MAPD

Intro to MapD

Todd Mostak, CEO

November 6, 2017

Traditional GIS

Small, static datasets

- **Data mapping static assets:** Natural resource assets, land parcels, transportation grid, utilities
- **Human-curated:** Significant time spent precisely measuring and mapping the location of these assets
- **Nature of data limits scale**: Hand-curation means data size measured in thousands, not billions

Traditional GIS

Geospatial is key

- **Emphasis on geospatial operations:** Which light poles are within 5m of a road? How many land parcels lie in a flood zone?
- **Limited number of GIS experts using data**: Technical nature of data and problem domains in combo with specialized tools limit access to a small group of trained experts
- **Use of dedicated GIS platforms:** Only a small set of tools (i.e. ESRI) capable of depth of GIS operations needed

New Geoanalytic Challenges…

New Geoanalytic Challenges

Large, high velocity data

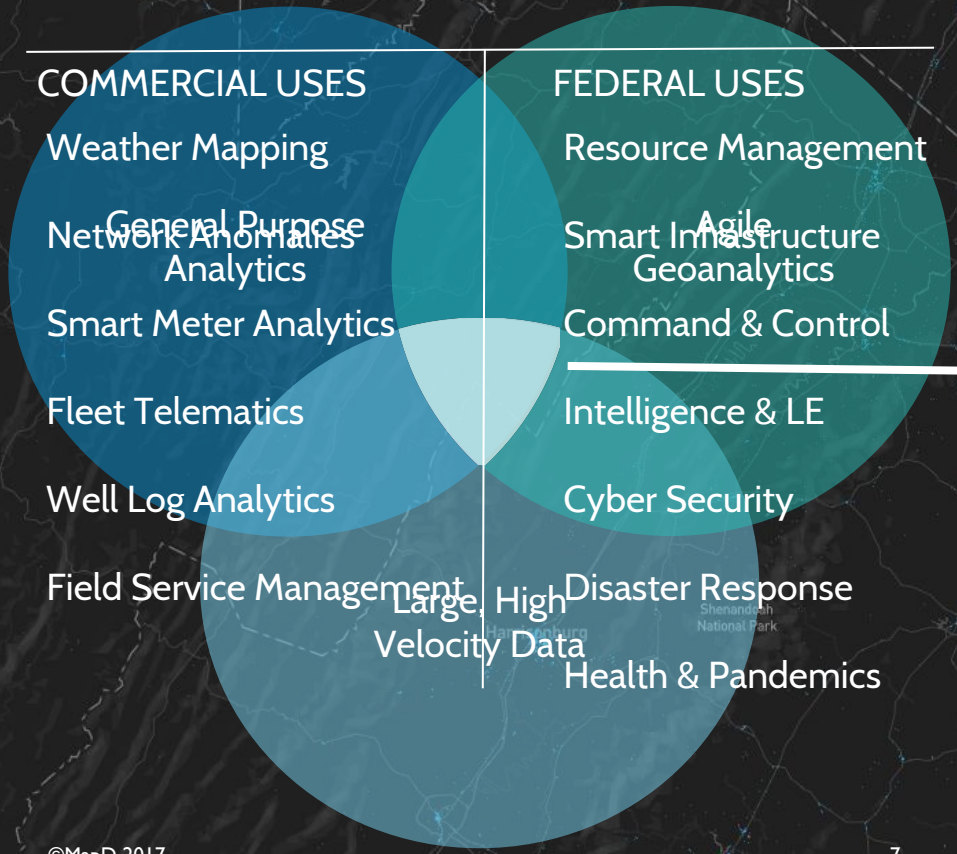
- **Data is primarily sensor-generated:** Cell handsets, vehicle telemetry, LIDAR, satellite, social media
- **Data size is unbounded**, no longer limited by the rate at which humans can create it.
- **Data velocity can be extremely high**, with billions of sources each reporting multiple times per second.

New Geoanalytic Challenges

Geolocation in context

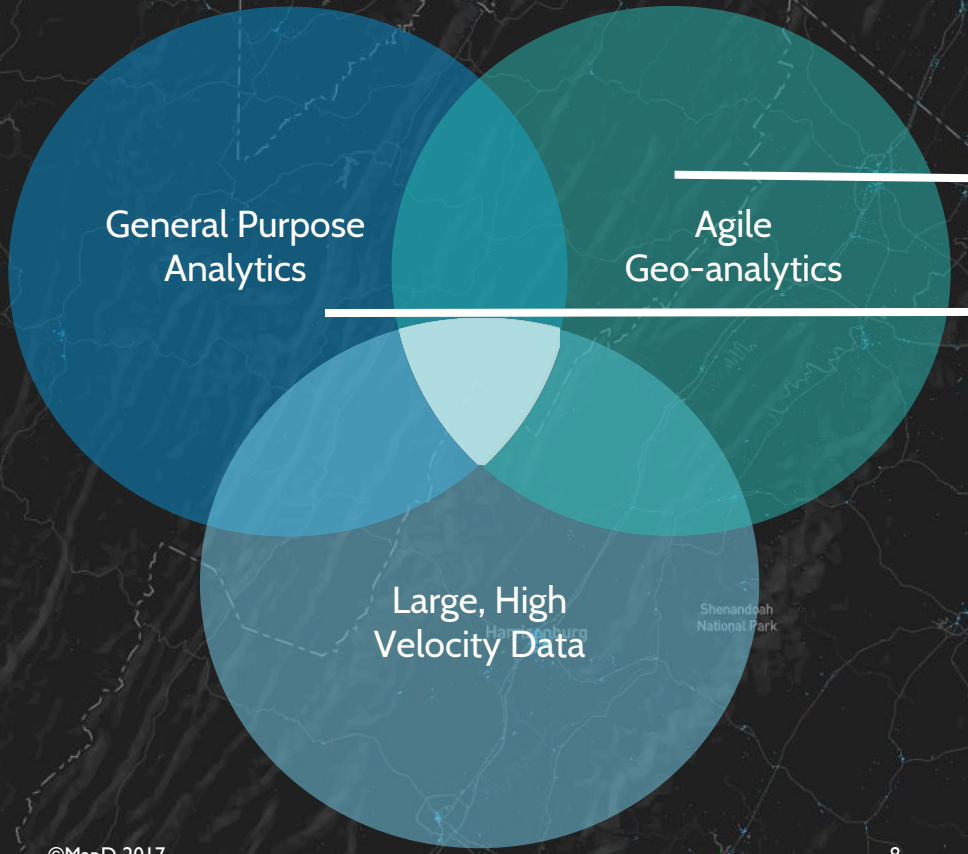
- **Geo is just one attribute of data:** Other attributes can be equally as important
- **Lower complexity of geospatial analytics:** Often limited to visualization and measuring distance and spatial overlap
- **New audience of analysts and data scientists:** Deep GIS background often no longer required

…Require New Spatially Enriched Analytics



What are some of the top use cases?

Breaking Point



Traditional GIS

Traditional BI

"There's no way to render everything at once."

"I downsample for a living."

"Latency is crushing us. These insights are too old to be useful."

" Wait for it…."

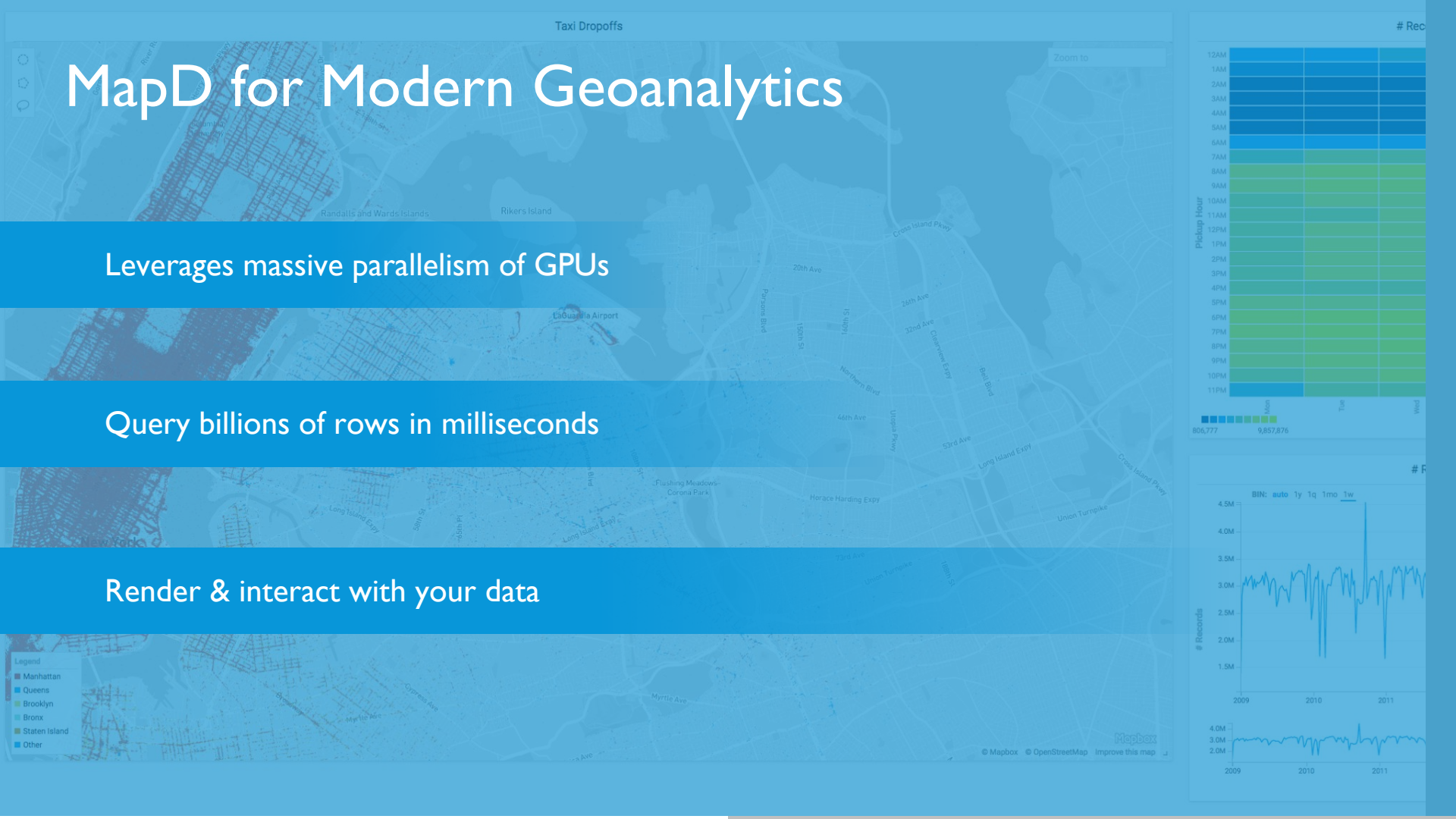
"I have to write a support ticket and wait days for a new dashboard."

MapD for Modern Geoanalytics

Leverages massive parallelism of GPUs

Query billions of rows in milliseconds

Render & interact with your data



DEMO

How We Do It

Blindingly Fast

- Parallelize SQL for GPUs
- Three-tier data caching
- LLVM Compilation Engine

Native Rendering

Vega API for declarative rendering of data in-situ on the GPU

Intuitive to Use

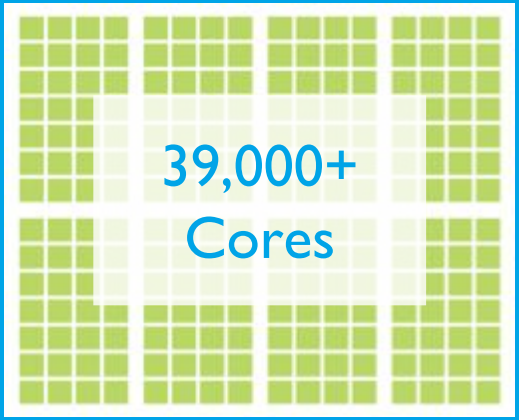
MapD Immerse provides drag-and-drop interactive visualization at scale

Core Density Makes a Huge Difference

CPU Processing

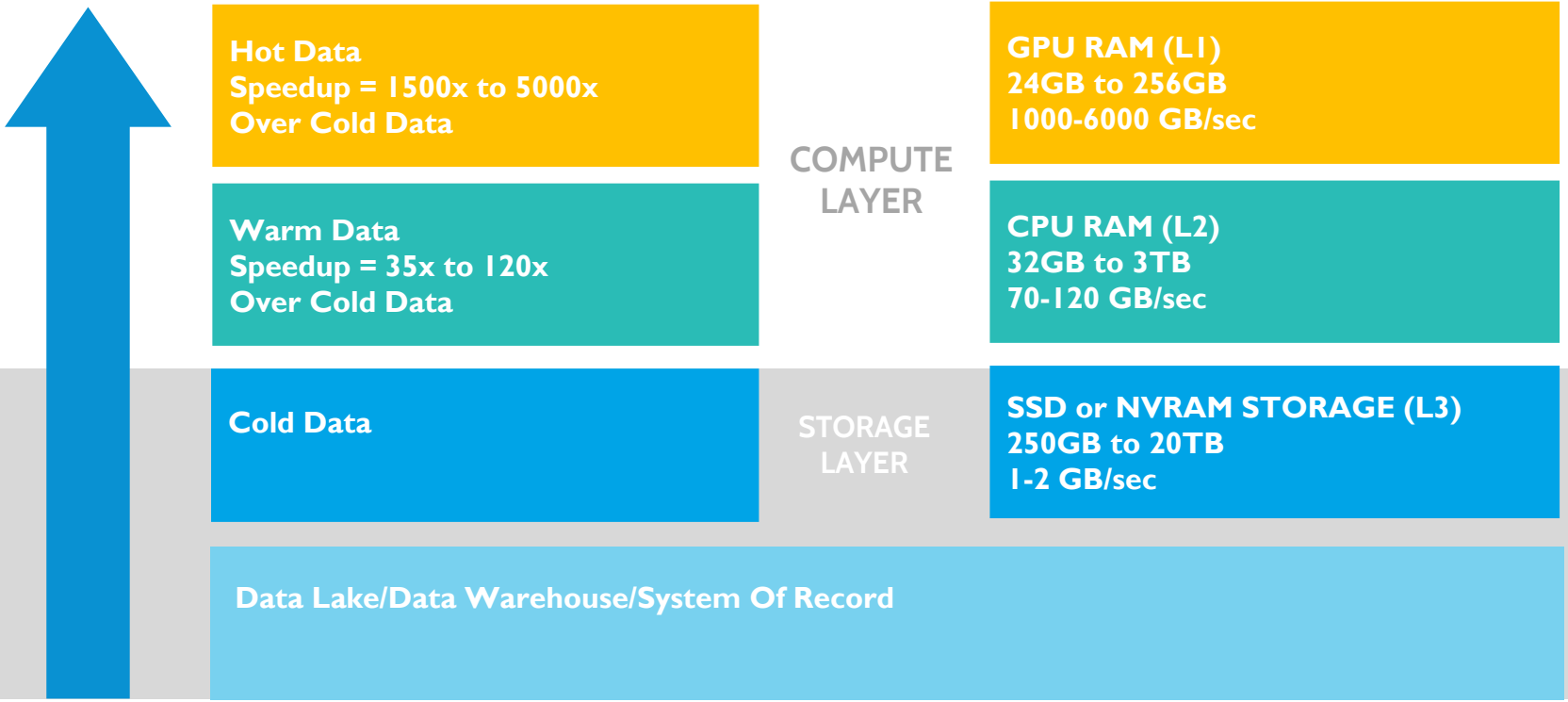
20 Cores

GPU Processing



Keeping Data Close to Compute

MapD maximizes performance by optimizing memory use



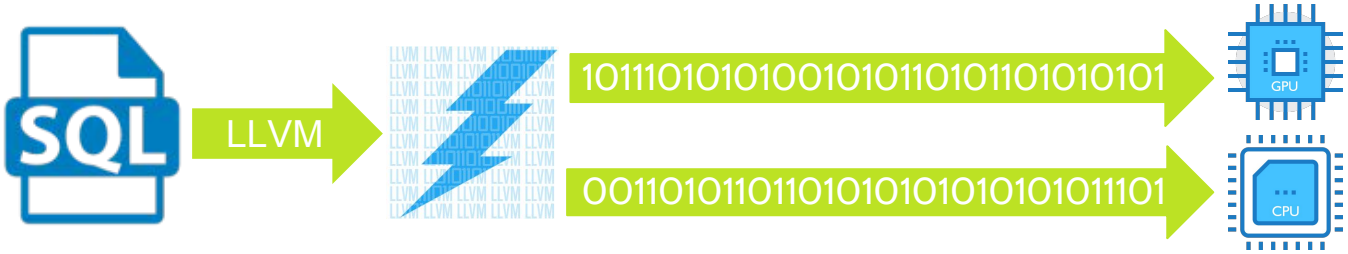
Query Compilation with LLVM

Traditional DBs can be highly inefficient

- each operator in SQL treated as a separate function
- incurs tremendous overhead and prevents vectorization

MapD compiles queries w/LLVM to create one custom function

- Queries run at speeds approaching hand-written functions
- LLVM enables generic targeting of different architectures (GPUs, X86, ARM, etc).
- Code can be generated to run query on CPU and GPU simultaneously



MapD Benchmarks

Blogger Mark Litwintschik benchmarked MapD on a billion-row taxi data set and found it to be **orders-of-magnitude faster than the fastest CPU databases**

Query 1	Query 2	Query 3	Query 4	Setup
0.021	0.053	0.165	0.51	MapD & 8 Nvidia Pascal Titan Xs
0.027	0.083	0.163	0.891	MapD & 8 Nvidia Tesla K80s
0.028	0.2	0.237	0.578	MapD & 4-node g2.8xlarge cluster
0.034	0.061	0.178	0.498	MapD & 2-node p2.8xlarge cluster
0.036	0.131	0.439	0.964	MapD & 4 Nvidia Titan Xs
0.051	0.146	0.047	0.794	kdb+/q & 4 Intel Xeon Phi 7210 CPUs
0.762	2.472	4.131	6.041	BrytlytDB & 2-node p2.16xlarge cluster
1.034	3.058	5.354	12.748	ClickHouse, Intel Core i5 4670K
1.56	1.25	2.25	2.97	Redshift, 6-node ds2.8xlarge cluster
2	2	1	3	BigQuery
4	4	10	21	Presto, 50-node n1-standard-4 cluster
6.41	6.19	6.09	6.63	Amazon Athena
8.1	18.18	n/a	n/a	Elasticsearch (heavily tuned)
10.19	8.134	19.624	85.942	Spark 2.1, 11 x m3.xlarge cluster w/ HDFS
11	10	21	31	Presto, 10-node n1-standard-4 cluster
14.389	32.148	33.448	67.312	Vertica, Intel Core i5 4670K
34.48	63.3	n/a	n/a	Elasticsearch (lightly tuned)
35	39	64	81	Presto, 5-node m3.xlarge cluster w/ HDFS
43	45	27	44	Presto, 50-node m3.xlarge cluster w/ S3
152	175	235	368	PostgreSQL 9.5 & cstore_fdw
264	313	620	961	Spark 1.6, 5-node m3.xlarge cluster w/ S3

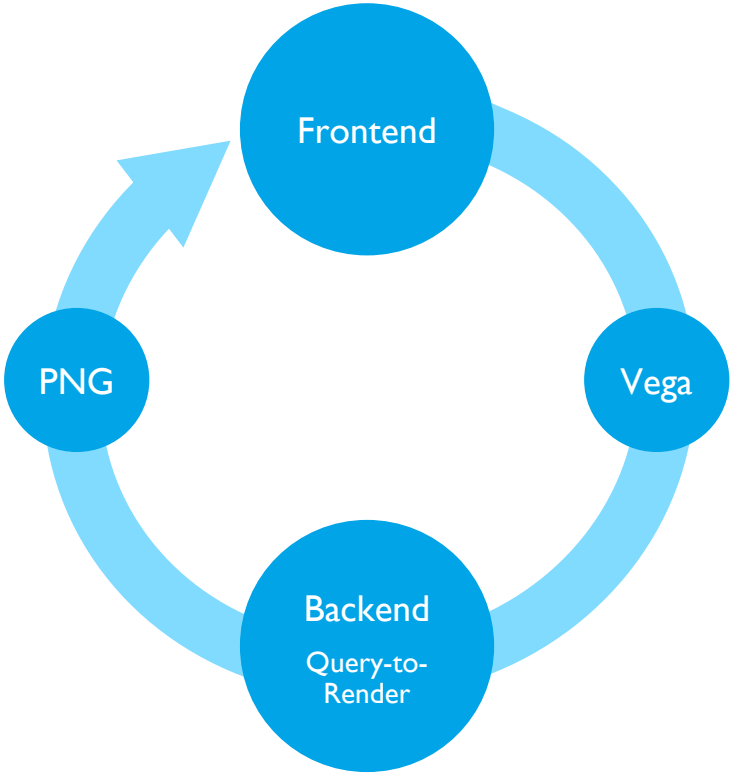
Source: http://tech.marksblogg.com/benchmarks.html

MapD Core: Comparative Query Acceleration*

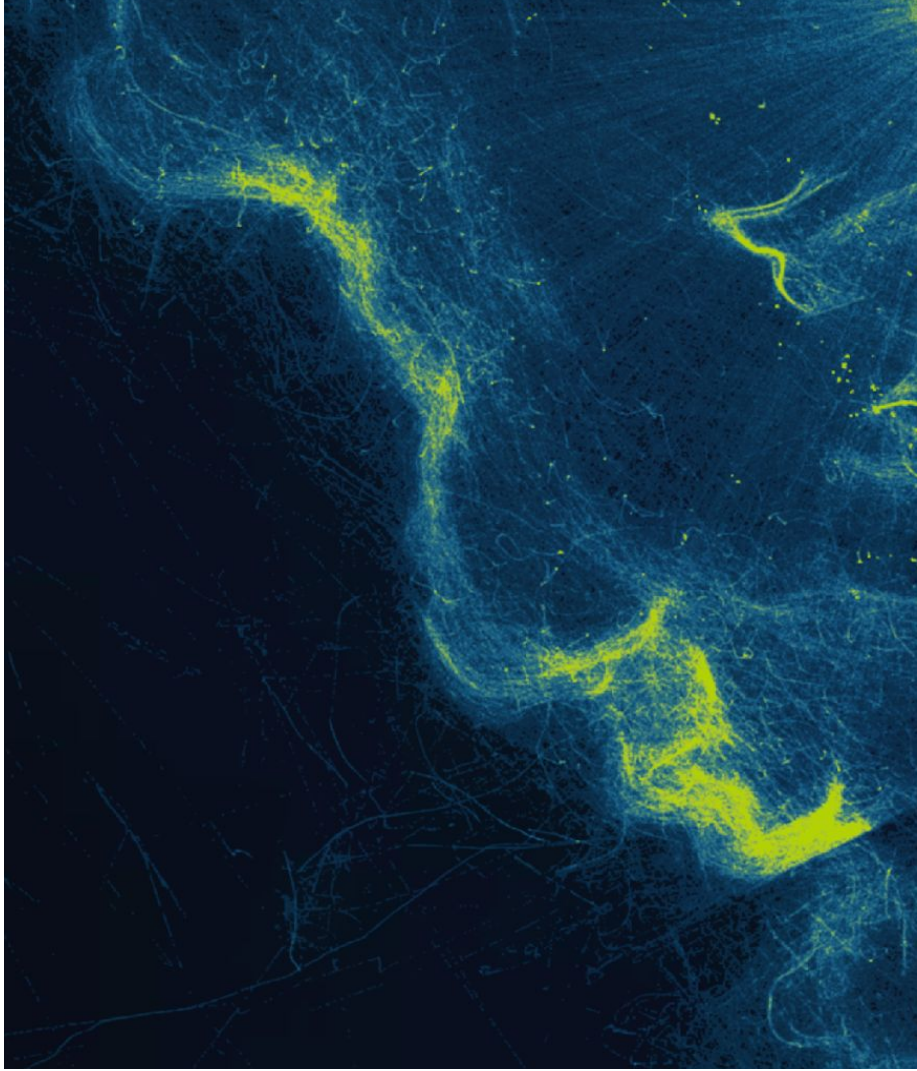
System	Q 1	Q 2	Q 3	Q 4
BrytlytDB & 2-node p2.16xlarge cluster	36x	47x	25x	12x
ClickHouse, Intel Core i5 4670K	49x	58x	32x	25x
Redshift, 6-node ds2.8xlarge cluster	74x	24x	14x	6x
BigQuery	95x	38x	6x	6x
Presto, 50-node n1-standard-4 cluster	190x	75x	61x	41x
Amazon Athena	305x	117x	37x	13x
Elasticsearch (heavily tuned)	386x	343x	n/a	n/a
Spark 2.1, 11 x m3.xlarge cluster w/ HDFS	485x	153x	119x	169x
Presto, 10-node n1-standard-4 cluster	524x	189x	127x	61x
Vertica, Intel Core i5 4670K	685x	607x	203x	132x
Elasticsearch (lightly tuned)	1,642x	1,194x	n/a	n/a
Presto, 5-node m3.xlarge cluster w/ HDFS	1,667x	735x	388x	159x
Presto, 50-node m3.xlarge cluster w/ S3	2,048x	849x	164x	86x
PostgreSQL 9.5 & cstore_fdw	7,238x	3,302x	1,424x	722x
Spark 1.6, 5-node m3.xlarge cluster w/ S3	12,571x	5,906x	3,758x	1,884x

**All speed comparisons are to the "MapD & 1 Nvidia Pascal DGX-1" benchmark*

Server-Side Rendering

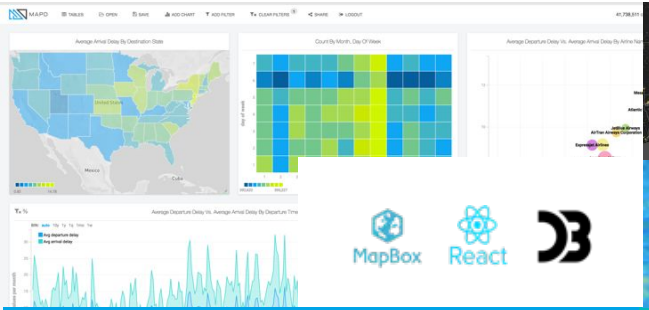


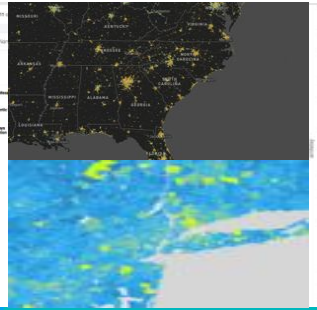
Ultra-fast CUDA->OpenGL->PNG pipeline for declarative rendering

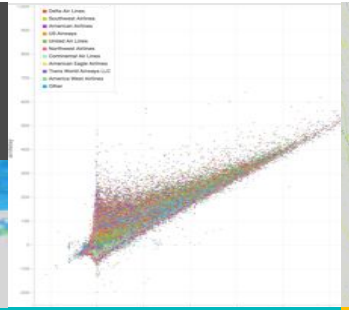


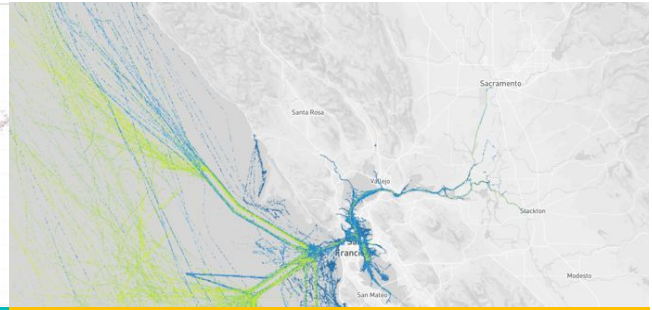
MapD Immerse

Using a hybrid approach to interactive visualization at scale









Basic charts are **frontend** rendered using D3, leveraging fast SQL of MapD Core

Scatterplots, pointmaps + polygons are **backend** rendered using the Rendering Engine on GPUs

Geo-Viz is **composited** over a frontend rendered basemap

Three Ways to Get Started

OPEN-SOURCE

GitHub Download

COMMUNITY

Website Download

AWS Cloud

ENTERPRISE

Contact MapD Sales

AWS Cloud

About MapD

Originated from MIT

$37 Million in funding

Used By 100+ Global Orgs

Open Source Community