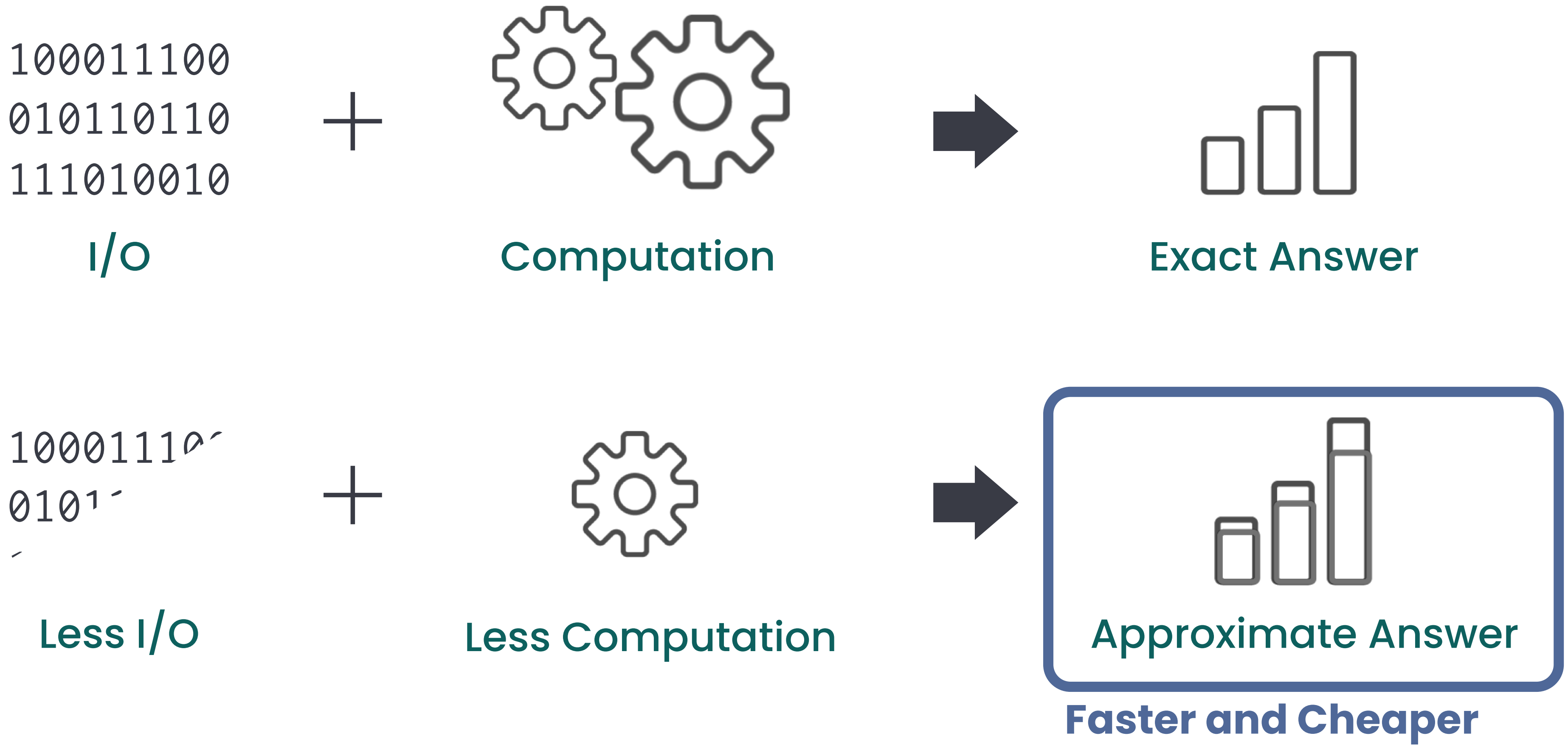
Approximation is Bliss

Approximate Computing in Database Systems

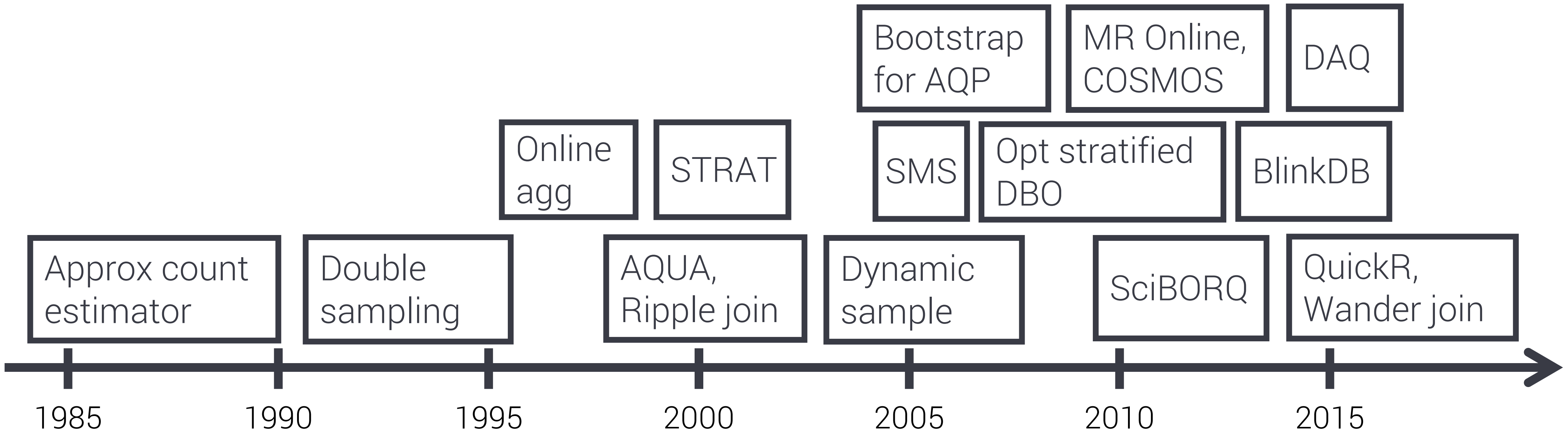
Yongjoo Park @Michigan

Approximate query processing is becoming more valuable

What is approximate query processing (AQP)?

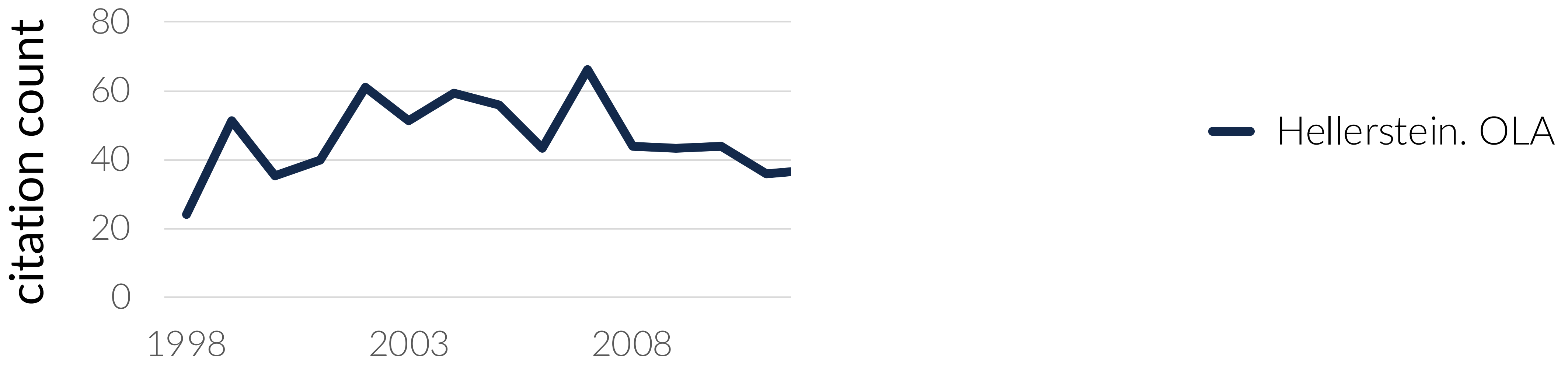


AQP research has a long history

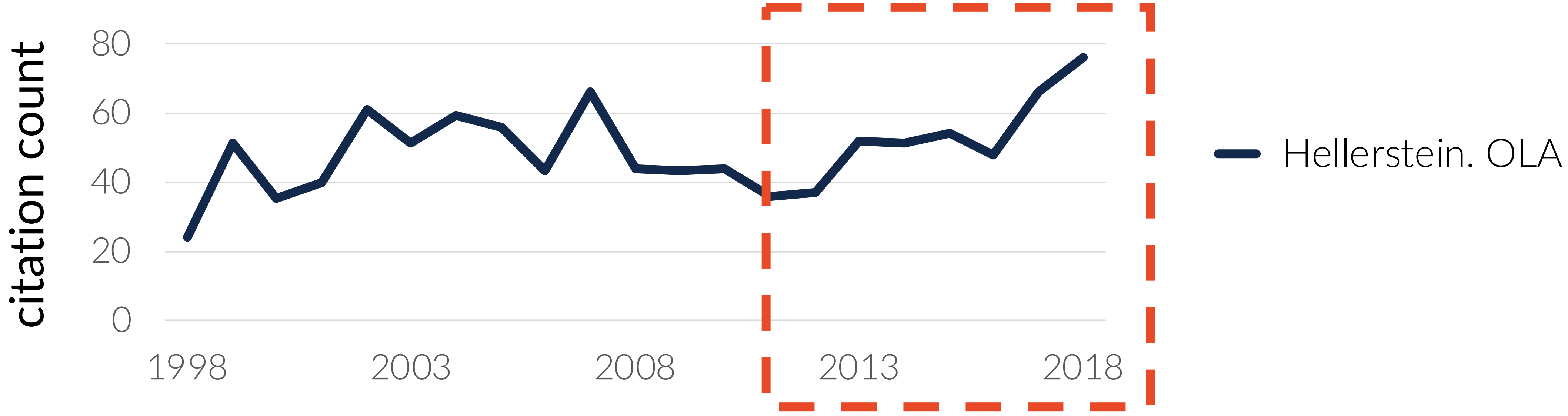


35 years of research

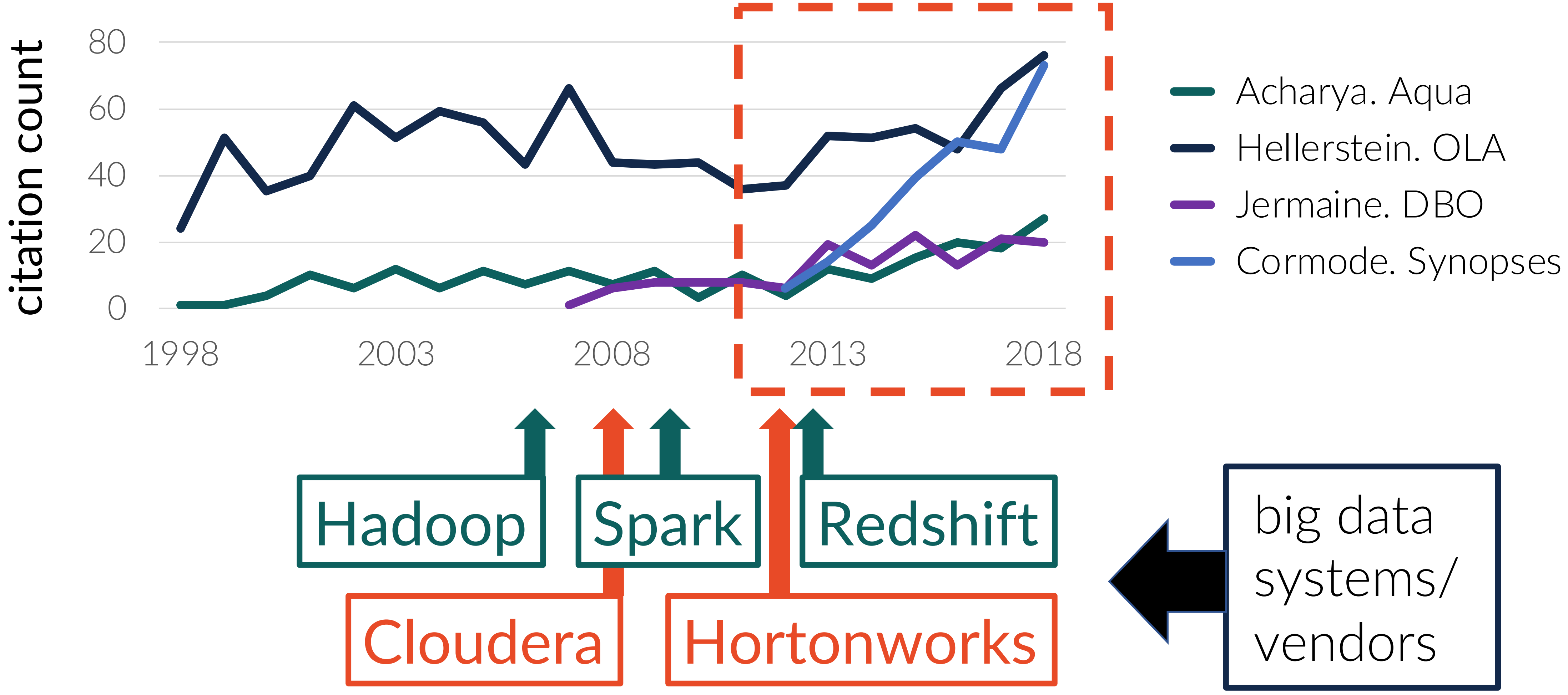
Resurgence of AQP research



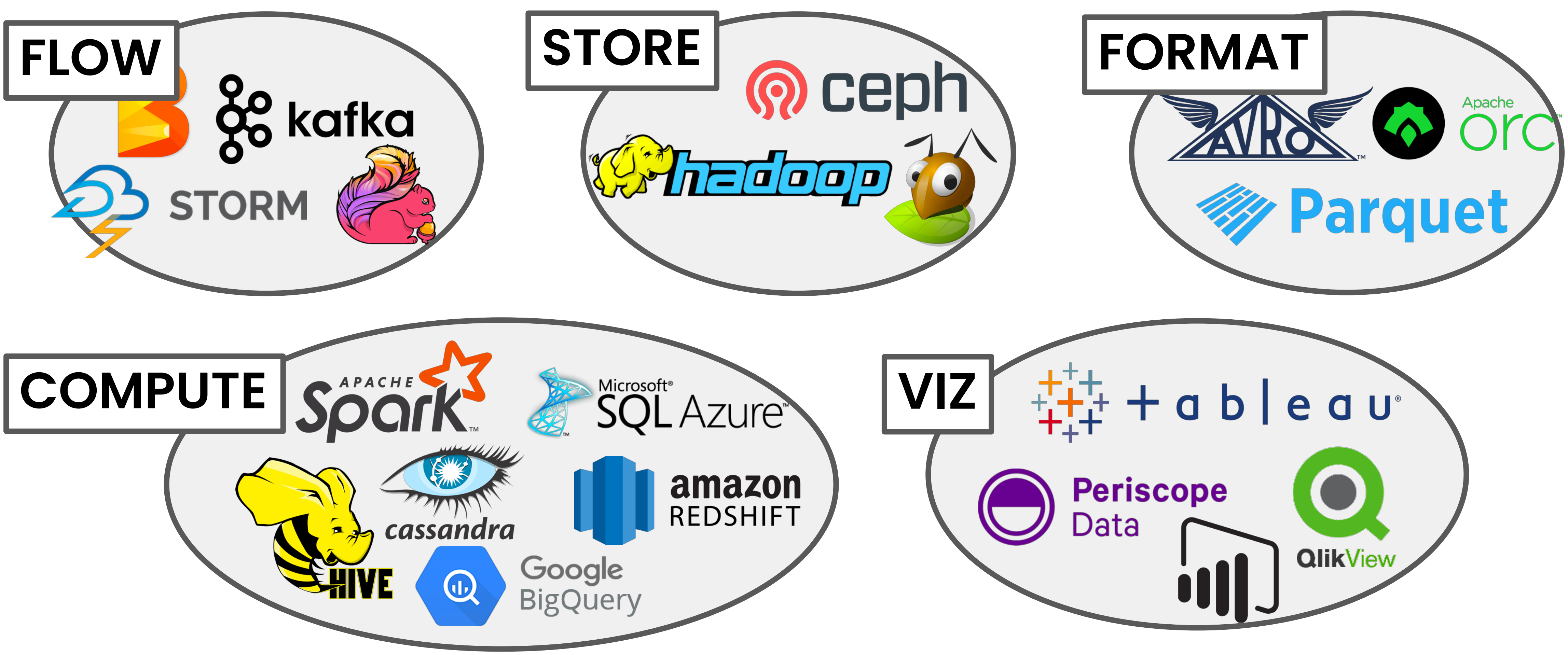
Resurgence of AQP research



Resurgence of AQP research



Today’s big data ecosystems



Today’s big data ecosystems

- Can process a large volume of data
- Slow (esp. for ad-hoc queries)
- Costly

Big data analytics is slow

Walmart

One of the largest retail corporations

Collects 70GB+ data/day

Ad-hoc queries with customer demographic filters

dunnhumby

One of the biggest customer science company in UK

Basic statistics + ML

LOCALLY

A location intelligence company

Billions of GPS points

Real-time responses required for its web-interface

Using commercial clusters (from MapR, Amazon, …)

10-20 minute query latencies

Big data analytics is costly

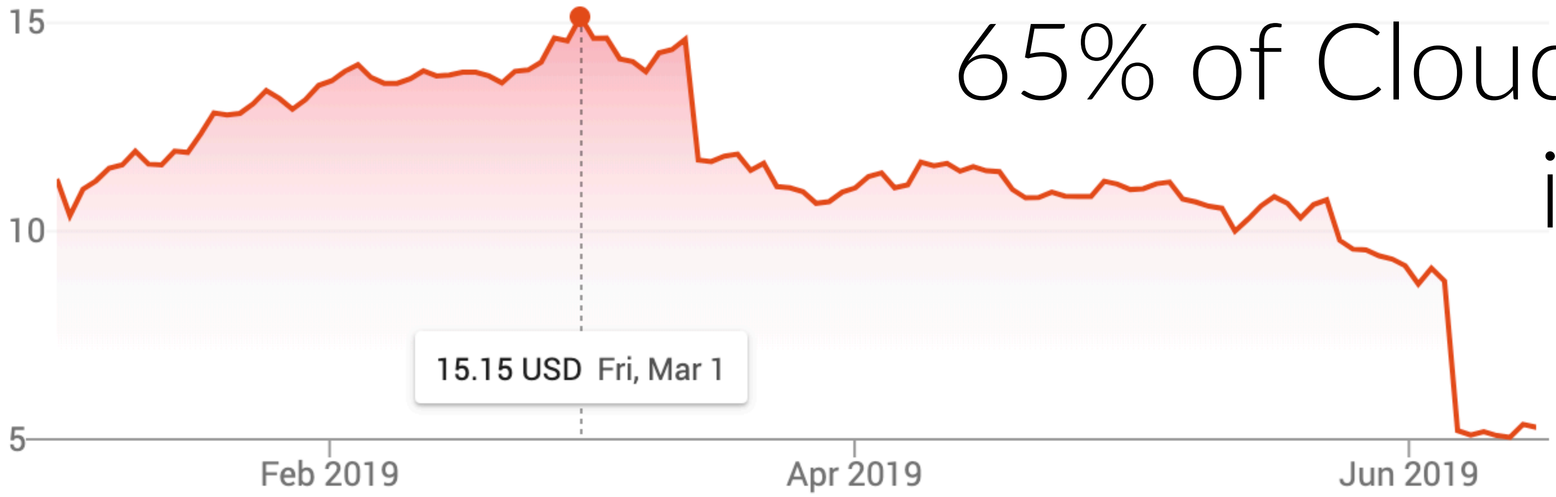
Case 80 GB/day, one-year data retention, 1000 queries/day

pay per node	amazon REDSHIFT	$48K/month
pay per query	Google BigQuery, Amazon Athena	$450K/month

HIGH COST

The cost increases with more data & queries

Big data: too much cost for its value?



65% of Cloudera is gone in 4 months

"We generate woefully **low amounts of value** relative to the amount spent."

Jesse Anderson, Director of Big Data Institute

https://www.jesse-anderson.com/2019/06/i-come-not-to-bury-cloudera-but-to-praise-it/

Approximation is bliss

100x faster or cheaper
by sacrificing 0.1% accuracy

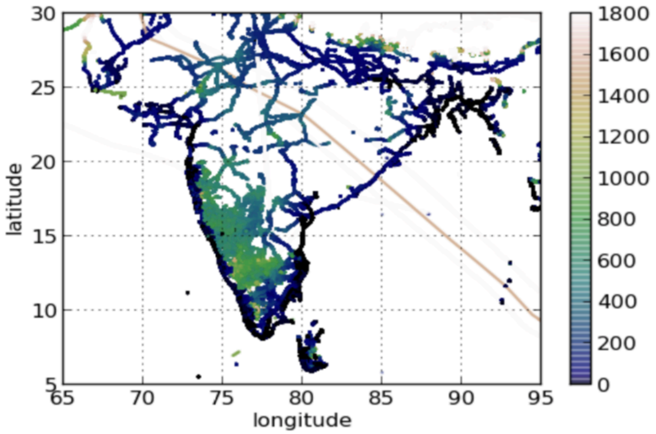
$$\boxed{err = f\left(\frac{1}{n} - \frac{1}{N}\right) \leq f\left(\frac{1}{n}\right)}$$

Faster: less I/O, less computation

Cheaper: same latency with less resource

AQP can produce indistinguishable results

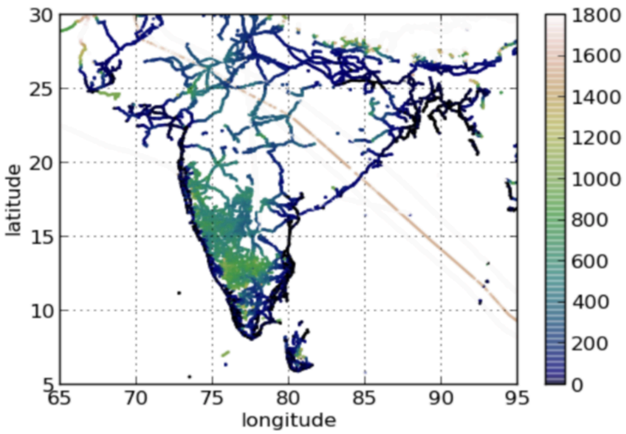
EXACT



2 billion points

Took **71 mins**

APPROXIMATE



1 million points

Took **3 secs**

[Park et al. ICDE'16]

Our contributions

1. 35 years of research, **little** industry adoption

   Our effort: Universal AQP [Park et al. SIGMOD'18]

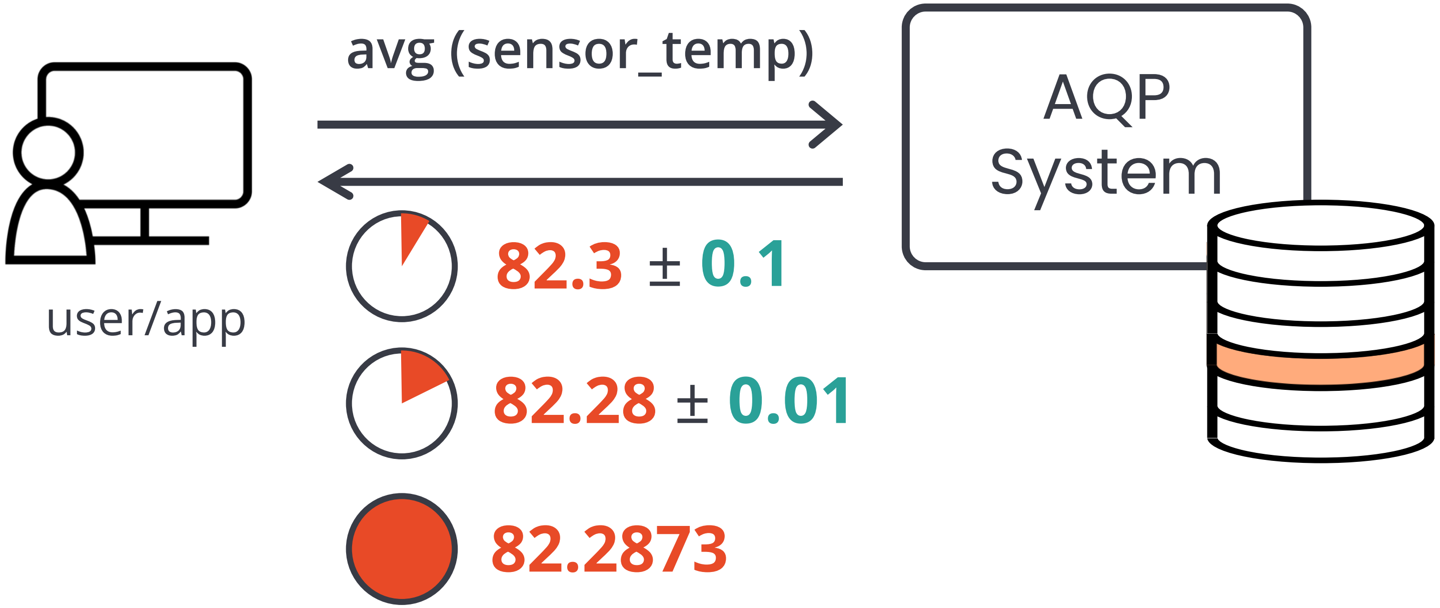
   sum, avg, count, count-distinct

2. **Limited** to simple aggregation

   Our effort: AQP for ML [Park et al. SIGMOD'19]

<Universal AQP>...

Typical AQP systems



[Aqua '99, JoinSynopses '99, BlinkDB '13, WanderJoin '16, Quickr '16, and more]

Barriers in adopting AQP

Vendor resistance: AQP requires significant changes to DBMS internals

- Traditional DBMS: stable codebases, reluctance to major changes
- Newer SQL-on-Hadoop: busy catching up on standard features

User resistance:

- users don't typically abandon their existing systems
- vendor lock-in makes data migration almost impossible

[BlinkDB '13, G-OLA '15, Join Synopses '99, WanderJoin ,16, ABM '14, ...]

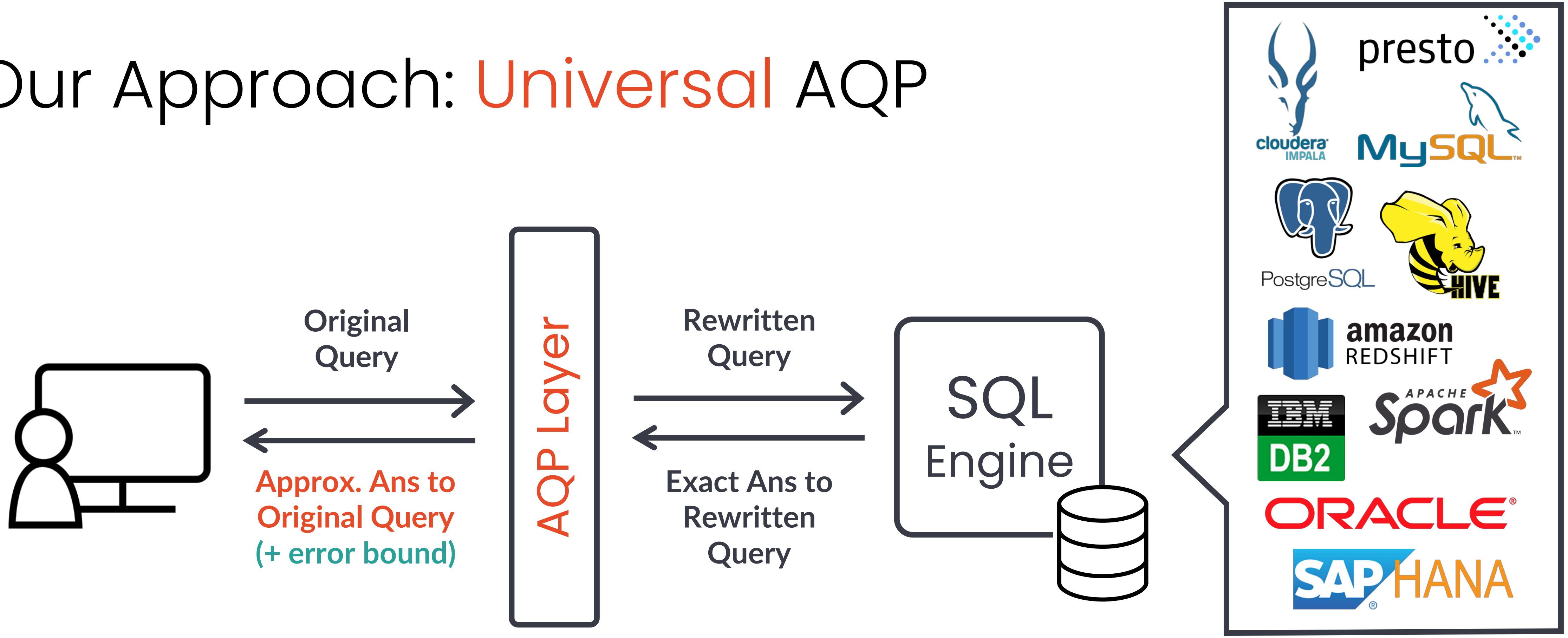
One example of user resistance

"Since our organization is huge, it would be **difficult** to ask infrastructure team to **apply patch on spark** for supporting BlinkDB."

Giridhar Addepalli, Walmart

https://github.com/sameeragarwal/blinkdb/issues/14

Our Approach: Universal AQP



The rewritten query runs faster because it uses a **sample table** instead of the **original table**

Universal AQP: criteria

	VerdictDB https://github.com/mozafari/verdictdb
Consistency	We focus on append-only data
Accuracy guarantee	New error estimation logic
Efficiency	Comparable to built-in AQP

VerdictDB: offers large speedups

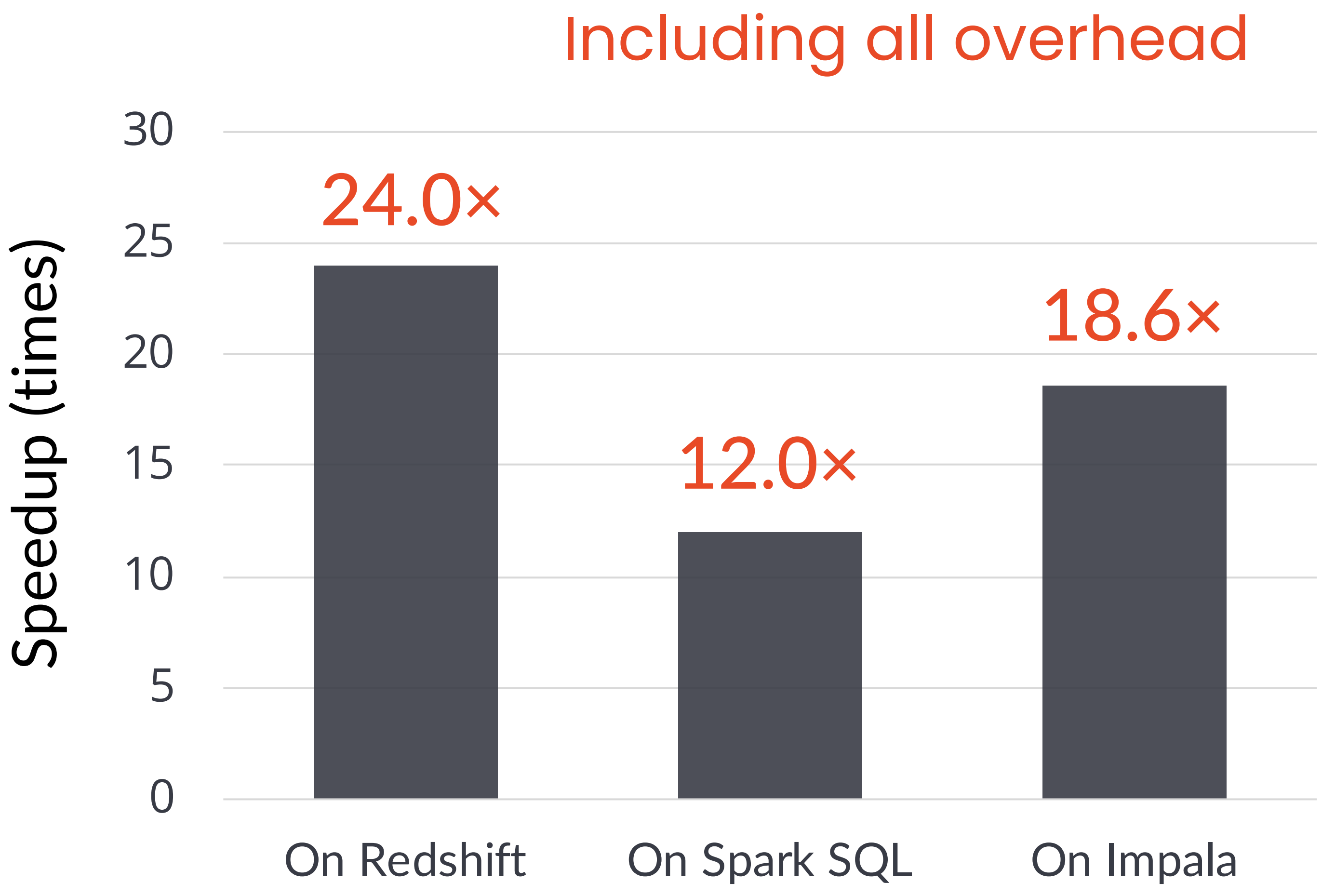
Datasets:
- 500GB TPC-H benchmark
- 200GB Instacart dataset

Workloads:
- TPC-H, microbenchmark

Data Systems:
- Redshift, Spark SQL, Impala
- 10+1 r4.xlarge cluster

Used columnar formats for all systems

2% relative errors



VerdictDB: comparable to built-in AQP

Datasets:
- 200GB Instacart dataset

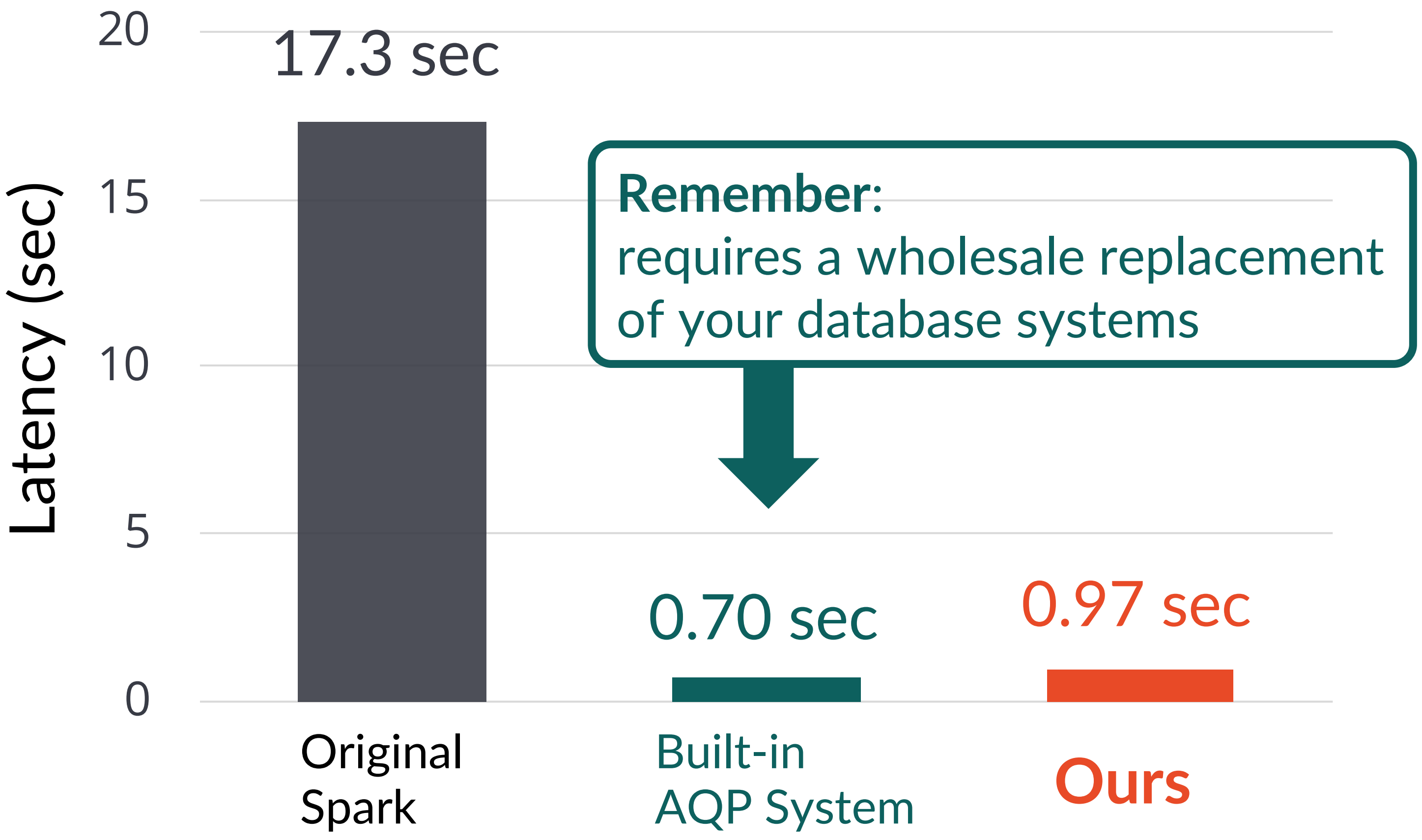
Workloads:
- microbenchmark

Data Systems:
- Spark SQL
- 10+1 r4.xlarge cluster

Built-in AQP System:

SnappyData

(a commercial version of BlinkDB)



jupyter VerdictDB on Presto Last Checkpoint: an hour ago (unsaved changes)

Logout

File Edit View Insert Cell Kernel Widgets Help

Trusted Python 3

Run Code

Sales by Aisle

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

...</Universal AQP>

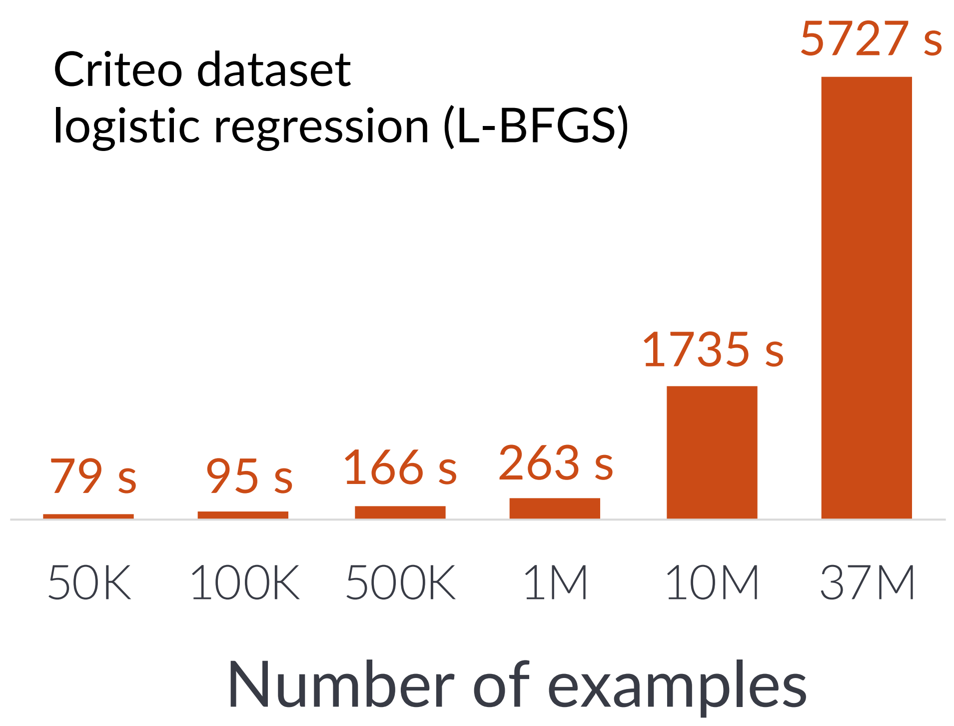
<AQP for ML>...

Machine learning can be time-consuming

More data, slower training

We train multiple models

- new training data
- feature engineering [Anderson, CIDR'13]



Sampling may accelerate training

Training: iterative gradient computation

$$\theta_{t+1} = \theta_t - \alpha \cdot grad(\theta_t) \qquad \text{(until convergence)}$$

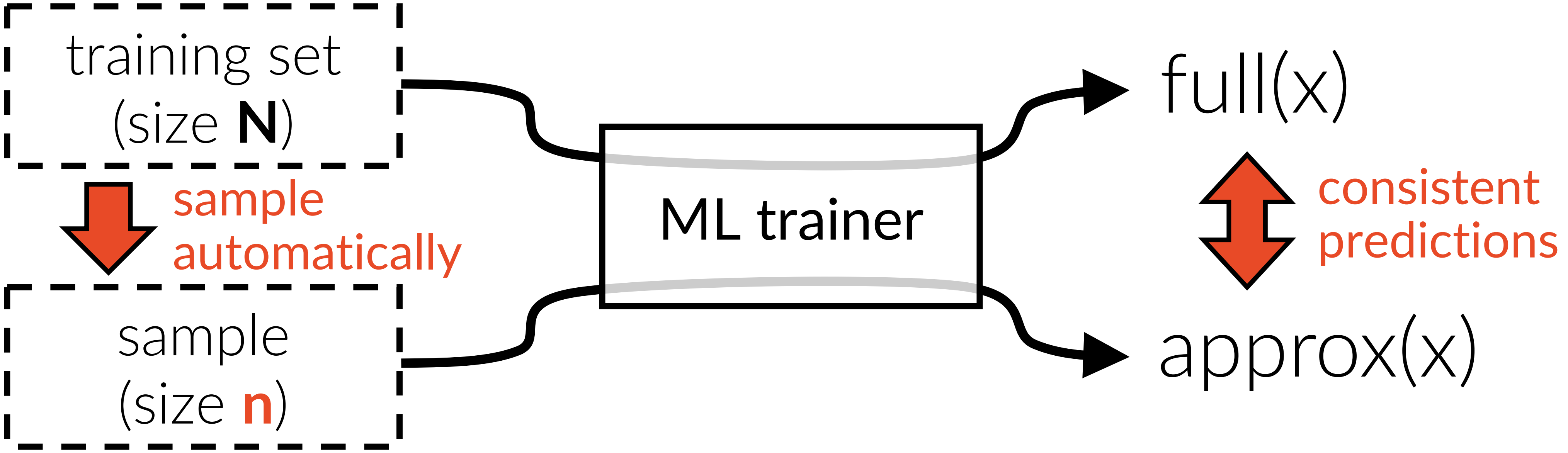
Sampling: gradient computation becomes faster

$$grad(\theta_t) = (1/N) \sum_{i=1..N} f(x_i \mid \theta_t)$$

Benefits if (savings from sampling) > (increase in # of iterations)

1. Ad-hoc approach: no accuracy guarantee
2. Biased sampling: not efficient for feature engineering

BlinkML: uniform sampling w/ accuracy guarantee



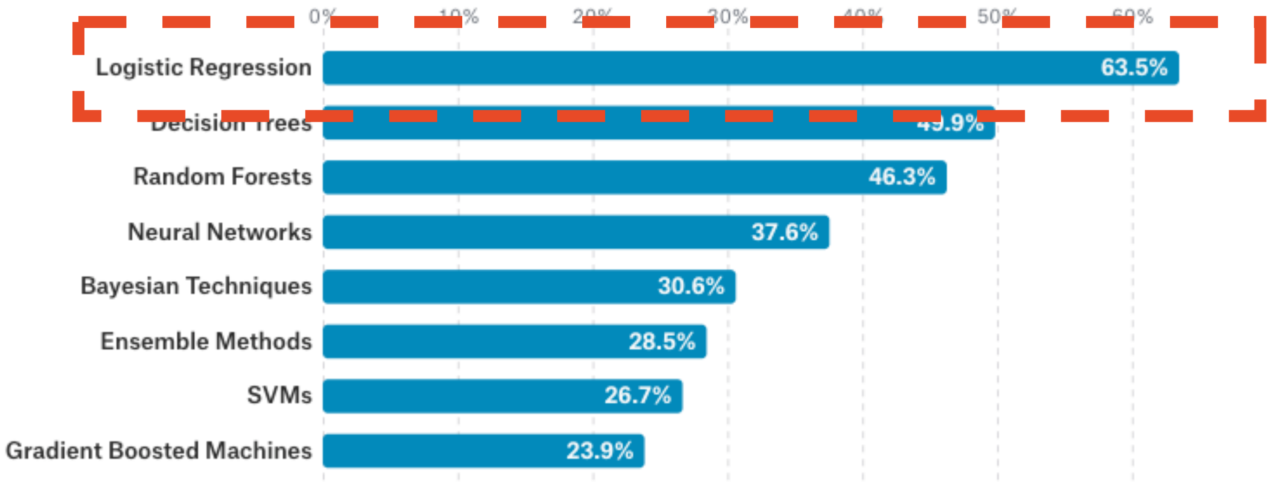
Consistency is expressed as

$E_x[\text{full}(x) \neq \text{approx}(x)] < \varepsilon$ with high probability

BlinkML supports commonly-used models

Supports convex MLE models:

- linear regression
- logistic regression
- probabilistic PCA
- generalized linear models



https://www.kaggle.com/surveys/2017/

Accuracy guarantee exploits the property of MLE models:

$$\mathrm{grad}(\theta_{opt}) = (1/N) \sum_{i=1..N} f(x_i \mid \theta_{opt}) = 0$$

BlinkML introduces computational optimization

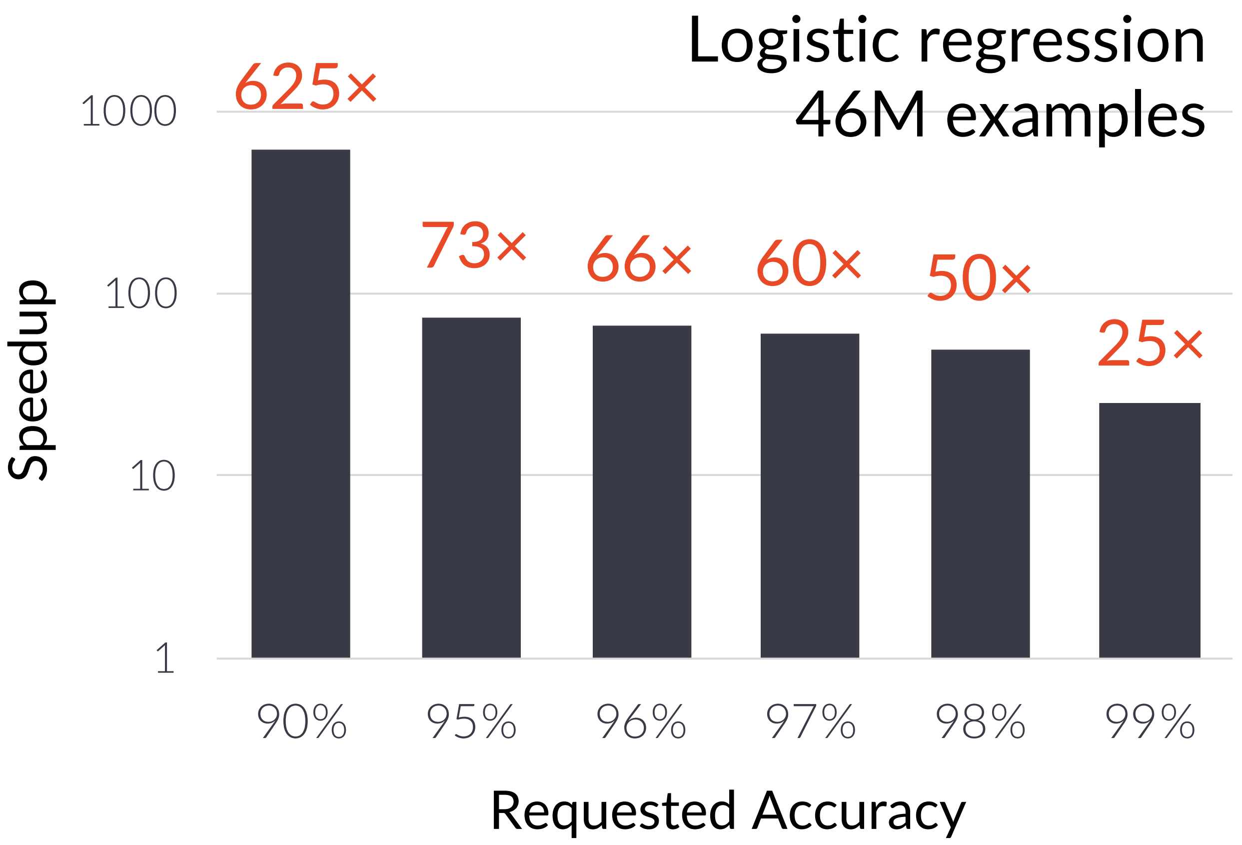
BlinkML offers large speedups

Datasets:

- Size: 2.86 GB on disk
- # of features: 998K

Systems:

- Optimization: Scipy
- 5+1 m5.2xlarge



...</AQP for ML>

Summary

1. **AQP:** becoming more valuable
2. **VerdictDB:** enables AQP on any platforms
3. **BlinkML:** trains MLE models with bounded errors

Thank you!