bigdata
@CSAIL

ISTC
BIG DATA

Interactive Data Analytics: the New Frontier

Sam Madden
madden@csail.mit.edu

With a cast of many…

SOCC'15 Keynote – Kona, HI

CSAIL

bigdata
@CSAIL

ISTC
BIG DATA

BIG Data

facebook

MIT COMPUTER SCIENCE AND ARTIFICIAL INTELLIGENCE LABORATORY

When Do You Have a Big Data Problem?

- **Too many bytes (Volume)**
- **Too high a rate (Velocity)**
- **Too many sources (Variety)**

Real Challenge: Understanding Data

Required interactivity is poorly supported by today's data intensive systems

What does the data look like?

Show me unusual patterns, events, or outliers?

Where are these anomalies and outliers coming from?

Quickly, as data changes, for arbitrary subsets of the data

Three Interactive Analytics Data Processing Tools We've Built

- **MapD**
  - Interactive data exploration
- **SeeDB**
  - Automatic visualization
- **Scorpion**
  - Understanding "why" in aggregate queries

Can work w/ conventional databases but do better with custom data processing engines

MapD: Interactive Large-Scale Visualization using a GPU Database

MAPD (MASSIVELY PARALLEL DATABASE)

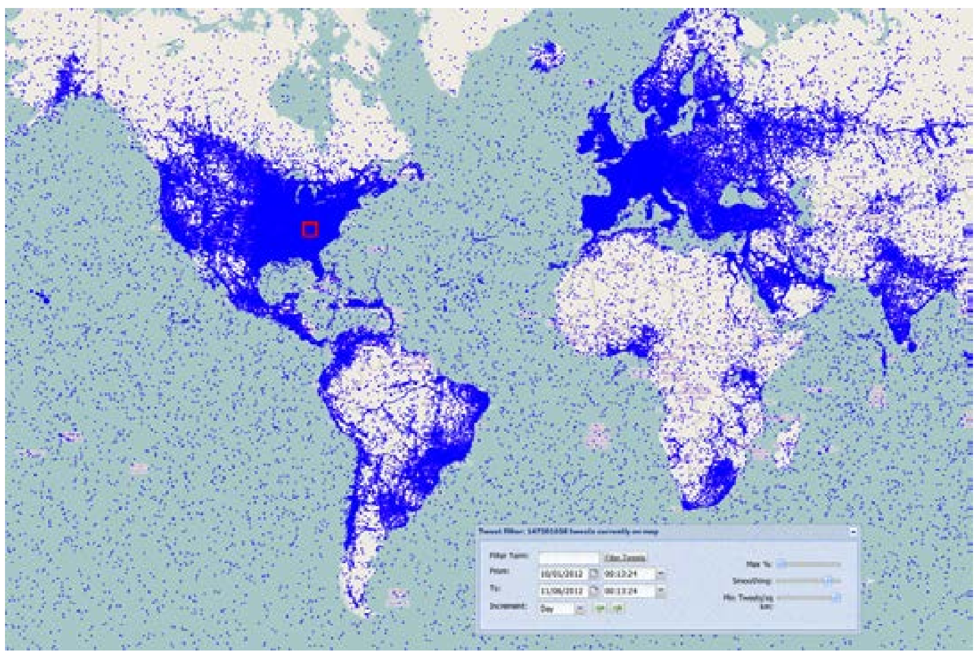
USING GPUS FOR REAL-TIME QUERYING AND VISUALIZATION OF BIG DATA

w/ Todd Mostak

The Need for Interactive Analytics

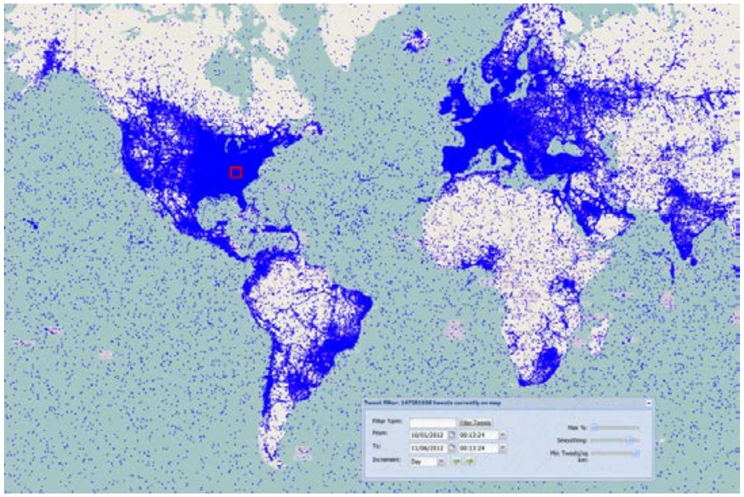
- **First step in analysis is *browsing***
  - *Often visualization*

➔ad-hoc analytics, with millisecond response times

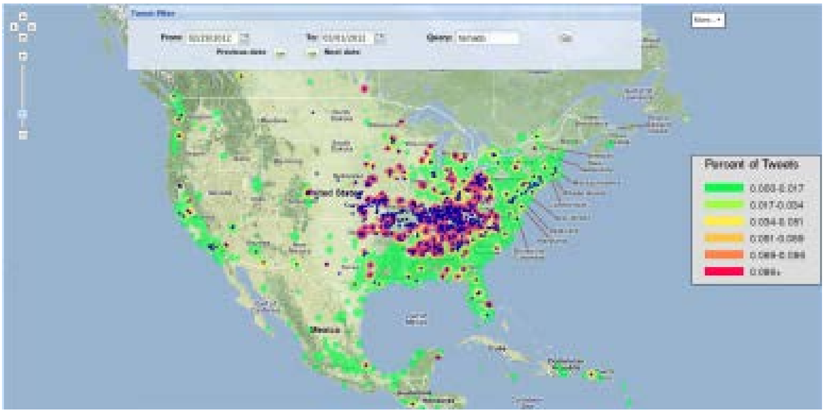


MapD: GPU Accelerated SQL Database

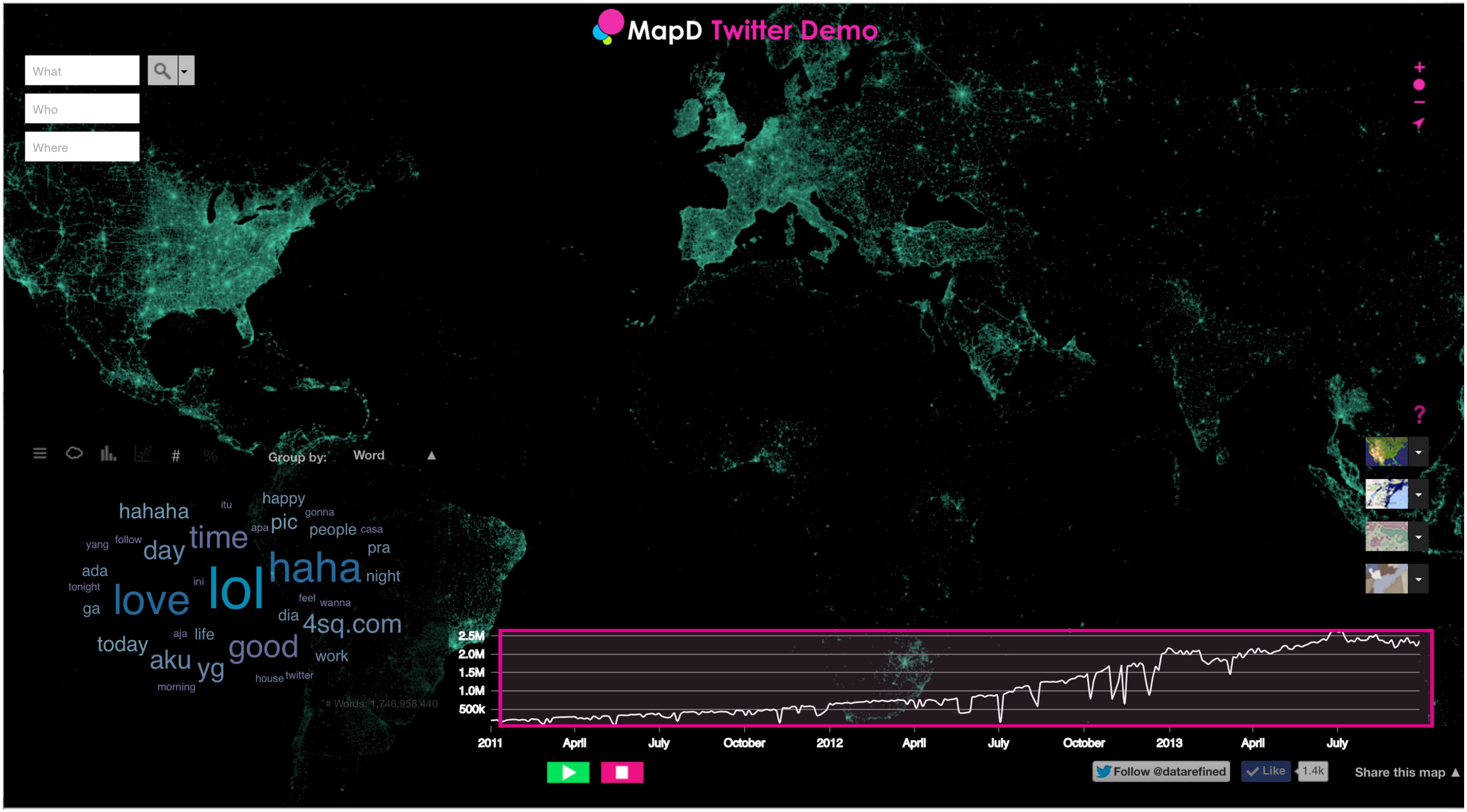
- ***Key insight:*** **GPUs have enough memory that a cluster of them can store substantial amounts of data**
- **Not an accelerator, but a full blown query processor!**
- **Massive parallelism enables interactive browsing interfaces**
  - 4x GPUs can provide > 1 TB/sec of bandwidth
  - 12 Tflops compute
  - Order of magnitude speedups over CPUs, when data is on GPU
- **"Shared nothing" arrangement**



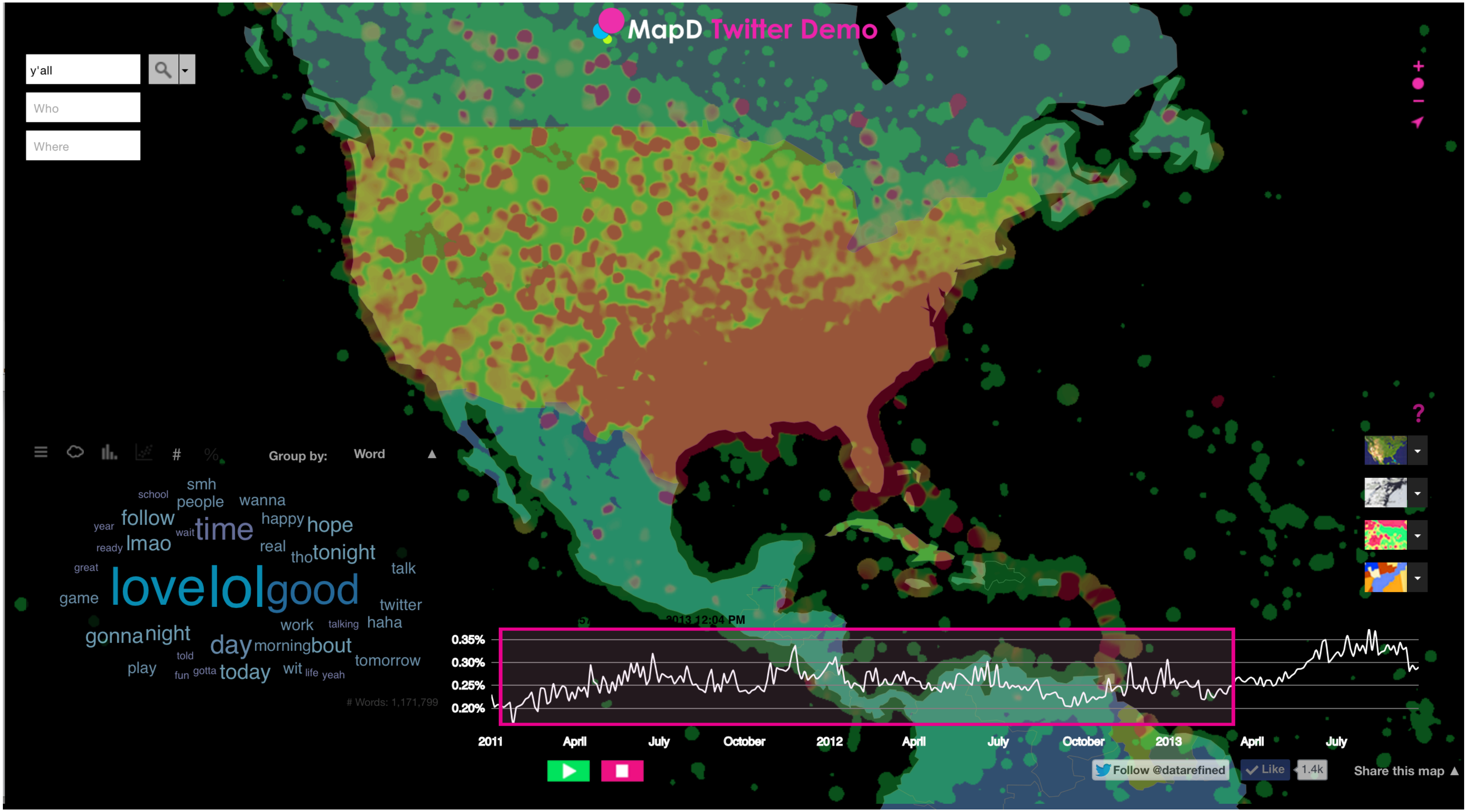
147,201,658 tweets from Oct 1, 2012 to Nov 6, 2012



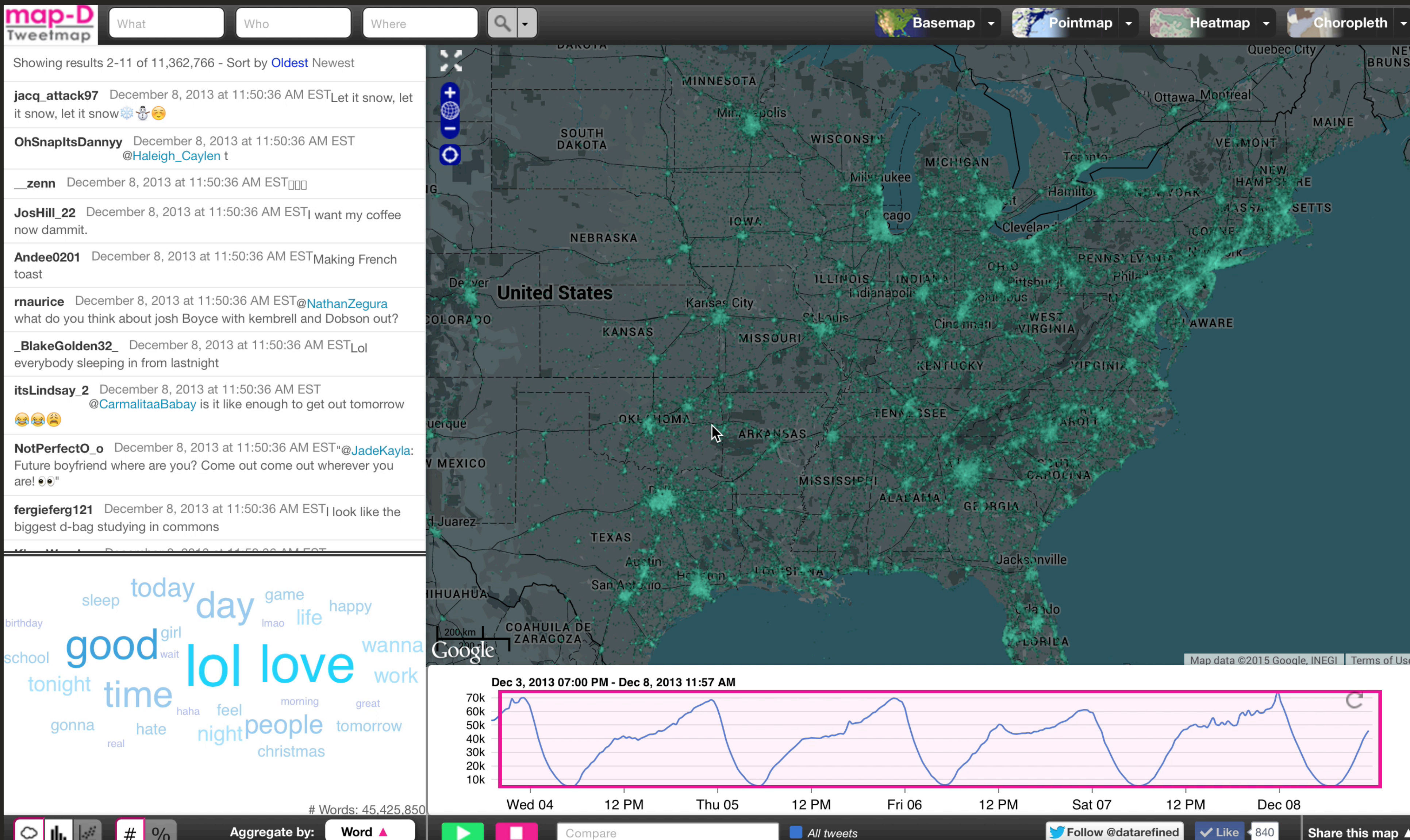
Relative intensity of "tornado" on Twitter (with point overlay) from Febuary 29, 2012 to March 1, 2012



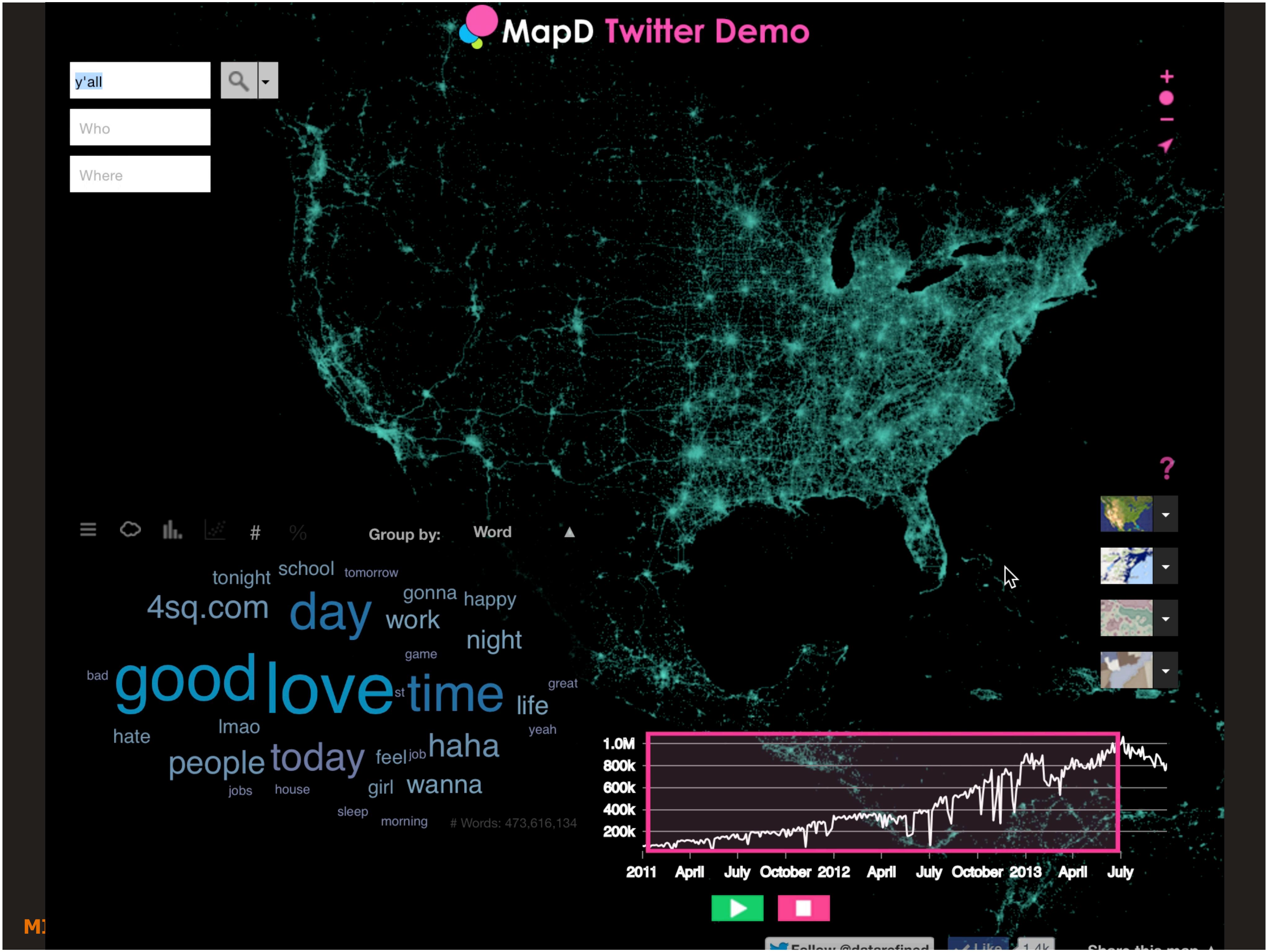
MIT COMPUTER SCIENCE AND ARTIFICIAL INTELLIGENCE LABORATORY



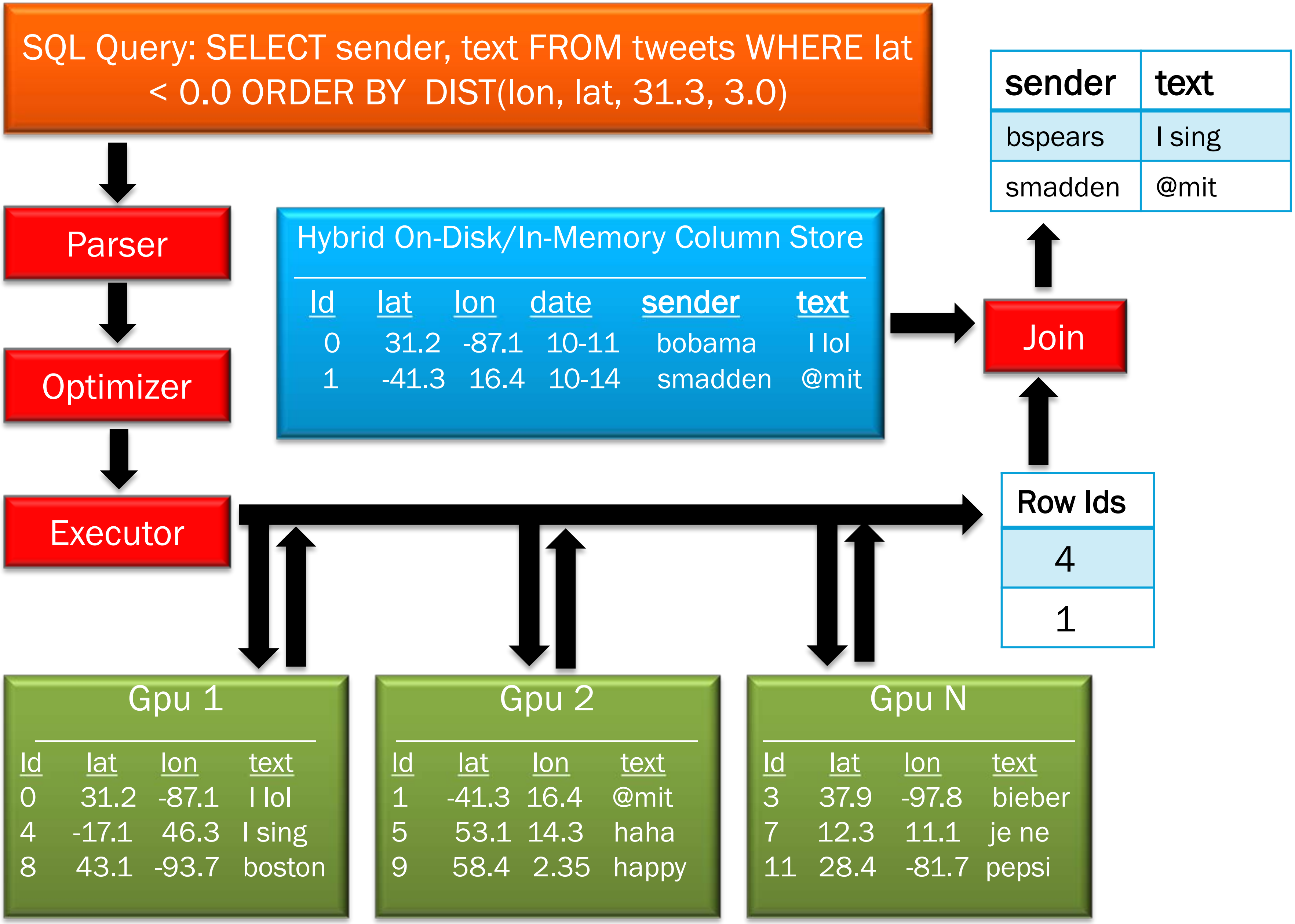
MIT COMPUTER SCIENCE AND ARTIFICIAL INTELLIGENCE LABORATORY



MIT COMPUTER SCIENCE AND ARTIFICIAL INTELLIGENCE LABORATORY

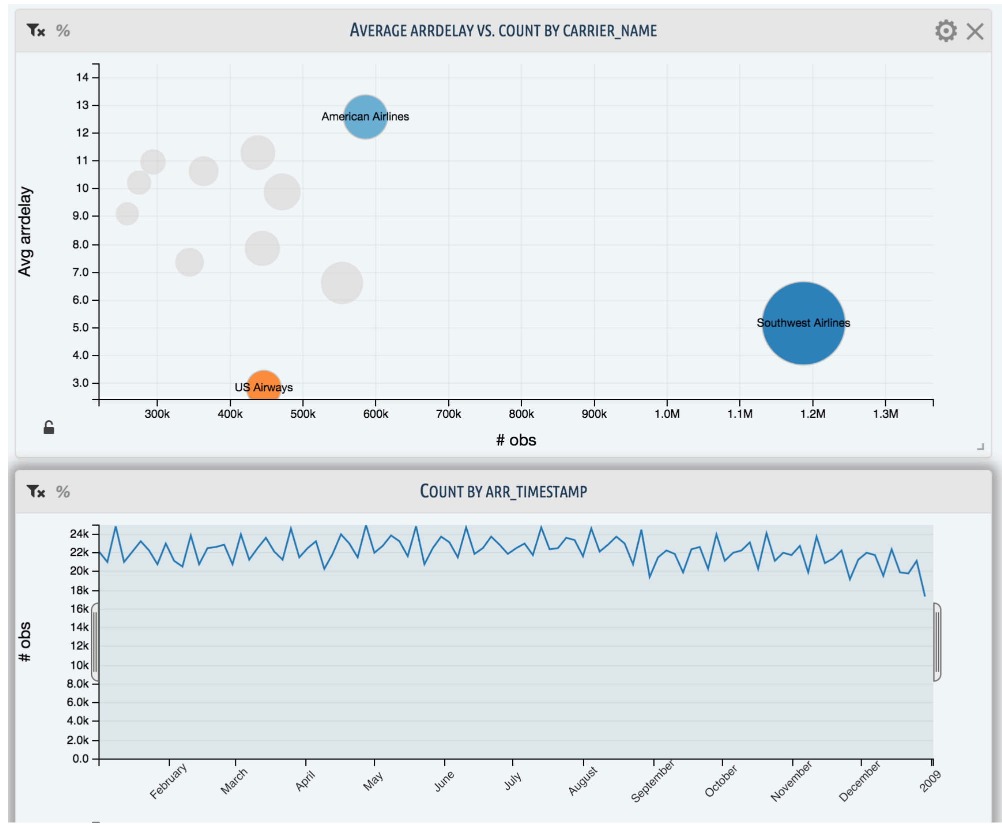

MapD Twitter Demo
y'all
Who
Where
Group by:
Word
tonight
school
tomorrow
4sq.com
day
gonna
happy
work
night
game
bad
good
love
st
time
great
life
hate
lmao
yeah
people
today
feel
job
haha
jobs
house
girl
wanna
sleep
morning
Words: 473,616,134
1.0M
800k
600k
400k
200k
2011
April
July
October
2012
April
July
October
2013
April
July

ANATOMY OF A QUERY



Next Steps

- Scale out to many nodes, automate layout algorithms
- Add various advanced analytics (e.g., machine learning algorithms)
- Generalize visualization beyond maps

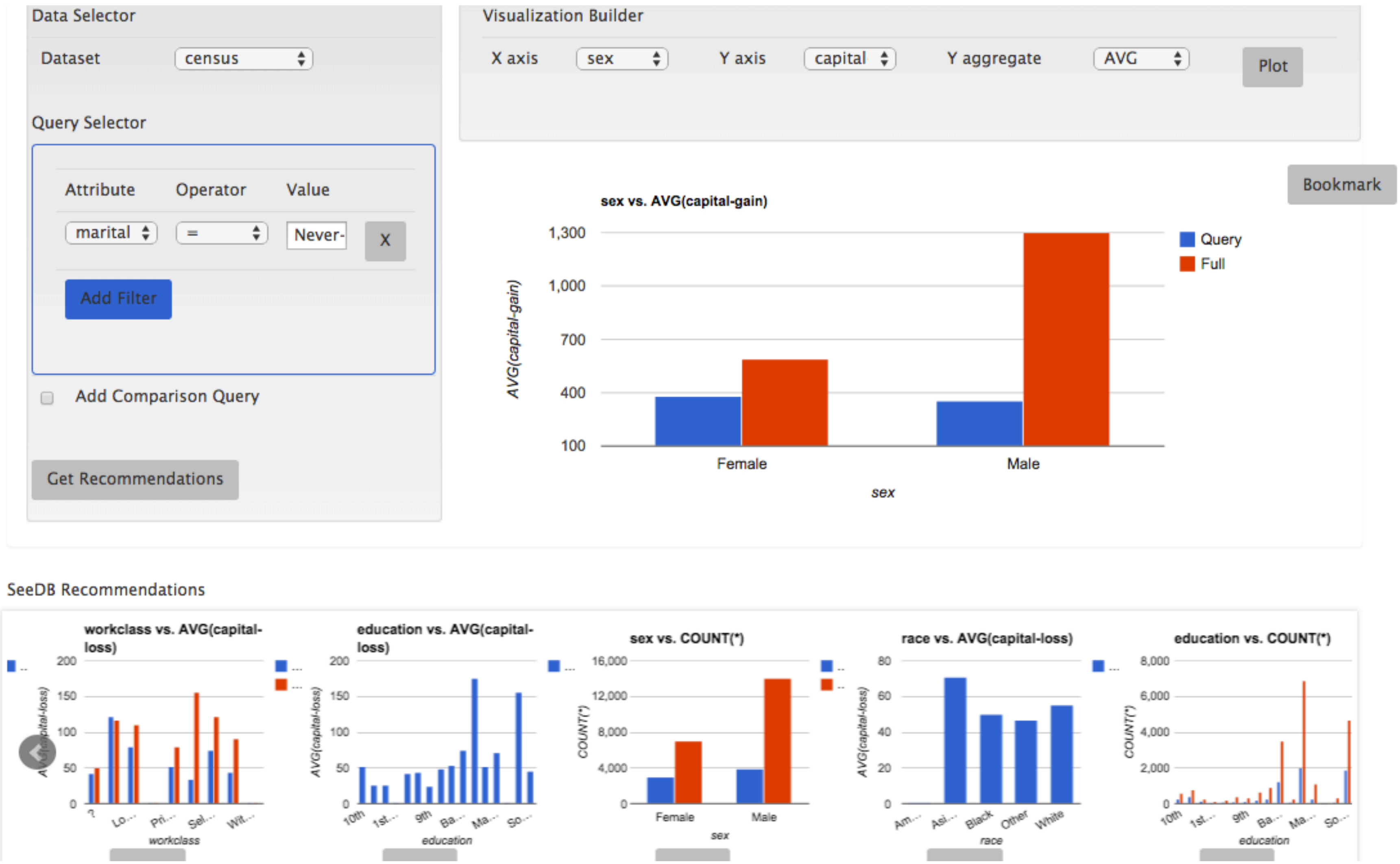


Three Interactive Analytics Data Processing Tools We've Built

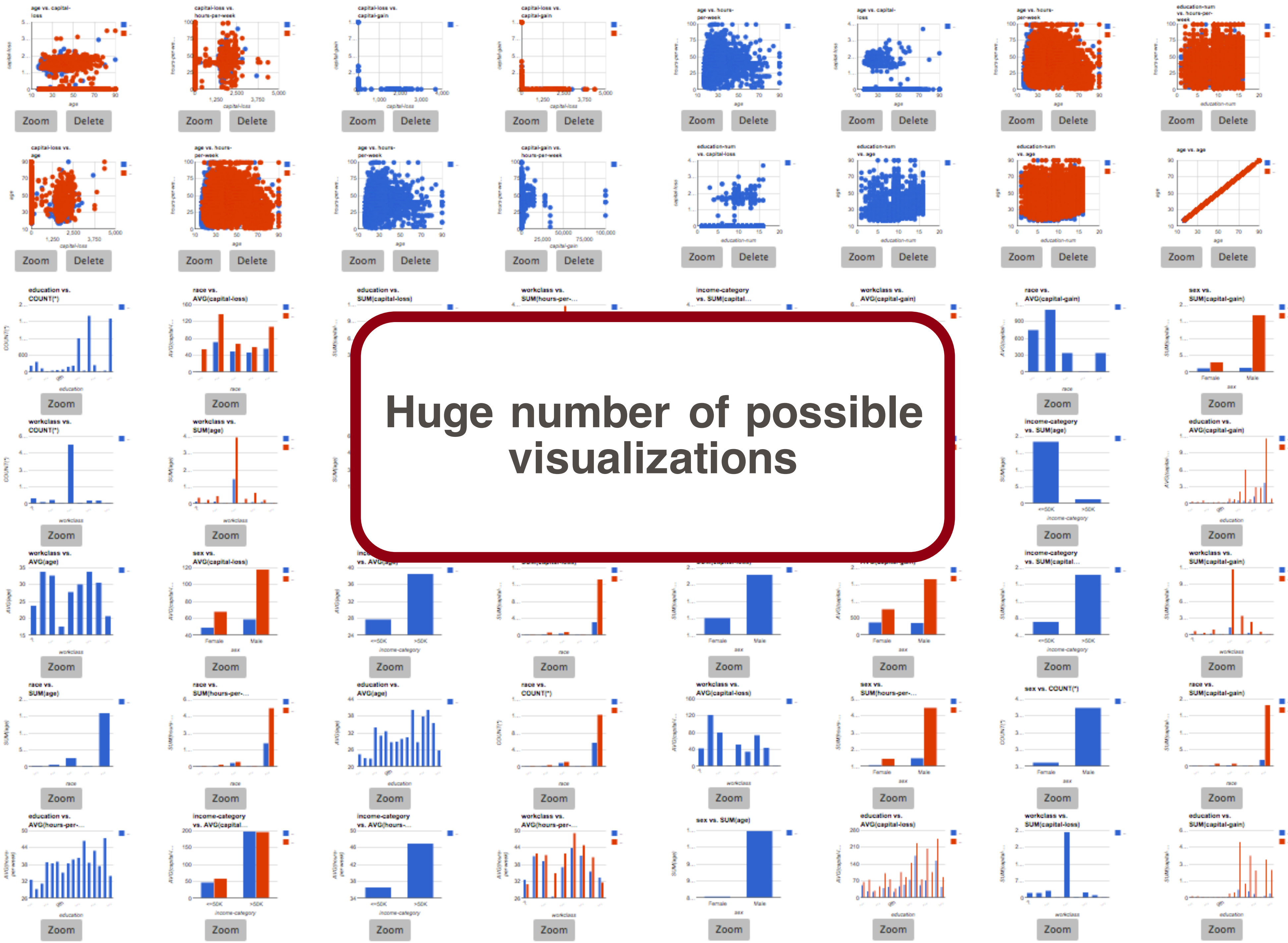
- **MapD**
  - Interactive data exploration
- **SeeDB**
  - Automatic visualization
- **Scorpion**
  - Understanding "why" in aggregate queries

Can work w/ conventional databases but do better with custom engines

SeeDB: Visual Recommendations



w/ Manasi Varak, Aditya Parameswaran, Neoklis Polyzotis



Huge number of possible visualizations

Recommending Visualizations

- **How to find relevant visualizations?**
  - Need a utility metric
  - Axes: Data, User Preferences, Aesthetics

- **Goal: interactive recommendations?**
  - Scale to large number of rows
  - Manage curse of dimensionality

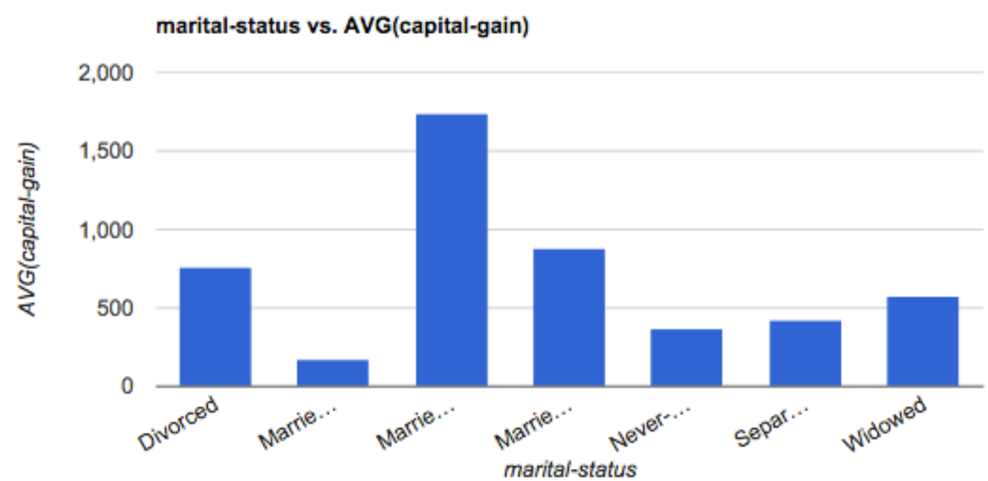
SeeDB Visualizations

- $V_i = (d : \text{dimension}, \ m : \text{measure}, \ f : \text{aggregate})$
- AGGREGATE + GROUP BY queries

```
SELECT d, f(m) FROM table GROUP BY d
WHERE selection_predicate
```

Naïvely evaluated through sequential scans of dataset

Result: bar chart



Deviation-based Utility Metric

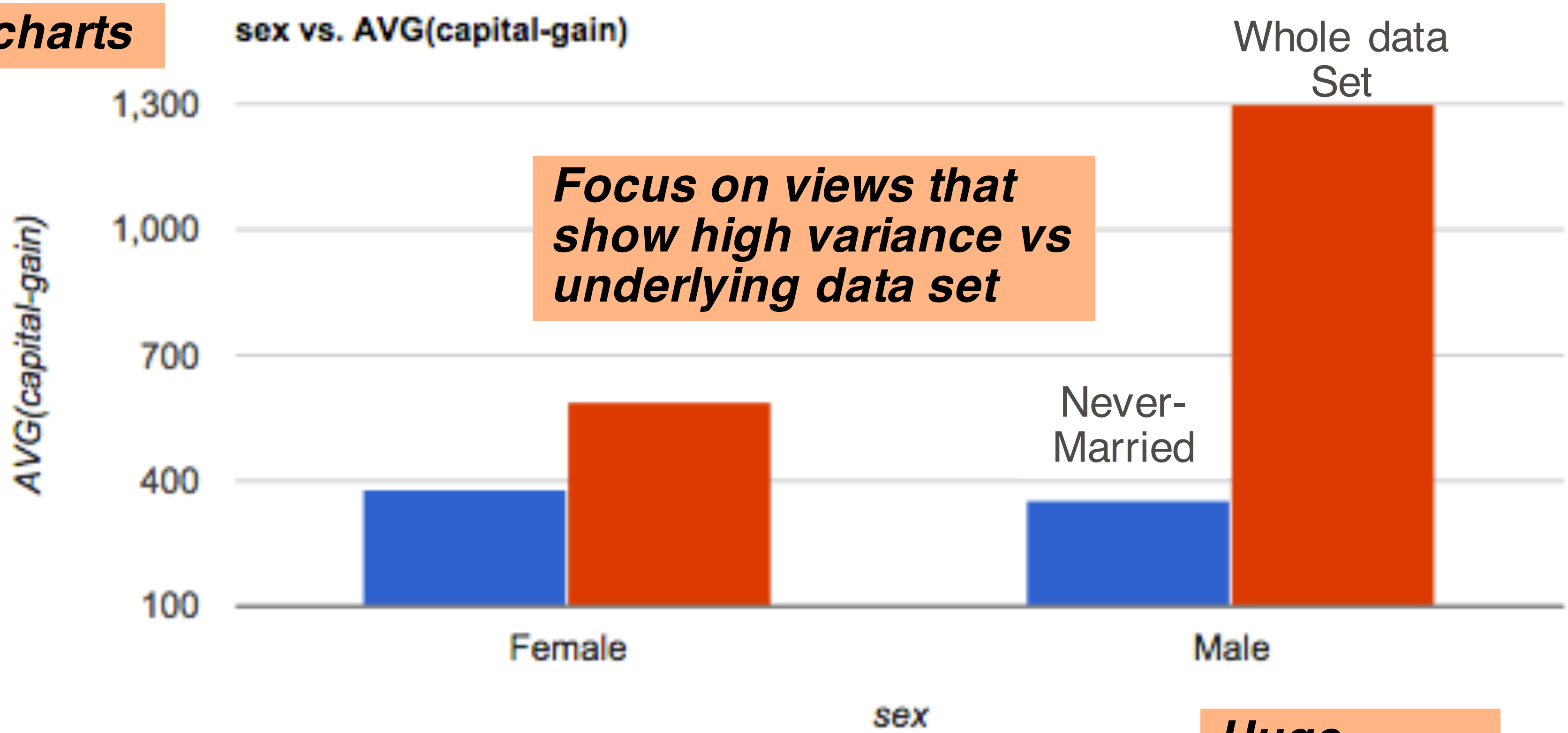
Find visualizations (d, f(m) sets) such that the difference between the query with the selection_predicate and with no predicate is maximized

Recall our query template:

```
SELECT d, f(m) FROM table GROUP BY d
WHERE selection_predicate
```

Example Recommendation (Census Example)

Bar charts



Huge space of related problems!!!

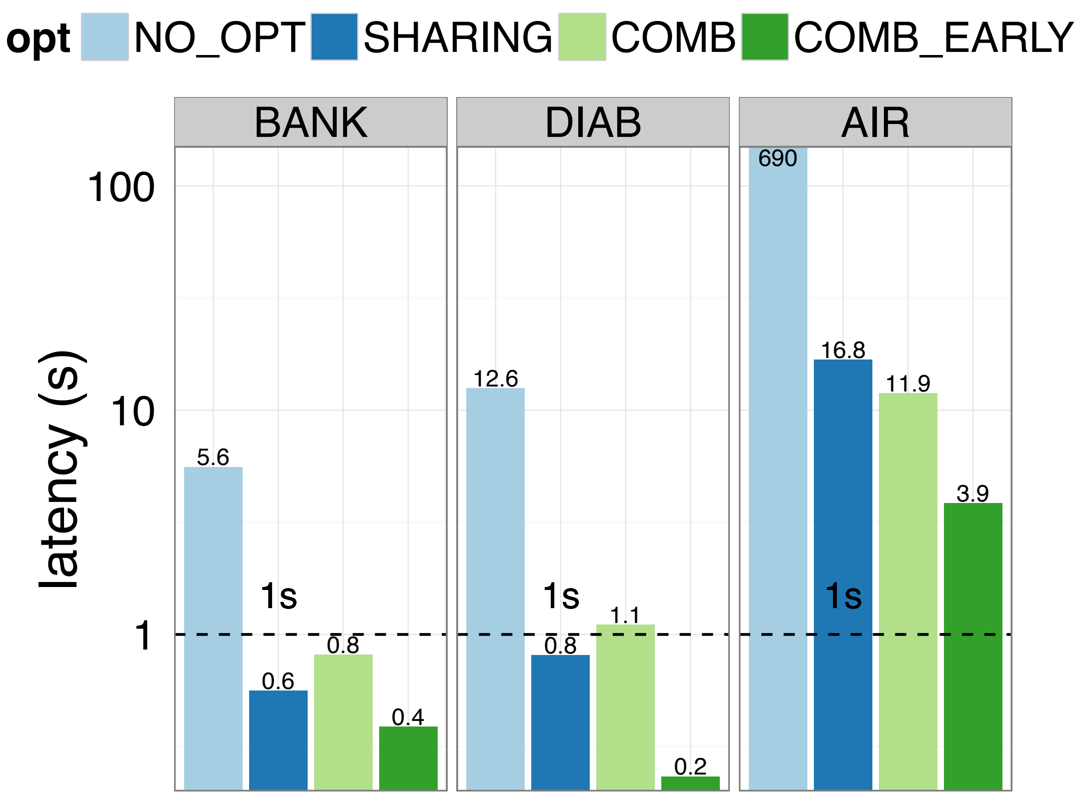
SELECT gender, AVG(capital-gain)
WHERE marital-status = NEVER_MARRIED
GROUP BY gender

Challenge and Solution Sketch

- **Exponential number of possible visualizations**
  - Can plot any set of attributes against any other set!

- **Solutions:**
  - Several optimizations to *batch* queries together, to explore the search space more efficiently
  - Algorithms to *prune* space of visualizations
    - **Idea: Quickly discarding those of low utility**
    - **Evaluate visualizations on a small sample of data**
      - Discard ones that perform poorly, and repeat

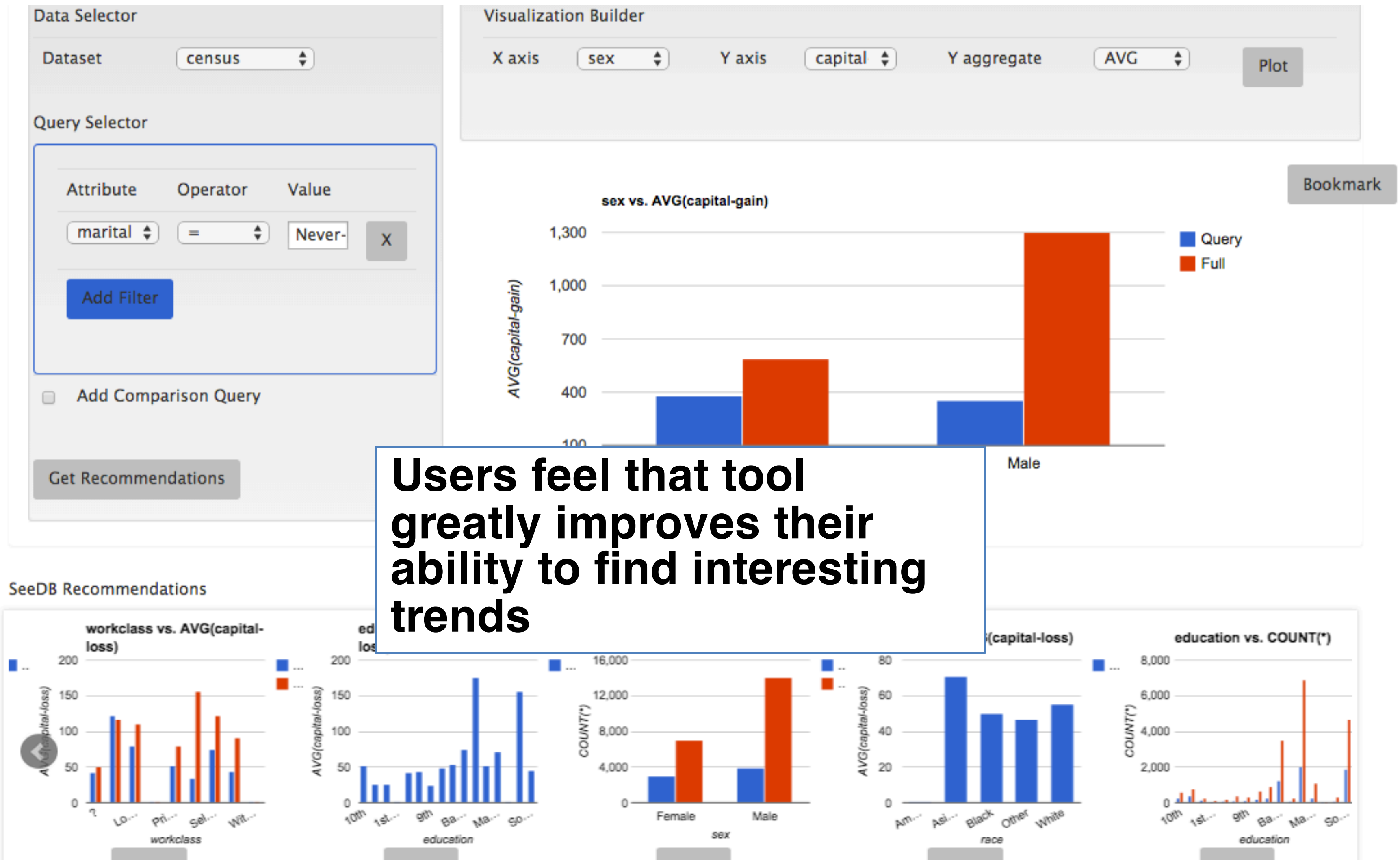
Time to Find Top 10 Visualizations



AIR: 10 GB flight dataset
DIAB: 1 GB diabetes patient dataset
BANK: 600 MB banking dataset

SeeDB returns results in < 4 s for all data sets vs > 700s for naïve approach

SeeDB: Visual Recommendations



Users feel that tool greatly improves their ability to find interesting trends

Three Interactive Analytics Data Processing Tools We've Built

- **MapD**
  - Interactive data exploration
- **SeeDB**
  - Automatic visualization
- **Scorpion**
  - Understanding "why" in aggregate queries

Can work w/ conventional databases but do better with custom engines

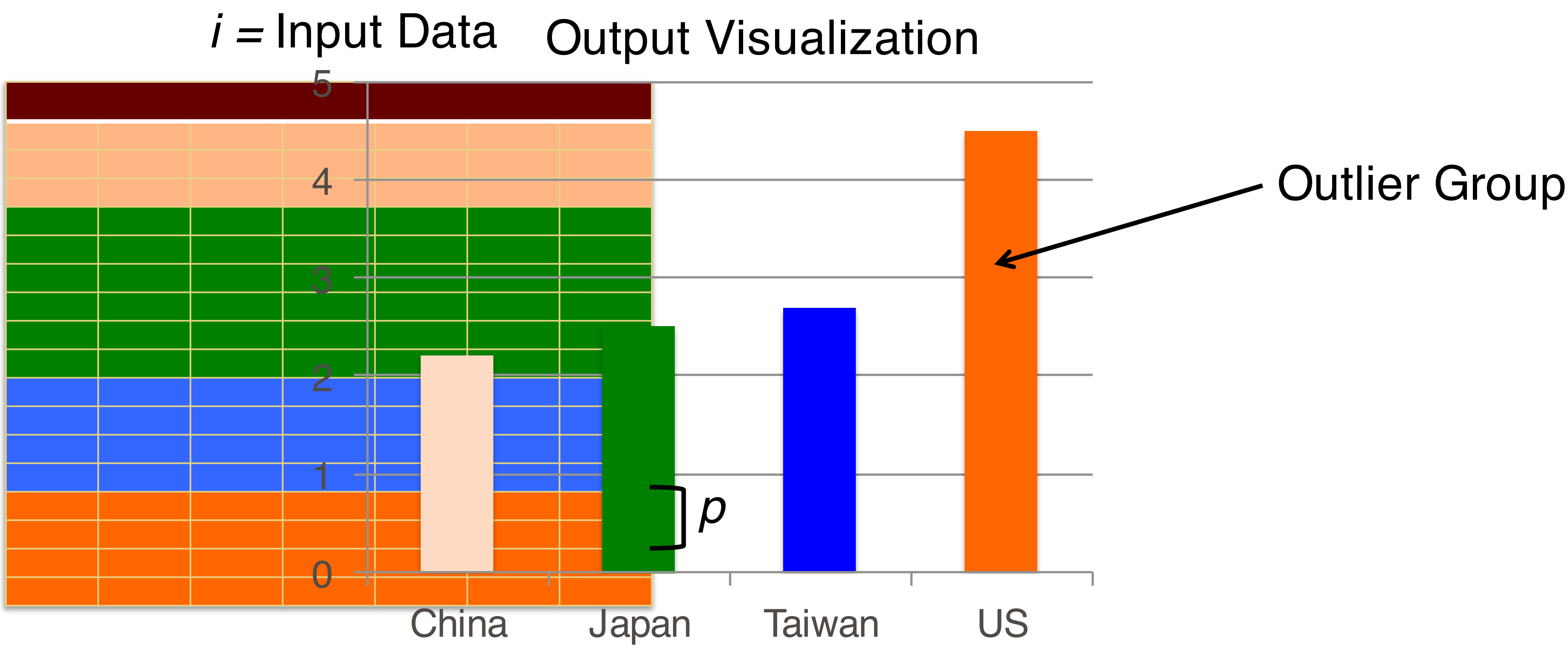
Scorpion

- After Se… …sting, now what?

- Commo…
- Need: a… …xist

Definition of Why

Given an outlier group, find a *predicate* over the inputs that makes the output no longer an outlier.

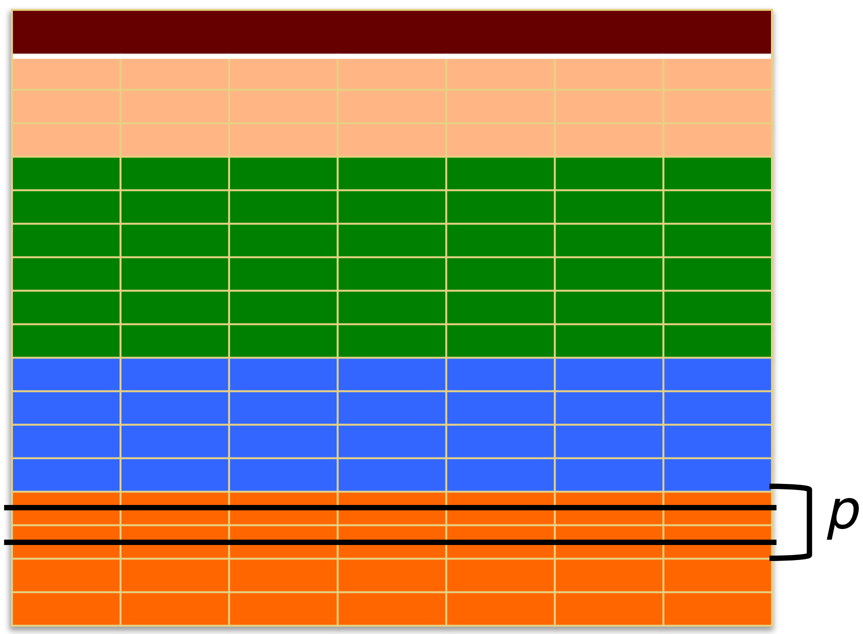


p = predicate

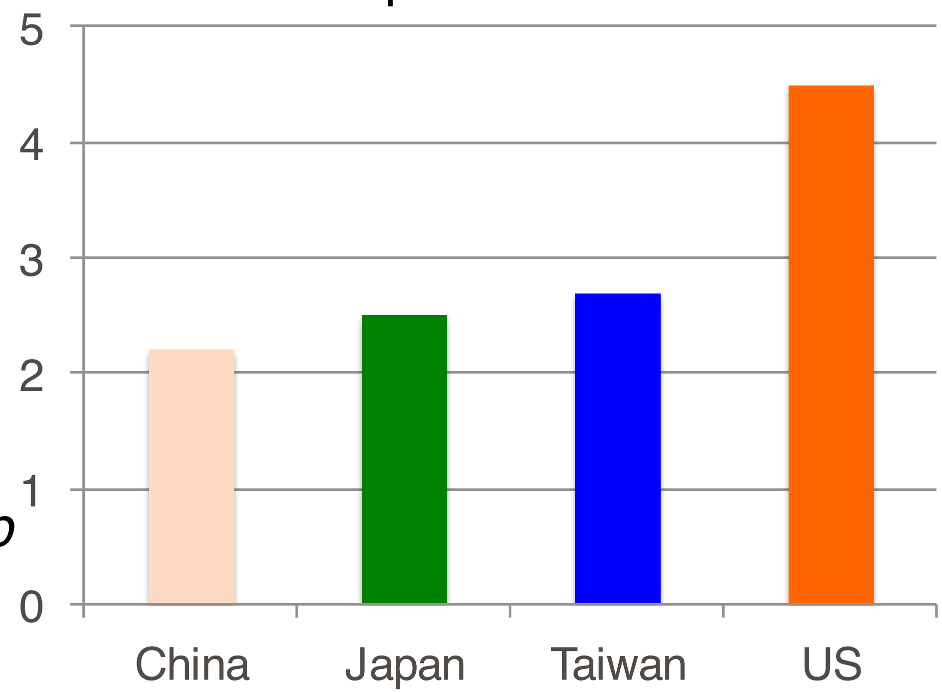
Definition of Why

Given an outlier group, find a *predicate* over the inputs that makes the output no longer an outlier.

i = Input Data



Output Visualization

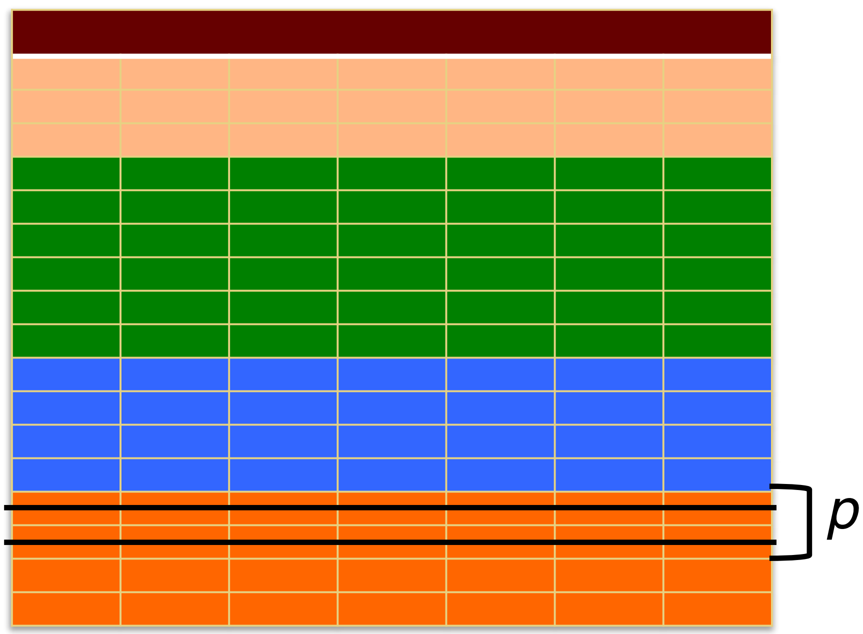


p = predicate

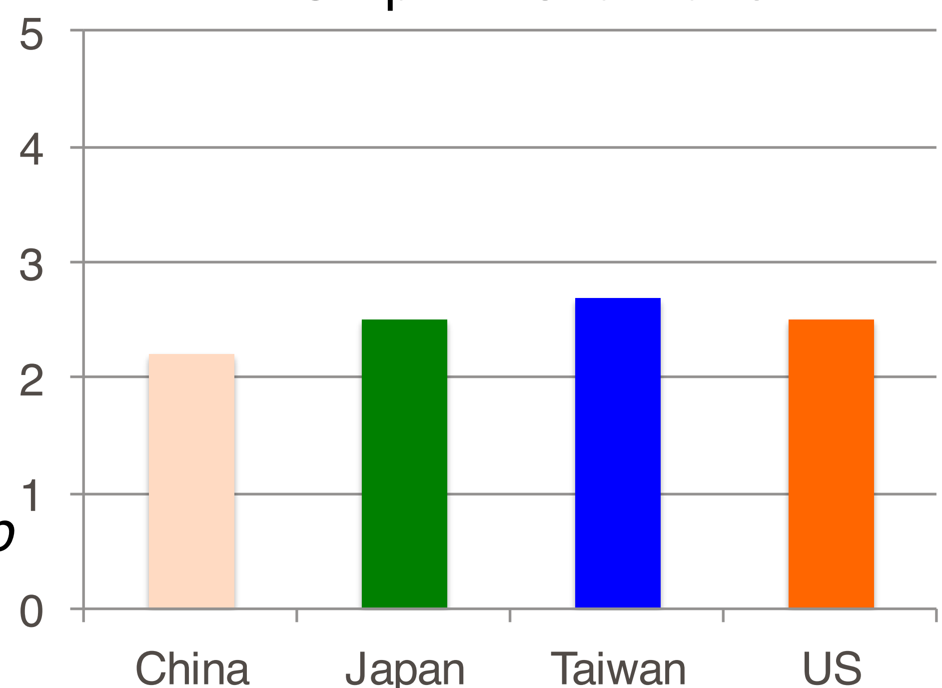
Definition of Why

Given an outlier group, find a *predicate* over the inputs that makes the output no longer an outlier.

i = Input Data



Output Visualization

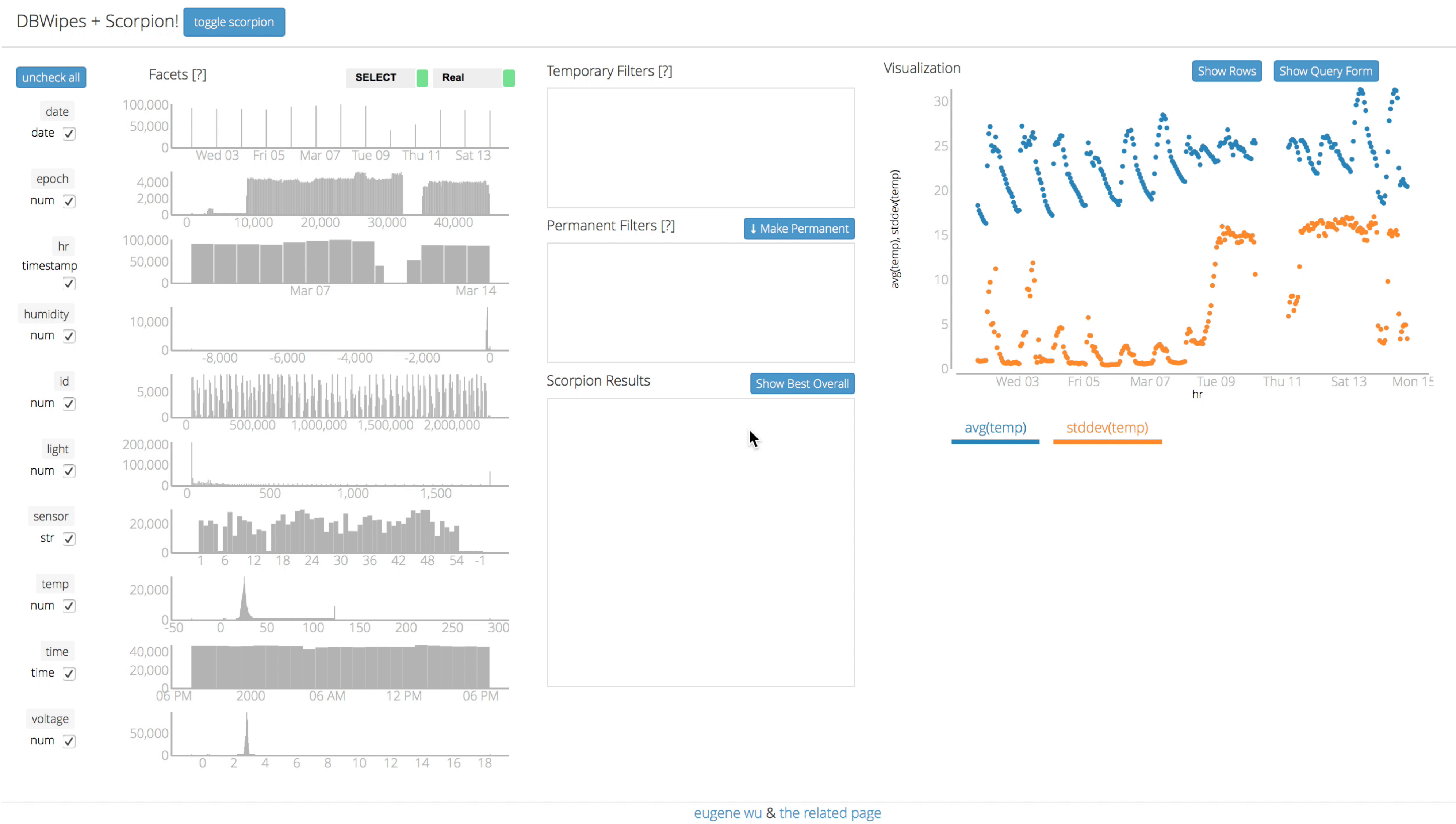


Removing the predicate makes US no longer an outlier

What are common properties of those records? *{Warren Buffet, Tim Cook}*

p: Job = CEO

Scorpion Demo



Existing Data Intensive Systems are a Poor Fit For Interactive Analytic Applications

Example: relational databases

- **Not optimized for interactivity**
  - a query that runs in a few seconds
  - disk-optimized (big pages, buffer p
  - ok for the optimizer to work for hun
  - synchronous APIs (JDBC)

All 3 examples employ custom data processing layers to circumvent these issues

- **Designed to perform well on a known workload**
  - careful physical tuning
- **Designed to be the "system of record"**
  - ➔ approximation bad!
- **(Traditional) focus on point lookups & transactional updates**
  - these hurt scan (analytic) performance
- ***Similar statements can be made about Hadoop & Spark***

Four Research Opportunities

1. Move away from “index first” and up-front load

Move away from index first

- **TPC-H Scale 10 Load Times on Postgres**

  (~10 GB data, on 4 core MacBook Pro w/ SSD)

  Load: 7 mins (23 MB/sec)

  Creating keys: 13 mins

  Indexing: 18 mins

  Untenable if just doing a first pass on the data

  Opportunity: index & partition data on the fly

  (Some work on avoiding loading too – see, e.g., Alagiannis et al, "NoDB", SIGMOD 2012)

Example: Database cracking

Index attributes as they are accessed, instead of up front!

Column A
13
16
4
9
2
12
7
1
19
3
14
11
8
6

Idreos et al. "Database Cracking", CIDR 2007.

Example: Adaptive Partitioning

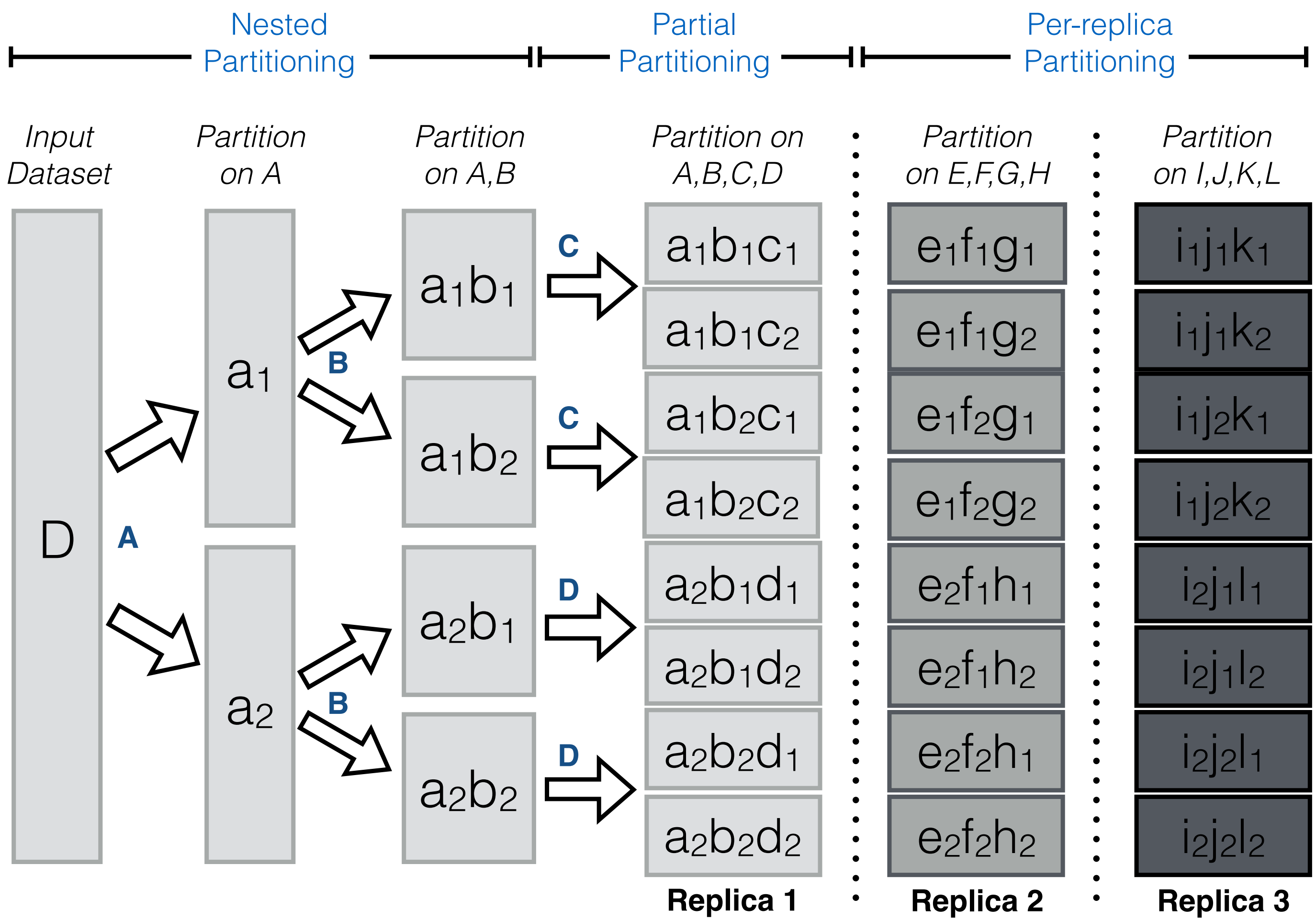
- **Data partitioning is key to good performance in modern parallel data systems**
  - Read just the partitions you need
  - Partition is expensive (requires shuffling data)

- **Challenge: how to partition?**
  - Typical choice: partition on frequently queried attributes
  - What if those aren't known?

- **Idea: adaptively partition data as it is queried**

w/ Alekh Jindal, Qui Nguyen, Anil Shanbhag , Aaron Elmore, Divy Agarawal, Jorge Quiane Ruiz

Example: Adaptive Partitioning



Whenever a query arrives, choose whether we should re-partition a block or not

Four Research Opportunities

1. Move away from “index first”
2. Build *analytic* engines for main memory

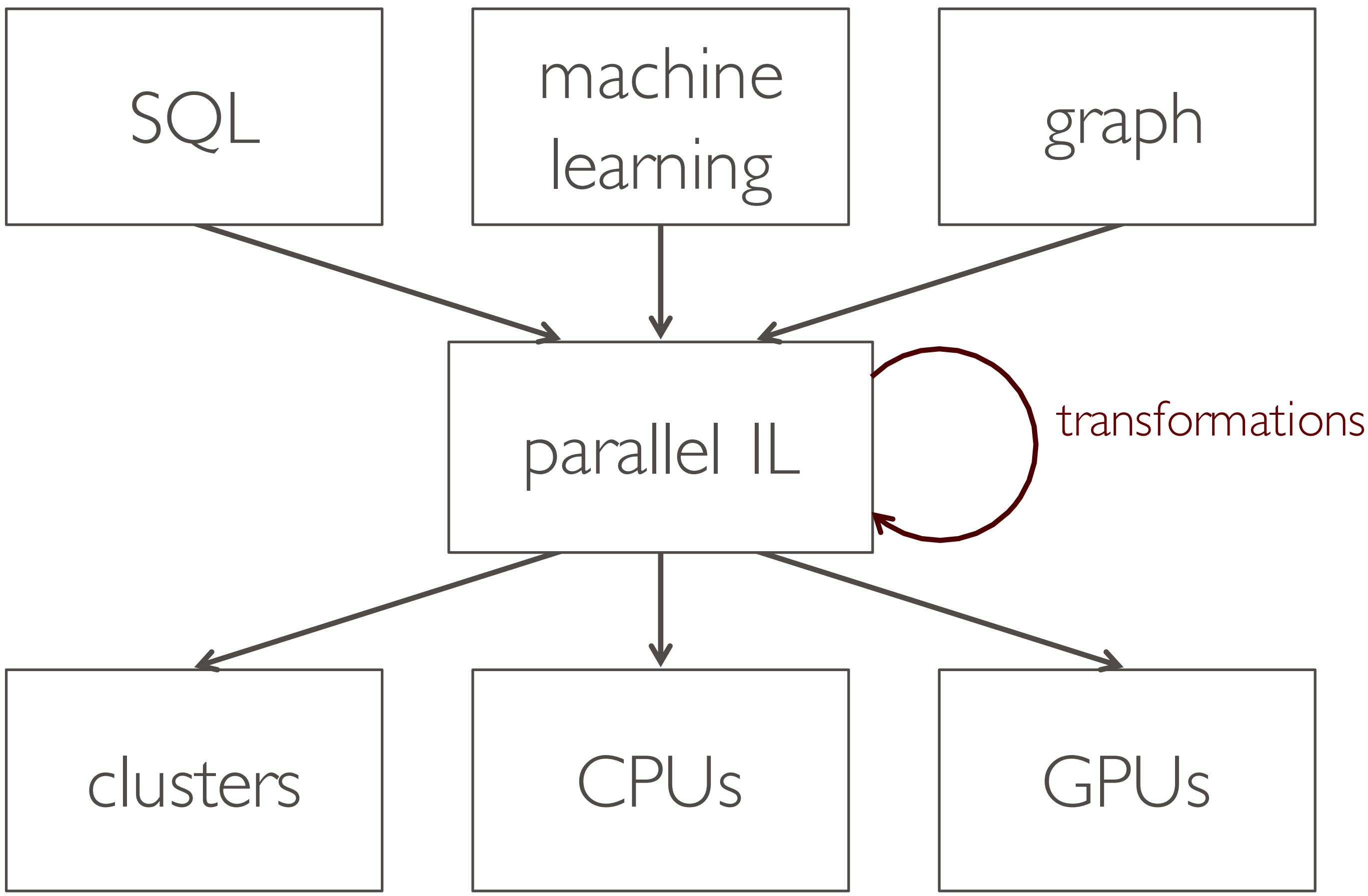
Analytic engines for main memory: Voodoo Parallel IL

with Oscar Moll, Holger Pirk, Yunming Zhang, Saman Amarasinghe, Matei Zaharia

- **To optimize across libraries automatically, need to express them in a common intermediate language**
- **Design a data-parallel IL that:**
  - Captures common data processing tasks
  - Allows rich transformations at the level of the IL
  - Maps efficiently to hardware (clusters, CPU, GPU)
- **Focus on main memory & interactive performance**

Related: Hyper (TU Munich)

The Goal



SQL
machine learning
graph
parallel IL
transformations
clusters
CPUs
GPUs

Example Transformations: Fusing

```
// library function
def scoreFit(data: vec[vec[float]], param: vec[float]) = {
  sum = [0, 0]
  for (d <- data) { sum += dot(d, param)**2 }
}
```

```
// user code
params = [[1, 1], [3, 2]]
for (p <- params) { scoreFit(data, param) }
```

```
for (p <- params) {
  sum = [0, 0]
  for (d <- data) {
    sum += dot(d, p) ** 2
  }
}
```

```
sums = [[0, 0], [0, 0]]
for (d <- data) {
  for ((p, i) <- params) {
    sum[i] += dot(d, p) ** 2
  }
}
```

Example Transformations: Data Representation

```
// select sum(salary) from users where state == “MA”
def query(users: vec[{name:str, salary:int, state:str}]) = {
  sum = 0
  for (u <- users) {
    if (u.state == “MA”) { sum += u.salary }
  }
}
```

```
// column-oriented execution
def query(name: vec[str], salary: vec[int], state: vec[str]) = {
  sum = 0
  for (i <- 0..len(users)) {
    if (state[i] == “MA”) { sum += salary[i] }
  }
}
```

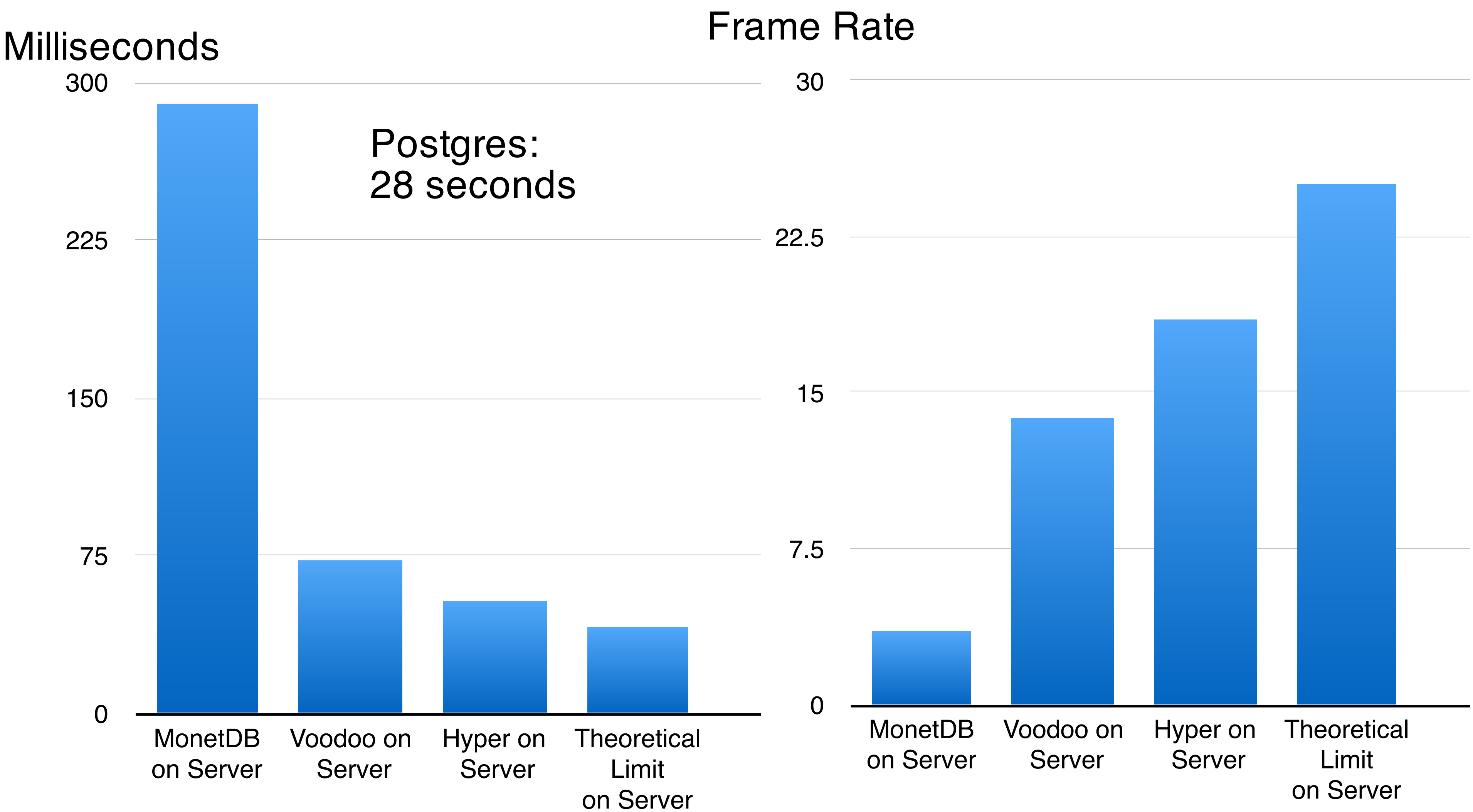
Voodoo Backend

- **Generates parallel code for a variety of hardware**
- **Takes as input an intermediate representation of *vectors***

```
// column-oriented execution
def query(name: vec[str], salary: vec[int], state: vec[str]) = {
  sum = 0
  for (i <- 0..len(users)) {
    if (state[i] == "MA") { sum += salary[i] }
  }
}
```

- **Hardware abstracted by vector size and number of parallel units**
  - Works for GPU, multicore, manycore
- **Currently acts as a drop in backend for MonetDB**

Voodoo Performance



TPC-H Query 6, Scale Factor 10 (6 GB scan)

Four Research Opportunities

1. Move away from "index first"
2. Build *analytic* engines for main memory
3. **Treat approximation as a first class citizen**
   – Exploit visual properties

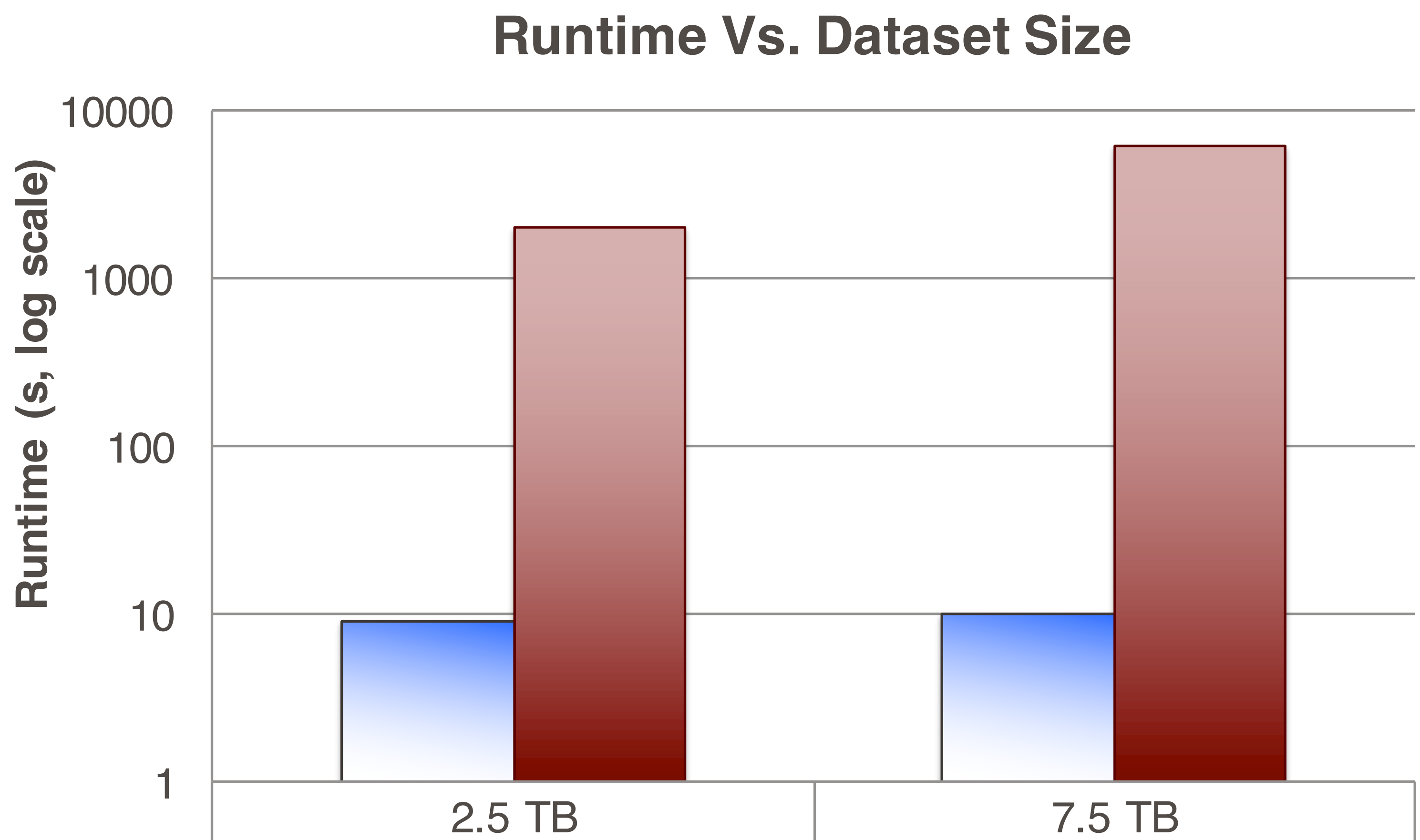
Approximate Data Systems

- **By operating on *samples* of data, can get big speedups**
  - Example: BlinkDB

BlinkDB

Exploratory analytics workload

42 queries, each of aggregates, groups, and filters on a different subset of attributes.



	2.5 TB	7.5 TB
BlinkDB (1% error)	9	10
Hive	2000	6000

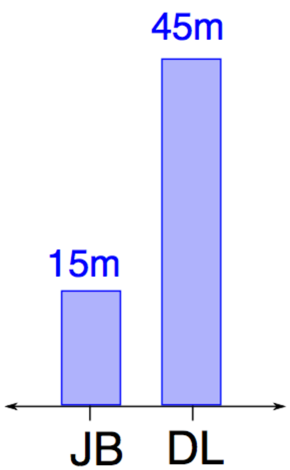
Approximate Data Systems

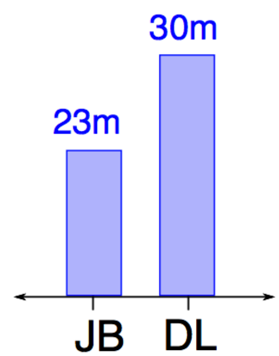
- **By operating on *samples* of data, can get big speedups**
  - Example: BlinkDB
- **Historically mildly popular database research topic**
  - AQUA, Control, SQL Server
  - Never seen much uptake
    - **DB users aren't comfortable with "close enough"**
- **Challenges:**
  - Queries over rare subgroups
    - **BlinkDB *stratifies* on popular attributes**
  - How to compute and maintain random samples
  - What type of sampling to use?

Visually-aware sampling

Correct ordering property

Original	Sample
$\mu_i < \mu_j$	$\nu_i < \nu_j$



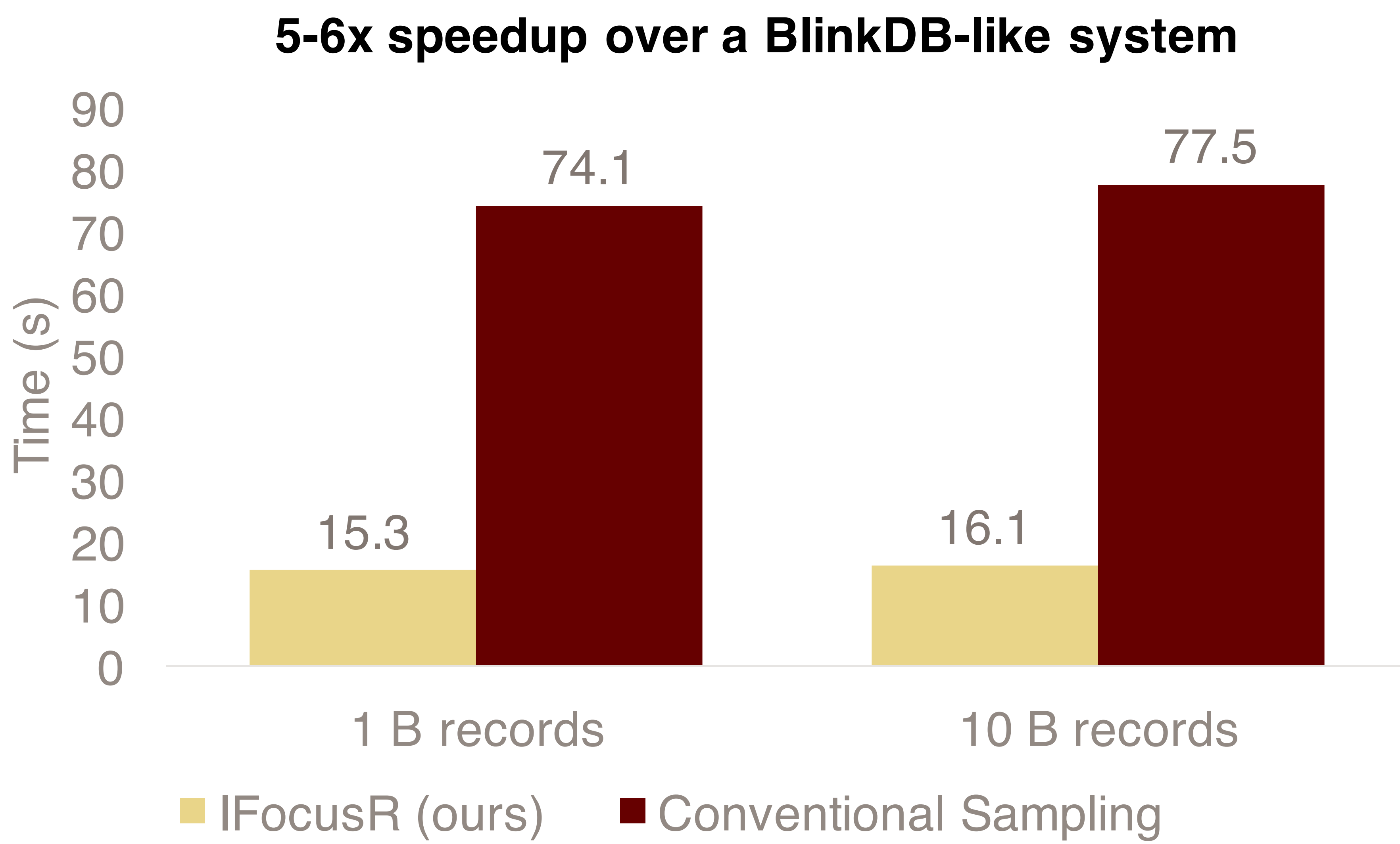


Algorithm sketch: sample groups whose confidence intervals overlap; don't sample others

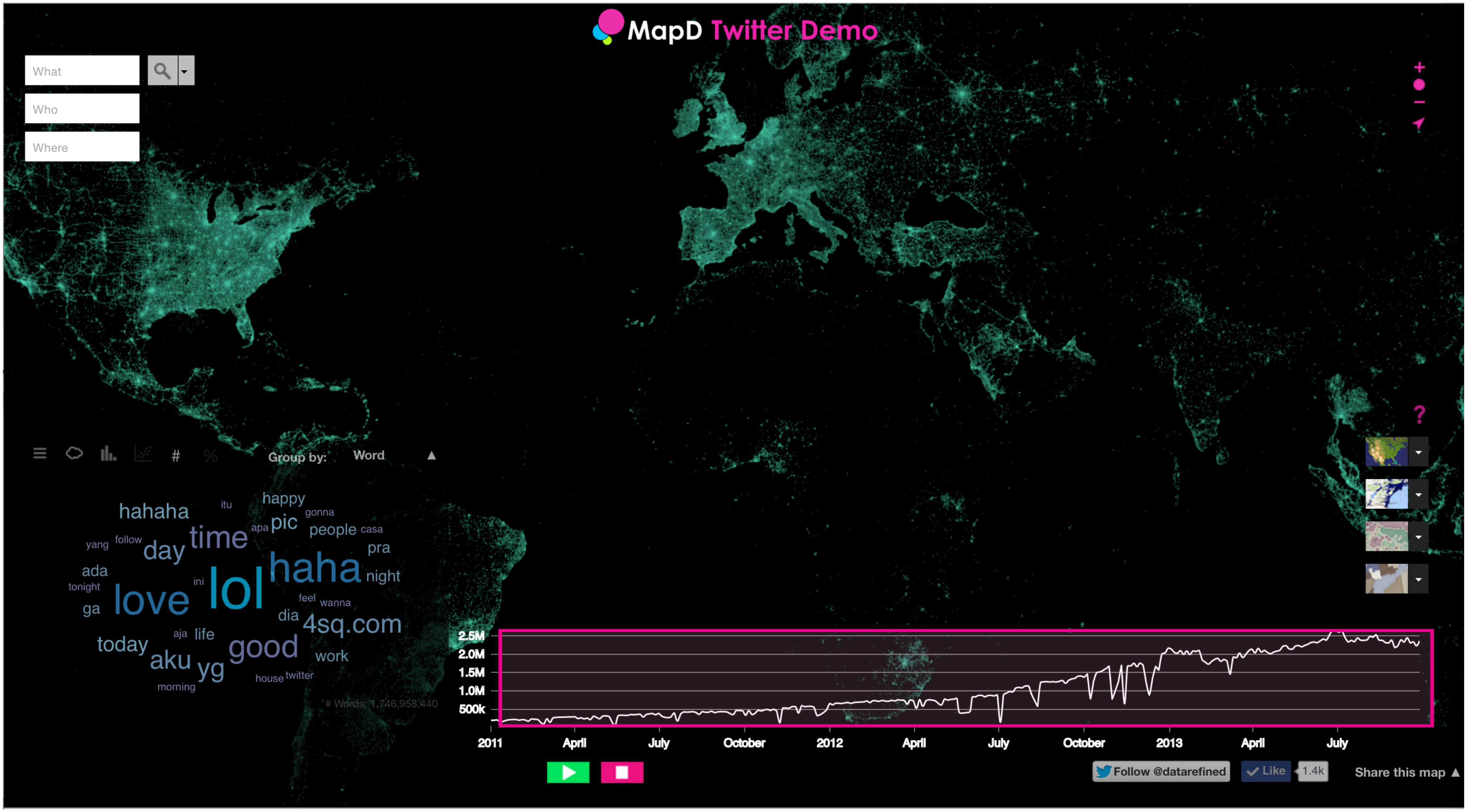
$$\mu_1 < \mu_2 < \cdots < \mu_k \quad \Rightarrow \quad \nu_1 < \nu_2 < \cdots < \nu_k$$

Kim et al, *Rapid Sampling For Visualizations with Order Guarantees* VLDB 2015

Visual Sampling Performance on Flight Dataset
Average Delay by Airline



Visual sampling: don't paint every pixel



Four Research Opportunities

1. **Move away from "index first"**
2. **Build *analytic* engines for main memory**
3. **Treat approximation as a first class citizen**
   - Exploit visual properties
4. **Develop new asynchronous interfaces**
   - "Linked views", where V1 updates when V2 changes
   - Incremental refresh of visualizations
     - **Ex. Meteor Framework: "optimistic UI"**

Conclusion

Interactive analytics is a new frontier

Huge performance gulf between current data processing systems (cloud-based or otherwise) and what is required, even on simple tasks

As demand for complex analytics and automated inferences/insight grows, **this gap will get worse**

This creates research opportunities

- In memory engines
- Visually aware approximate processing
- Load less, query more
- New interface abstractions