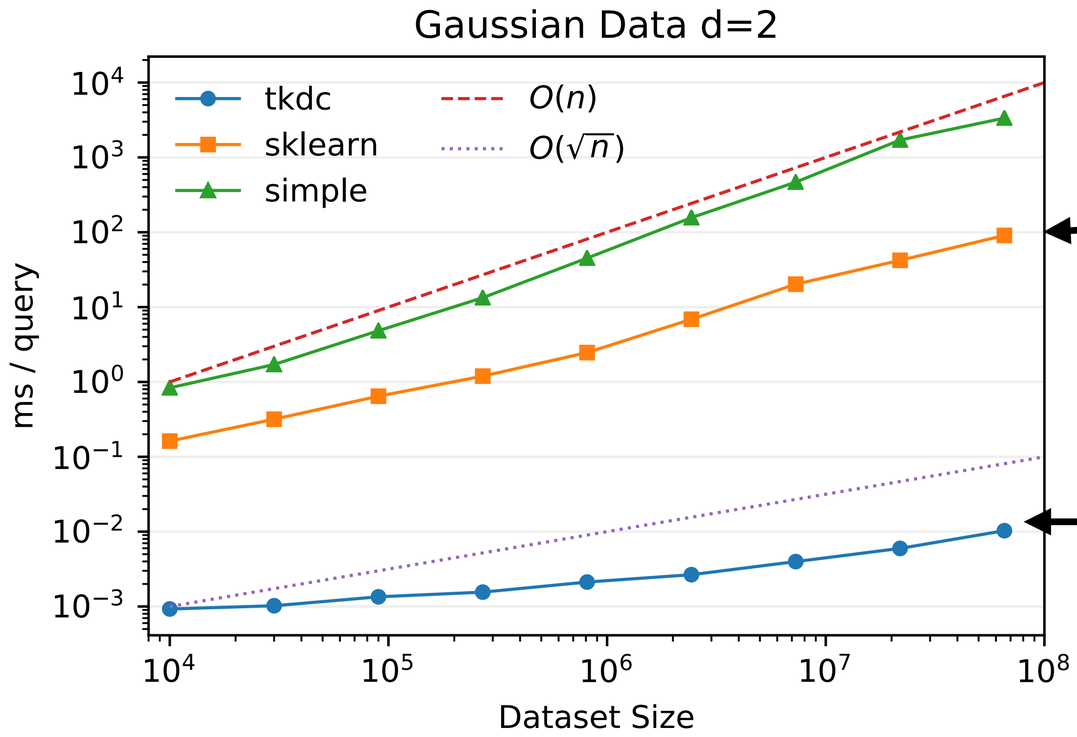


Difficult classification: accumulate finer segments

Simple classification: use fewer coarse segments

On average, number of segments accumulated reduced to $O(n^{\frac{d-1}{d}})$

TKDC prioritizes hierarchical segments to accelerate classification



Sci-kit learn uses region summaries without classification cutoff optimization

Runtime scaling is less than $O(n^{\frac{d-1}{d}})$ bound

Outline and structure

Introduction

A. Motivating Context: ASP systems

B. Challenge: Query performance for high-cardinality analytics

Three new summaries to reduce performance bottlenecks:

1. Cooperative Summaries: Improved accuracy for quantiles and frequencies
2. Moments Sketches: Improved runtime for quantiles
3. [Briefly] TKDC: Fast and accurate kernel density classification

➡ Discussion and Conclusion

Discussion: Themes and Future Work

1) Cooperative Summaries **2) Moments Sketches** **3) TKDC**

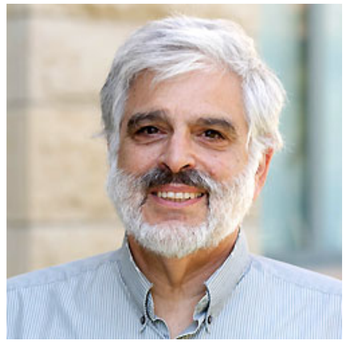
Improved scalability to high-cardinality aggregations requires trade-offs:

- Reduced accuracy or query time at low-cardinalities
- Specialization to common but specific query types

Open Question: Can we develop a framework to navigate these trade-offs?

Acknowledgements: Orals Committee

John Duchi



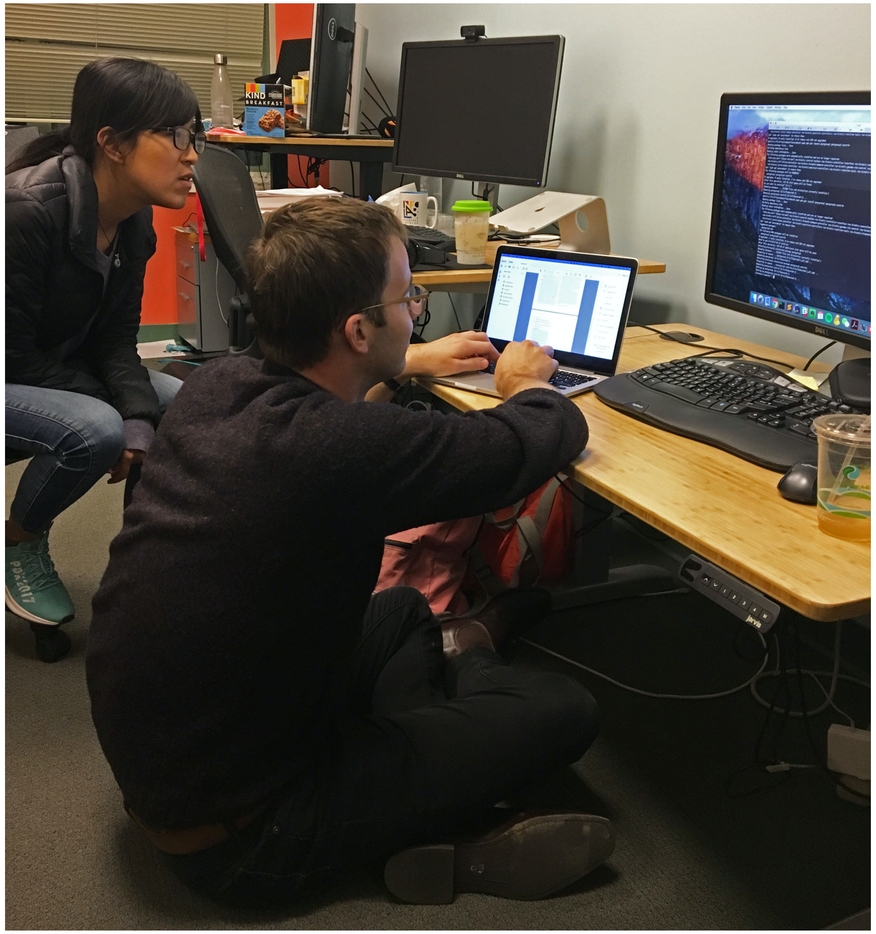
Jeff Ullman

Matei Zaharia

Moses Charikar

Acknowledgements: PhD Advisor

Peter Bailis



Acknowledgements

Industry Collaborators:
Atul Shenoy, Will Mortl, Jon Wei, Gian Merino

Research Collaborators:
Christopher Aberger, Firas Abuzaid, Daniel Kang, Jialin Deng, Justin Chen, Kai Sheng Tai, Kexin Rong, Sahaana Suri, Todd Warszawski, Vatsal Sharan

Senior Researchers:
Greg Valiant, Chris Re, Tatsunori Hashimoto, Percy Liang

Friends & Family:
Hao Gan, Wei Kang, Cynthia Gan, many others!

Summary

Novel summaries tailored for high-cardinality workloads can provide **end to end improvements** in accuracy, space usage, and runtime.

1) Cooperative Summaries

Accuracy + Space

by exploiting common selection patterns

2) Moments Sketches

Runtime + Space

by leveraging efficient statistical estimators

3) TKDC

Runtime + Accuracy

by prioritizing resolution of segments