Apache® DataSketches™

A Production Quality Sketching Library for the Analysis of Big Data

Lee Rhodes
PMC Chair, Apache DataSketches

Outline

Problematic Queries of Big Data

Where traditional analysis methods don’t work well

Big Data Analysis Using Sketches

How using stochastic processes and probabilistic analysis wins in a systems architecture context

Our Mission, Sketch Design, Research

The sketch design process

Our collaboration with scientists around the world

The Open Source Apache DataSketches Library

A quick overview of this unique library dedicated to production systems that process big data.

Big Data is…big…and growing

IDC Global Datasphere Whitepaper, published Nov 2018,

(new data created each year)

- 2018**: 33 Zetabytes***
- 2025**: 175 Zetabytes,** CAGR = 27%
- **479 Exebytes*** created daily.
- **150B** connected devices, most of which are creating data

Fortune Business Insights (2024):

- In 2023, the global big data & business analytics market was **$307.5B** and is projected to reach **$924.4B** by 2032. CAGR = 13.0%.

* Zeta =10^{21}; Exe = 10^{18}; Peta = 10^{15}; Tera = 10^{12}; Giga = B = 10^{9}; Mega = 10^{6}; Kilo = 10^{3}

The Data Analysis Challenge ...

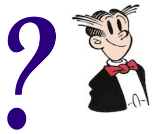
Example: Web Site Logs

Time Stamp	User ID	Device ID	Site	Time Spent Sec	Items Viewed
9:00 AM	U1	D1	Apps	59	5
9:30 AM	U2	D2	Apps	179	15
10:00 AM	U3	D3	Music	29	3
1:00 PM	U1	D4	Music	89	10
Billions of Rows or *K,V* Pairs …					

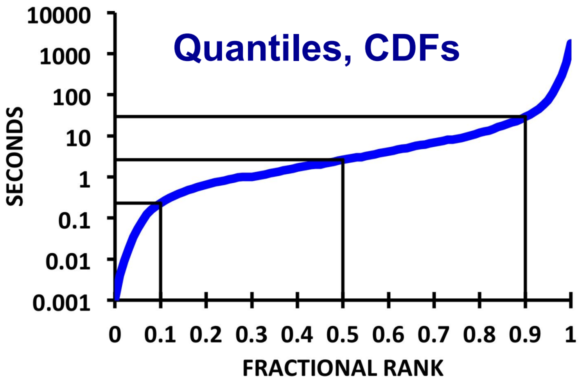
… Analyze This Data In Near-Real Time.

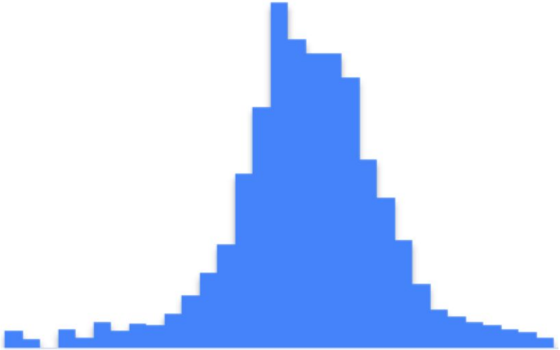
Some Very Common, but Problematic, Queries …

Unique Identifiers
with Set Expressions:
(A∪B)∩(C∪D) \ E

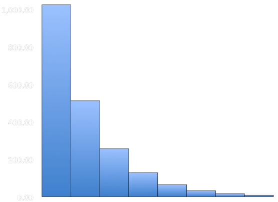


Set
Membership





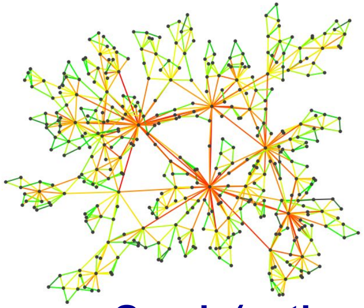
Histograms, PMFs



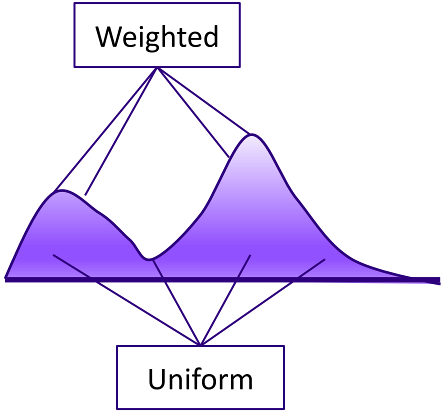
Frequent Items /
Heavy Hitters /
"Top-N"

$$\begin{Bmatrix}5 & \cdots & 2\\ \vdots & \ddots & \vdots\\ 4 & \cdots & 3\end{Bmatrix}$$

Vector & Matrix
Operations:
SVD, Density,
etc.



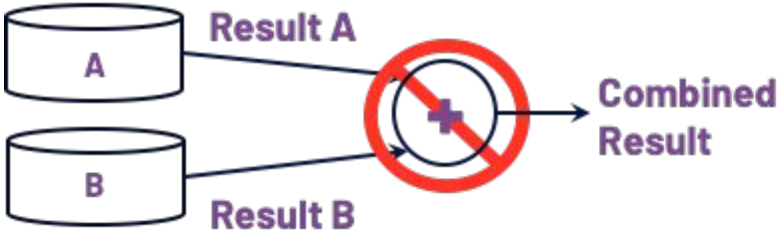
Graph (path,
funnel) Analysis



Reservoir Sampling

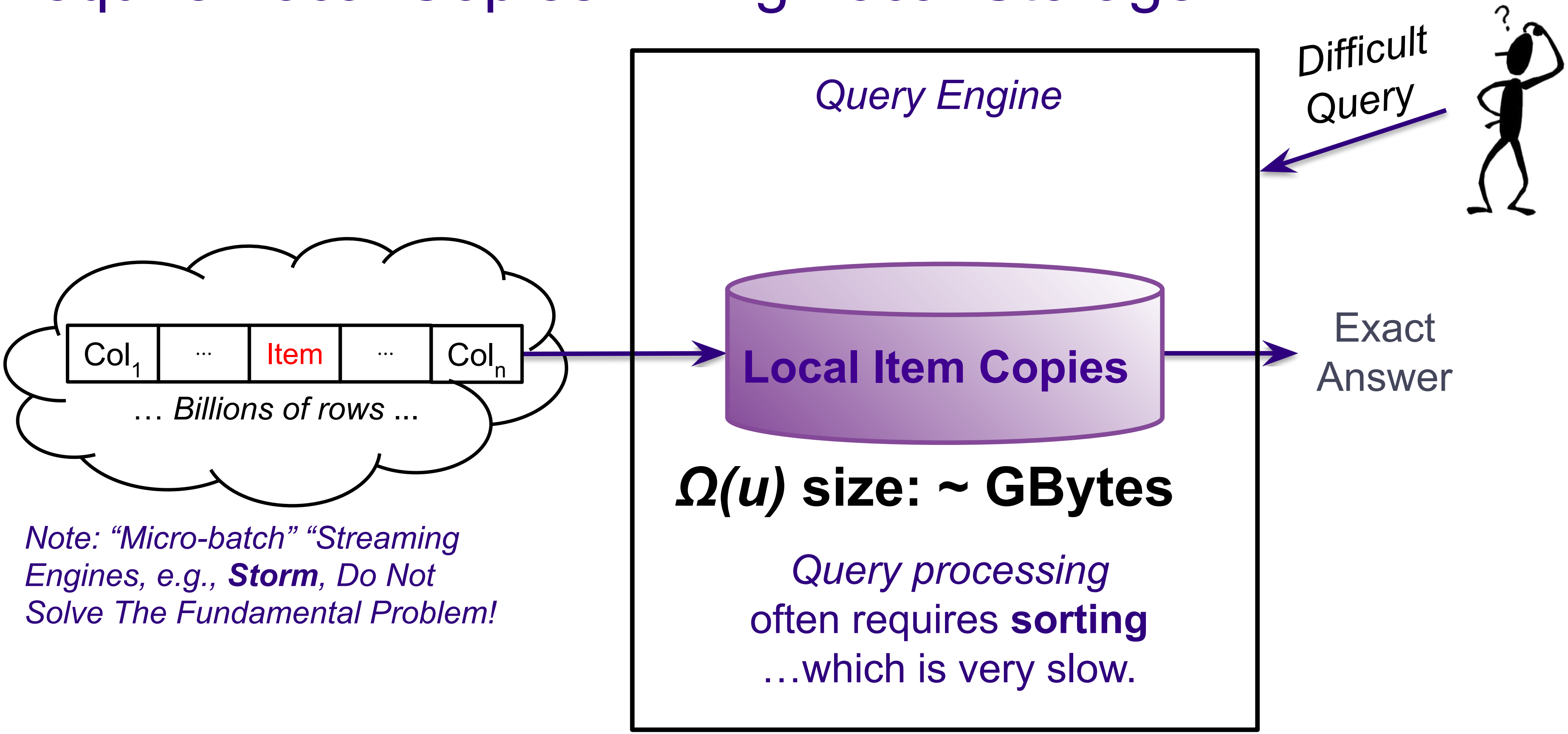
All Of These Queries Are Problematic:

- The Analysis Operations are *Non-Additive* or *Non-Linear*



- Addressing these queries with traditional methods requires multiple touches of the data.
- Time to process and resources required gets worse as the

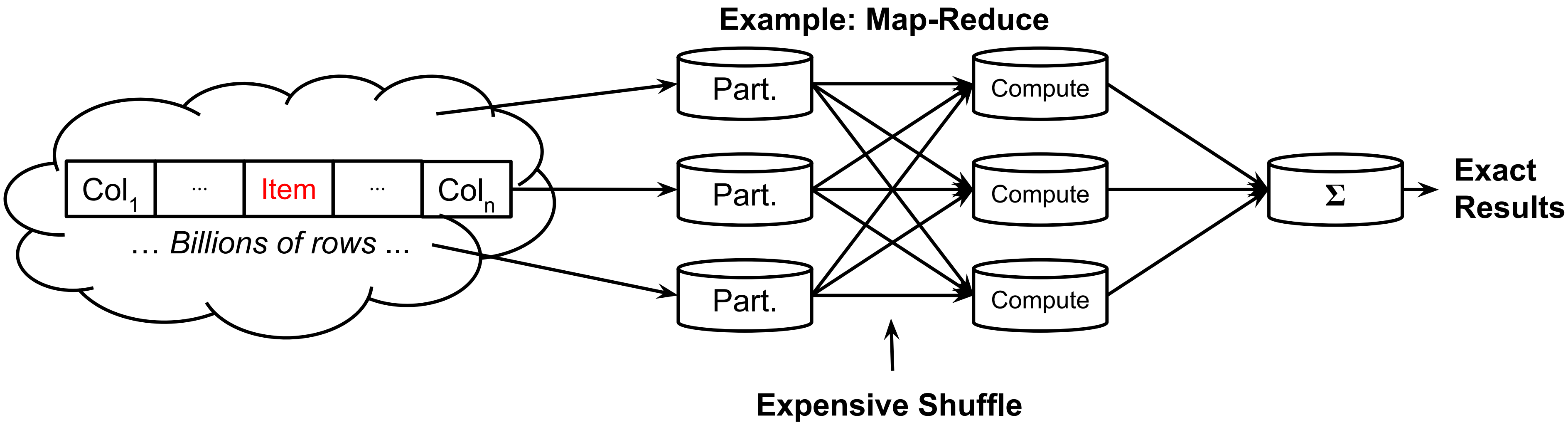
Traditional Exact Analysis Methods Require Local Copies → Big Local Storage



*Note: "Micro-batch" "Streaming Engines, e.g., **Storm**, Do Not Solve The Fundamental Problem!*

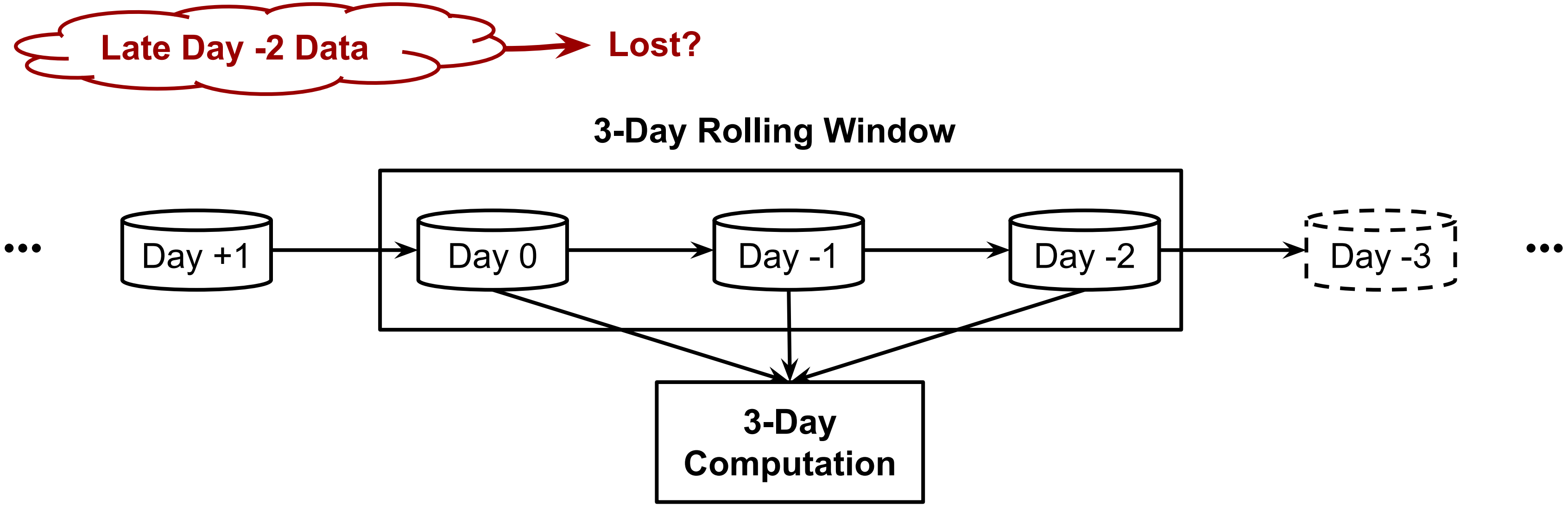
Parallelization is Not A Solution

- Because of Non-Additivity.
- Multiple touches means all the data must be available;
- And/or lots of data movement & lots of hardware



Traditional Time Windowing

Requires Multiple Touches of Every Item in Every Dataset



Every daily dataset is processed N times for a rolling N-day window!

Late Data Processing is not feasible.

Let’s Challenge the Fundamental Premise

… that our results must be exact!

If we can allow for some approximation,
along with some accuracy guarantees;

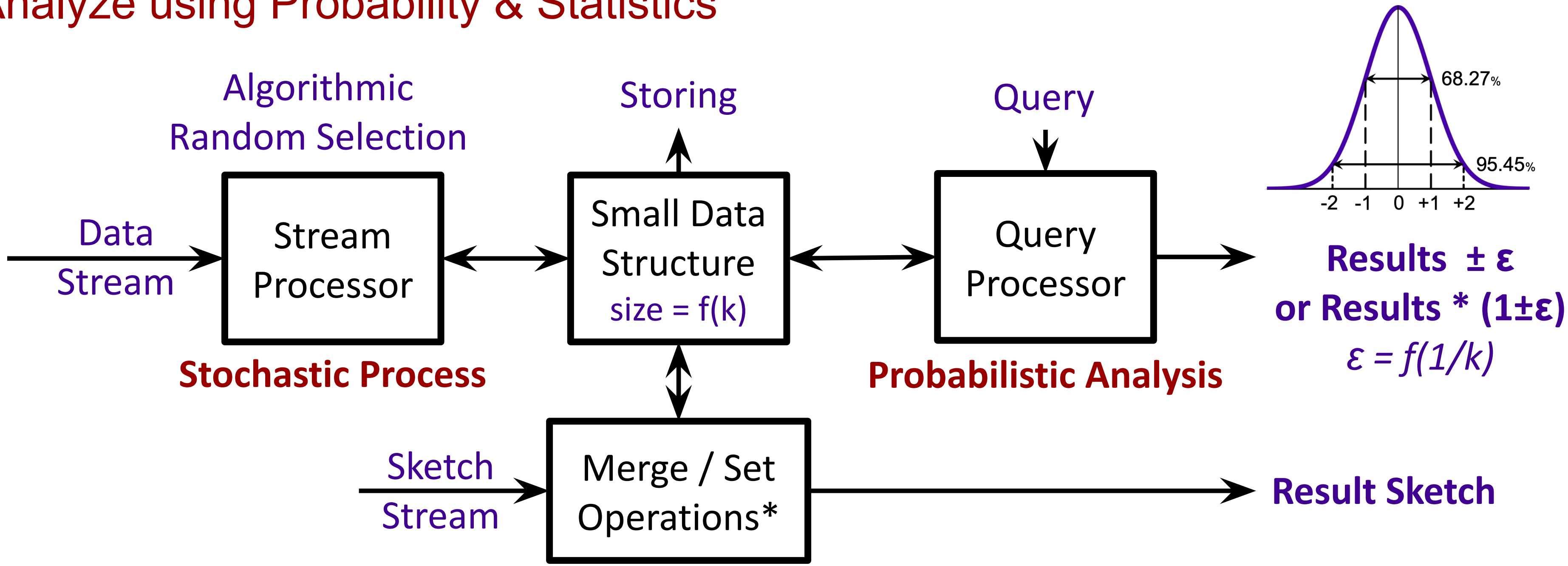
We can often achieve orders-of-magnitude improvement in

- Speed
- Storage
- Reduction of compute resources (CPU cycles)
  => Less Energy, Space, Heat, $$$

Introducing the ***Sketch*** (a.k.a, Stochastic Streaming Algorithm)

Model the Problem as a Stochastic Process with a Dynamic Data Structure.

Analyze using Probability & Statistics

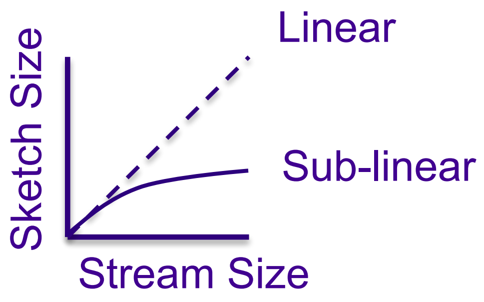


A Single Sketch Contains Many Algorithms

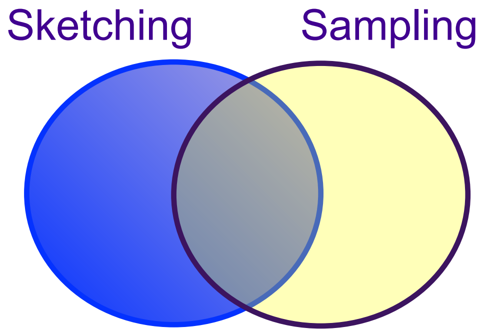
Key Sketch Properties

- Small Stored Size
- Sub-linear in Space
- Single-pass, “One-Touch”
- Data Insensitive
    - By distribution, order or value
- Mergeable
- **Mathematically Proven Error Bounds**
- Easy to use:

Sketch s = new Sketch();
while (data-exists) { s.update(datum); }
Result R = s.getResult();



Some Sketches Overlap with Sampling



Big Data Analysis Using Sketches

How & Why Sketches Achieve Superior Performance

For Systems Processing Massive Data

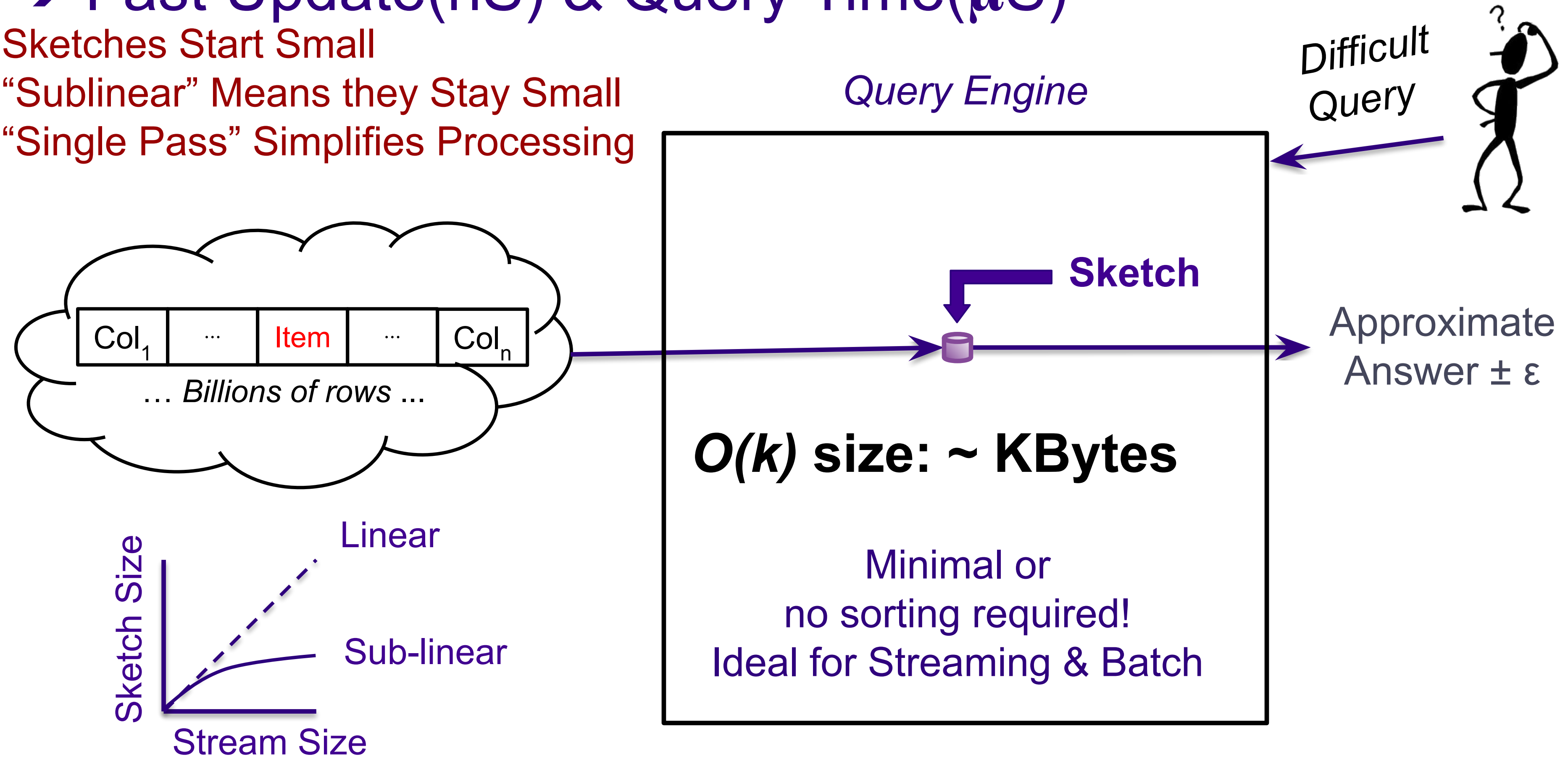
Win #1: Small Sketch Size = Small Query Space

→ Fast Update(nS) & Query Time(μS)

Sketches Start Small

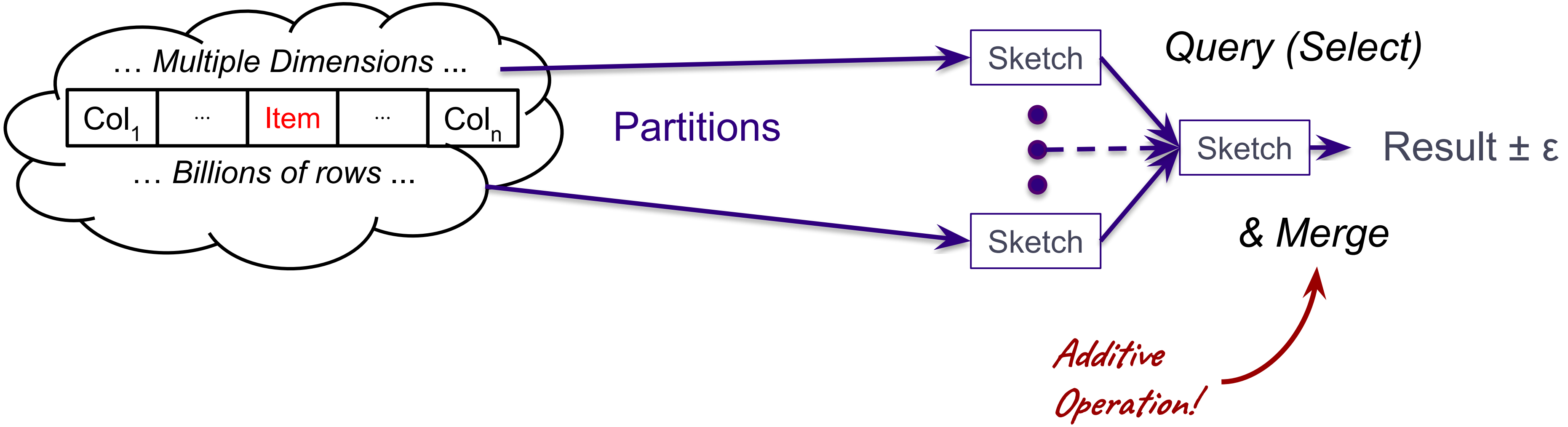
“Sublinear” Means they Stay Small

“Single Pass” Simplifies Processing



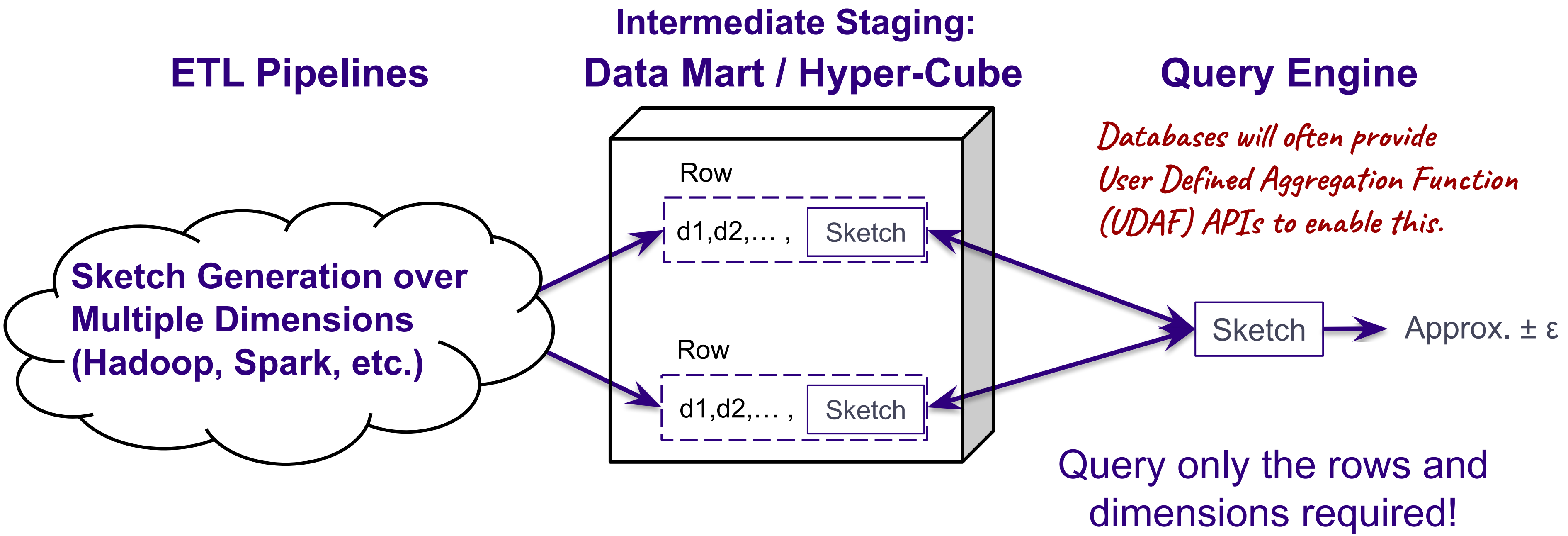
Win #2: Sketch Mergeability ➔ Parallelism ➔ Win #3: Near Real Time Performance

- Sketches transform **Non-Additive** operations into **Additive** operations
- With No Additional Loss of Accuracy!
- The Result of a Sketch Merge is Another Sketch
  - ➔ For Storage, Transport, or Set Expressions (Tuple Sketch)



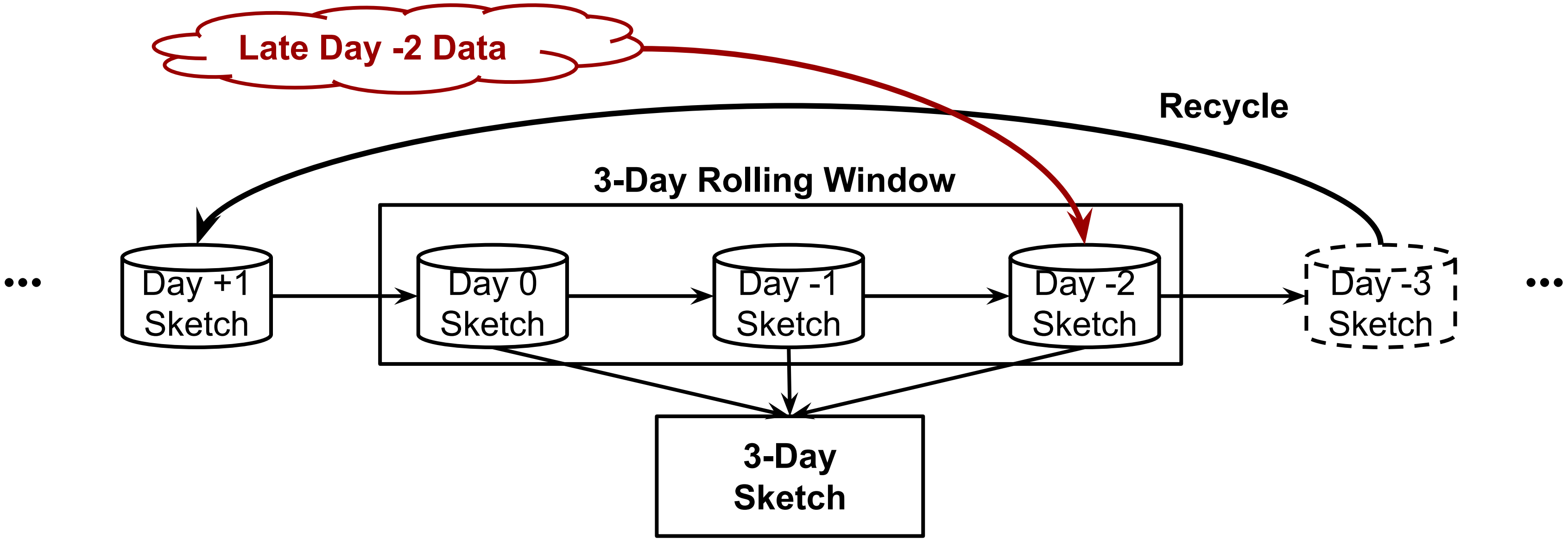
Win #4: Simpler Architecture

- Sketches are small enough to store in each row along with dimension & other data!
- Intermediate Hyper-Cube Staging Enables Fast, Multiple-Dimensional Predicate Queries



Stored Sketches Can Be Merged
By Any Dimensions, Including Time!

Win #5: Simplified Time Windowing & Late Data Processing



Every daily dataset is processed only **once** for a rolling N-day window!

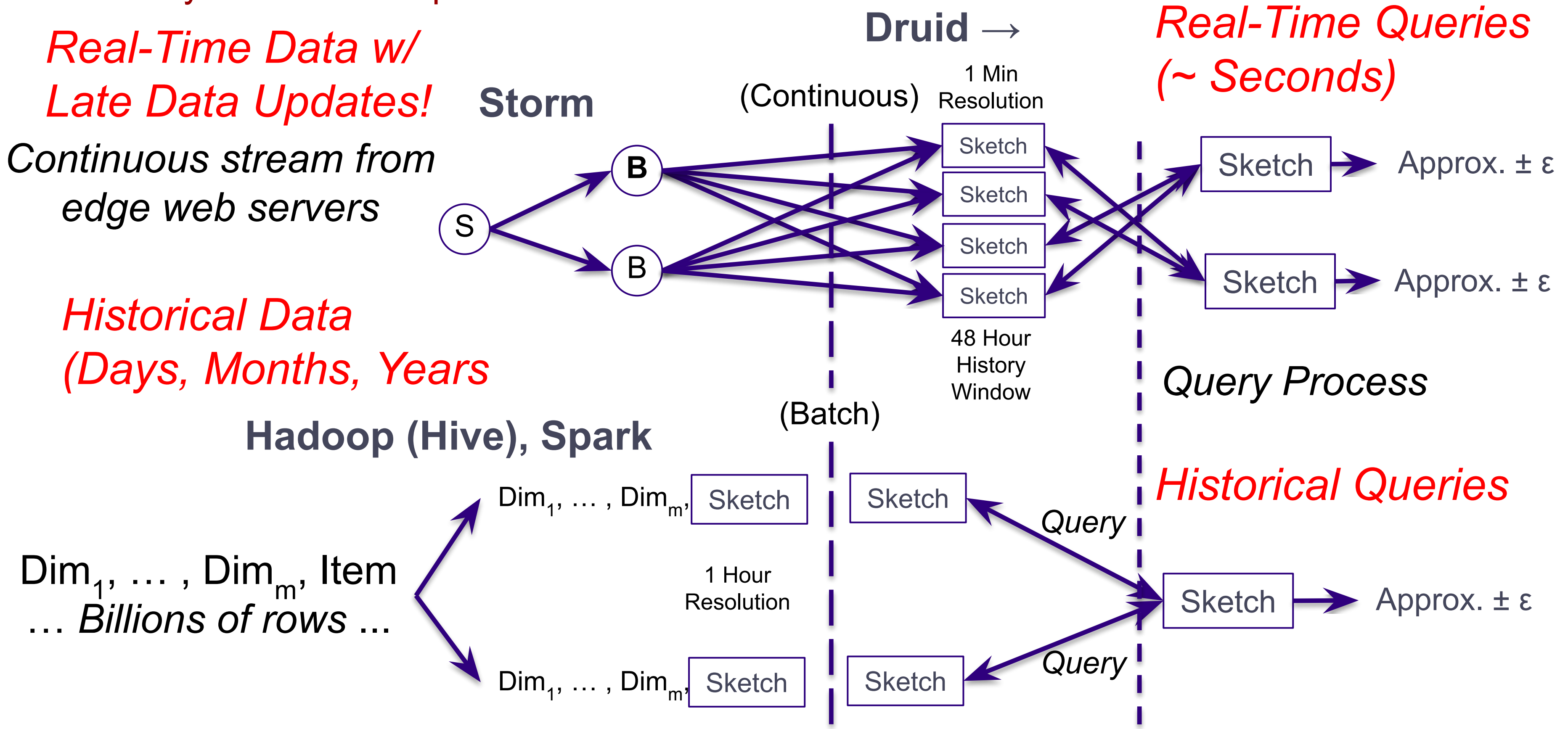
Late-data processing is now possible.

Sketches can be recycled.

Win #6: Near-Real Time Results, with History

Combine Off-Line, On-Line for Real-Time + History

Case Study: Storm/Hadoop/Druid Sketch Flow Architecture



Win #7: Lower System Cost ($)

Case Study : Before: Not Real-time; After: Near Real-time

- Customers: >250K
- Data: 40-50 TB per day
- Platform: 2 clusters X 80 Nodes = 160 Nodes
  - Node: 24 CPUs, 250GB RAM

Big Wins!
Near-Real Time,
Lower System $

	Before Sketches	After Sketches
VCS* / Mo.	~80B	~20B
Result Freshness	Daily: 2 to 8 hours; Weekly: ~3 days Real-time Results Not Feasible!	15 seconds!

* VCS: Virtual Core Seconds

Our Mission…

Combine Deep Science with Exceptional Engineering

To Develop **Production Quality** Sketches

That Address These Difficult Queries

The Sketch Design Process

1. The Art:
   Model a problem as a stochastic process and a data structure …

2. The Science:
   Analyze the data structure & algorithms using probability, statistics to extract the desired result with well understood error properties.
   Mathematically Prove that it works! Publish to Scientific Venues.
   https://datasketches.apache.org/docs/Community/Research.html

3. The Engineering:
   Transform the Art and the Science Theory into a Product!
   - Create useful APIs for use in production systems
   - Document with code examples for system engineers
   - Exhaustively test & characterize to ensure robustness
   - Publish to Open Source

World-Wide Science / Research Contacts

- ✯ Edith Cohen, Google Scientist, Israel & Bay Area
- ✯ Graham Cormode, Professor, University of Warwick, UK
- ✯ Charlie Dickens, (ex-Yahoo), Scientist, Entrepreneur, San Francisco
- ✯ Kevin Lang, (ex-Yahoo, deceased), Chief Scientist, DataSketches Project
- ✯ Edo Liberty, (ex-Yahoo), Chief Scientist, Founder, Pinecone Technologies, NY, Israel
- ✯ Justin Thaler, (ex-Yahoo), Professor, Georgetown University, Washington D.C.
- ✯ Daniel Ting, Scientist, Facebook/Meta, Seattle
- ✯ Pavel Vesely, Professor, Charles University, Prague
- ✯ …more…

The **Apache DataSketches** Library

Cardinality, 4 Families

- **HLL (on/off Heap)** A very high performance implementation of this well-known sketch
- **CPC** The best accuracy per space
- **Theta Sketches**: Set Expressions (e.g., Union, Intersection, Difference), on/off Heap
- **Tuple Sketches**: Generic, Associative Theta Sketches, multiple derived sketches:

Quantiles Sketches, 4 Families

- **Quantiles**, Histograms, PMF's and CDF's of streams of comparable objects, on/off Heap.
- **KLL**, highly optimized for accuracy-space.
- **Relative Error Quantiles(REQ),** Extremely accurate at the ends of the rank domain
- **T-Digest**, Empirical, yet very small and accurate for most data.

Frequency (Heavy-Hitters) Sketches, 3 Families

- **Frequent Items**: Weighted or Unweighted
- **Frequent Directions** (Approximate SVD) (a Vector Sketch)
- **Frequent Distinct Tuples**: Multi-dimensional Frequency & Distinct Analysis
- **CountMin:** Approx frequency for any item, enables deletions

Reservoir and Weighted-Sampling Sketches, 3 Families

- **Reservoir:** Uniform random to fixed-k sized buckets. Mergeable with different size k.
- **VarOpt:** Optimal variance weighted sampling for subset sums.
- **EBPPS Sampling**: Exact and Bounded Probability Proportional to Size

Specialty Sketches

- **Customer Engagement, Sketch Maps**, etc.
- **Differential Privacy,** (experimental)
- **Density Estimation**
- **Filters, e.g. BloomFilter, QuotientFilter (under development)**

Languages Supported:

- Java, C++, Python, Go*
- Binary Compatibility across languages and history (12+ years)

* In development

Who Uses DataSketches?…That we know of!

We only find out about users if they contact us.

● **Public Database Integrations:** Hive, Pig, PostgreSQL, Druid, Spark, Presto, GCHQ / Gaffer, Netflix / Atlas Database, Pinot, Iceberg, Vertica, Greenplum, ClickHouse, Impala, Quix.io, GoogleCloudPlatform/BigQuery, (AWS via *lift & shift*)…

● **Major users:** Microsoft Gray Labs, GCHQ, Canadian Communications Security Establishment (CSE), GameAnalytics, Visa, Nielsen, DataDog, Criteo, Imply, Permutive, Expedia …

● **Library Integrations:** TechAscent/tech.ml.dataset (TMD), WhyLabs (ML)

Bright Future for Sketching Technology & Solutions ...

Items (words, IDs, events, clicks, …)
- Count Distinct
- Frequent Items, Heavy-Hitters, etc
- Quantiles, Ranks, PMFs, CDFs, Histograms
- Set Operations
- Sampling: Uniform, Weighted, Proportional
- Mobile (IoT)
- Moment and Entropy Estimation
- Turnstyle Sketches
- Filters: Bloom, Quotient, Infinifilter, etc.

Graphs (Social Networks, Communications, …)
- Connectivity
- Cut Sparsification
- Weighted Matching
- Path & Funnel Analysis

Vectors (text docs, images, features, …) And Matrices (text corpora, recommendations, …)
- Dimensionality Reduction (SVD)
- Ridge Regression
- Covariance Estimation
- Low Rank Approximation
- Sparsification
- Clustering (k-means, k-median, …)
- Linear Regression
- Machine Learning (in some areas)
- Density Estimation

Differential Privacy (experimental)
- Reach, Frequency

Areas where we have sketch implementations / solutions
In Development
Areas of research (World-wide)

THANK YOU!

Learn more about Apache DataSketches
Visit Us and become a Contributor!
https://datasketches.apache.org

Apache DataSketches

Case Study 1: Simple Batch Distinct Counting

- **Web Logs:** *Dim1*: PageID, *Dim2*: Time-Stamp,
  Id1: Browser Cookie, *Id2*: UserID
  (+ many other fields)
- **Data Size**: ~245GB daily; ~7.6TB monthly
- **Task**: Report: *Count Distinct Id1* and *Id2* by PageID, and by hour, day, week, and month
- **Note**: This case study was run on Pig, Hive and Spark. The results below are from Pig. Hive and Spark showed similar results.

Case Study 1: Hourly Process

Exact: For Hourly Reports and Basis for Daily Reports

Sub-Task	Data Stored
Stage 1: • Read Raw Data • -> Hourly Tables	Create Table1: Group By {site, hour, id1} Create Table2: Group by {site, hour, id2}
Intermediate Size	33.4 GB 1 Month of Hourly
Stage 2a: • Read Hourly Tables • Count Uniques	Group By {site, hour}, count Id1 Group By {site, hour), count Id2
Stage 2b: • -> Hourly Report	Join: {site, hour, count(id1), count(id2)}
Total CPU Time	1.39M Sec = 16 d, 2hr

Ratio: 30.4 : 1

Ratio: 1.31 : 1

Sketches Cube: For All Reports

Sub-Task	Data Stored
Stage 1: • Read Raw Data • -> Data Cube	Create Sketches Cube: By Dim Combination {site, hour, sketch(id1), sketch(id2)}
Intermediate Size	1.1 GB
Stage 2 • Read Data Cube • Produce Hourly Report	Merge Sketches across Chosen Dimensions
Total CPU Time	1.06M Sec = 12d, 6hr

Case Study 1: Daily Rollups

Exact: For Daily Reports and Basis for Weekly and Monthly

Sub-Task	Data Stored
Stage 1: • Read Hourly Intermediates • -> Daily Tables	Create Table1: Group By {site, day, id1} Create Table2: Group by {site, day, id2}
Intermediate Size	16.0 GB just for Daily
Stage 2a: • Read Daily Intermediates • Count Uniques	Group By {site, day}, Count Id1 Group By {site, day), Count Id2
Stage 2b: • -> Hourly Report	Join: {site, day, count(id1), count(id2)}
Total CPU Time	96,300 sec =26.75 hrs

Ratio: 135.8 : 1

Sketches Cube: For All Reports

Sub-Task	Data Stored
Stage 1: • Read Data Cube • -> Produce Daily Report	N/A
Intermediate Size	N/A
Total CPU Time	709 Sec = 11.8 min

Case Study 1: Weekly, Monthly Rollups

Exact: For Wk/Mo Reports

Sub-Task	Data Stored
Stage 1: • Read Daily Tables	Create Temp Table1: Group By {site, wk/mo, id1} Create Temp Table2: Group by {site, wk/mo, id2}
Stage 2a: • Read Temp Tables • Count Uniques	Group By {site, wk/mo}, Count Id1 Group By {site, wk/mo), Count Id2
Stage 2b: • Produce Report	Join: {site, wk/mo, count(id1), count(id2)}
Total CPU Time	Week: 43,500 sec (12 hrs) Month: 46,500 sec (13h, via daily) Month: 70,900 sec (20h, via hourly)

Ratios:
Week: 102.6 : 1
Mo.: 99.79 : 1
Mo.: 152 : 1

Sketches Cube: For All Reports

Sub-Task	Data Stored
Stage 1: • Read Data Cube • -> Produce Weekly or Monthly Reports	N/A
Total CPU Time	Week: 424 Sec Month: 466 Sec

Case Study 1: Perspectives

- Only a few dimensions and metrics, moderate data size
  - Manageable with exact counting
  - However, sketching can still show substantial benefits, especially in real-time streaming
- Batch process (e.g. Pig, Hive)
  - Substantial job overhead **penalizes** the relative sketch compute time.
  - Contrast this to real-time reporting engines (e.g. Druid), where rollups can be computed in seconds.
- As the number of dimensions grows, and as the input size grows, the benefit of using sketches becomes even more dramatic

Recently Added Sketches

CountMin Sketch

- Approximate Frequency Estimation for any queried item in contrast to our Frequent Items Sketch for Heavy Hitters.
- Basic building block for numerous types of analysis
- Can be configured to enable deletions as well as insertions

Density Sketch

- Builds a coreset from the given set of input points
- Provides a density estimate at any given point
- Provides for a user defined Kernel Function as well as built-in ones for computing distance between two vectors.
- Handles multi-dimensional vectors

EB-PPS Sampling

- **E**xact and **B**ounded, **P**robability **P**roportional to **S**ize
- Produces a random sample of data from a stream of weighted items, ensuring that the probability of including an item is always exactly equal to the item's weight.
- This is in contrast to our VarOpt sketch, which provides optimal variance in the computation of subset sums given a random sample of weighted inputs.

T-Digest

- An empirical quantiles sketch that has excellent size and error properties.
- This sketch does not have mathematically proved error properties, in contrast with our other quantile sketches that do.

BloomFilter, QuotientFilter

Recently Added Sketch Enhancements

KLL & Classic Quantiles Items Sketches

- Added highly scalable Partitioning Feature for partitioning very large data sets into equally sized partitions.
- Added integer-weighted input capability
- Added data type “long”

Theta Sketch

- Added a new compression capability that provides for superior size reduction of serialized sketch images compared to standard compression tools