BigDAWG: Managing Heterogenous Data and Streaming

Dr. Vijay Gadepally
vijayg@ll.mit.edu

October 2016

Massachusetts Institute of Technology

With contributions from: Jeremy Kepner (MIT), Albert Reuther (MIT), Nesime Tatbul (intel/MIT), Mike Stonebraker (MIT), Stan Zdonik (Brown)

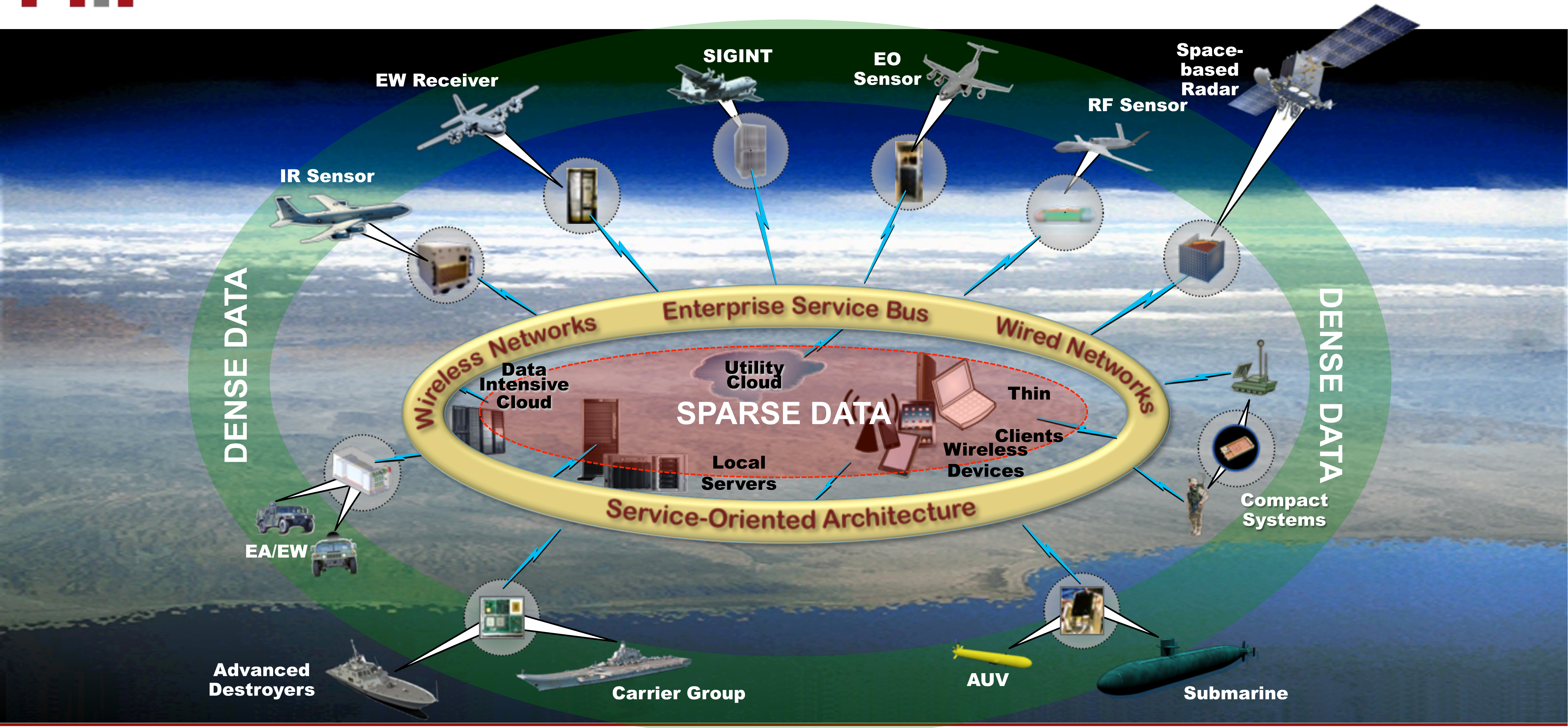
Acknowledgements

Outline

- Introduction and Background
- BigDAWG
  - What it is
  - BigDAWG Initial Results
  - BigDAWG in Action: Ocean Genomics
  - S-Store Streaming Engine
- Summary and Future Work

BigDAWG = Big Data Working Group

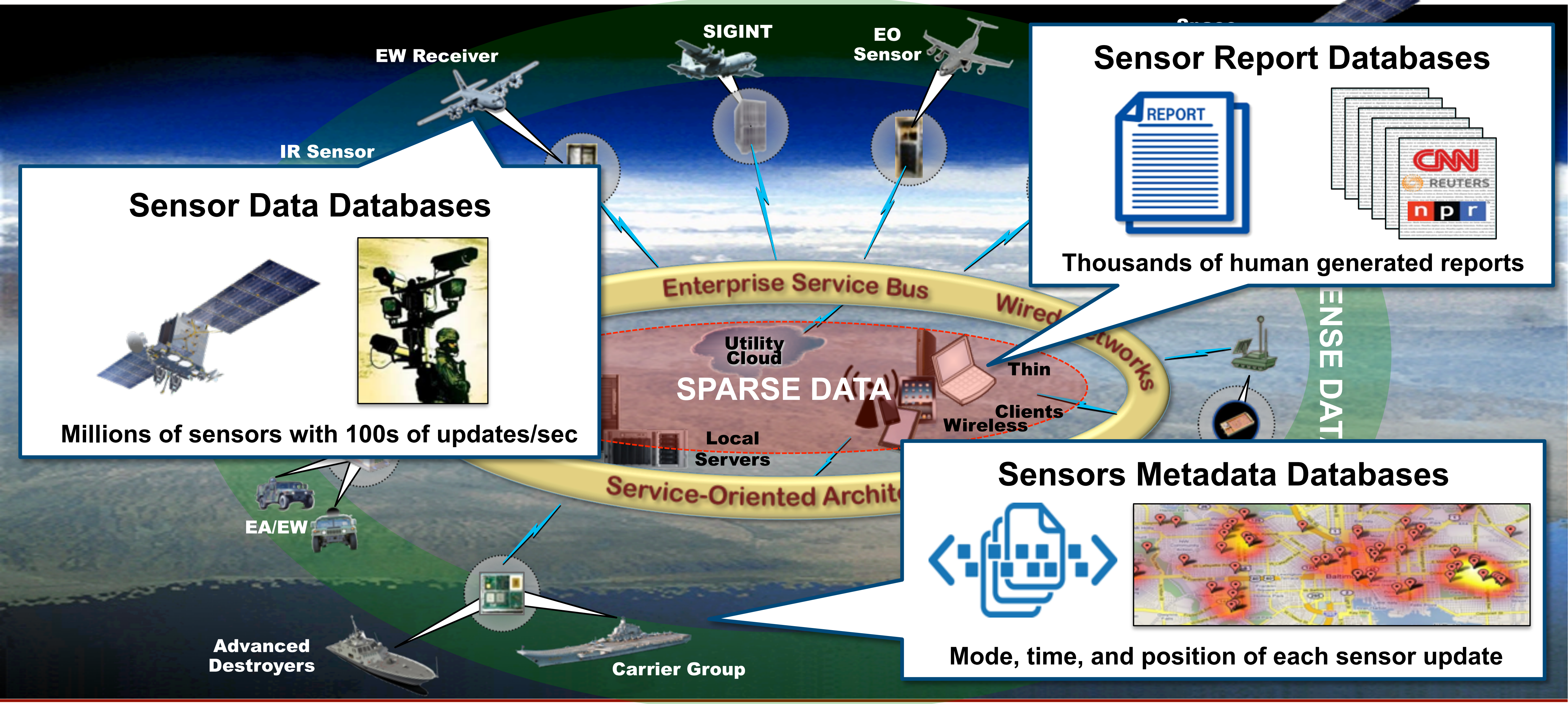
DoD Big Data



AUV = Autonomous Underwater Vehicle
EA = Electronic Attack
EW =Electronic Warfare,

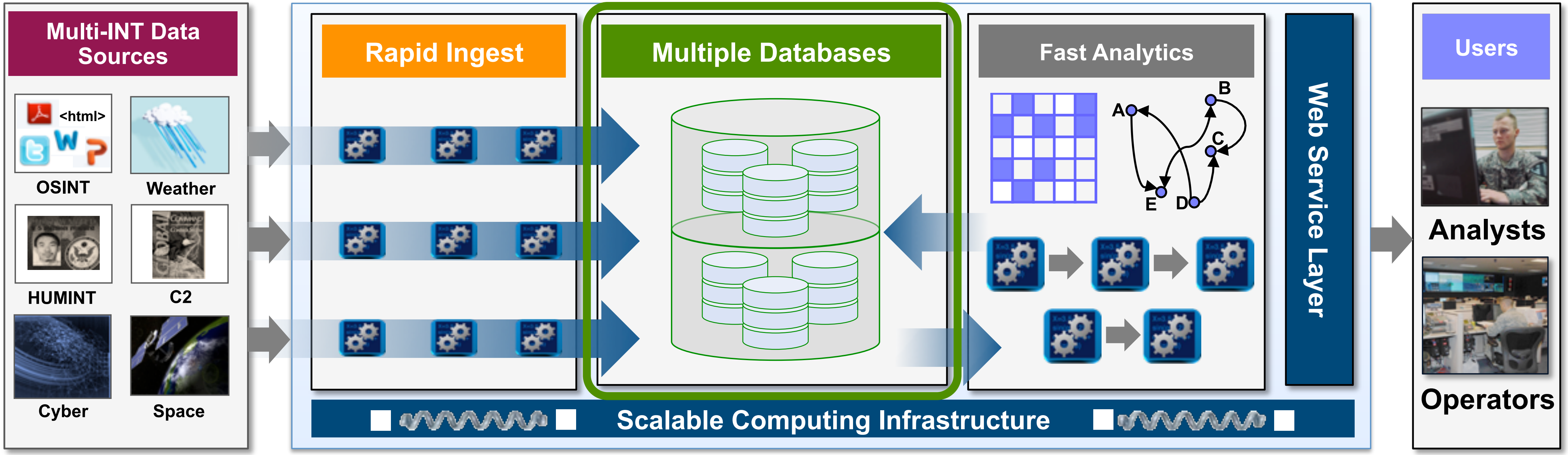
DoD Big Data

- Hundreds of Stovepiped Databases -



AUV = Autonomous Underwater Vehicle

Common Processing Architecture



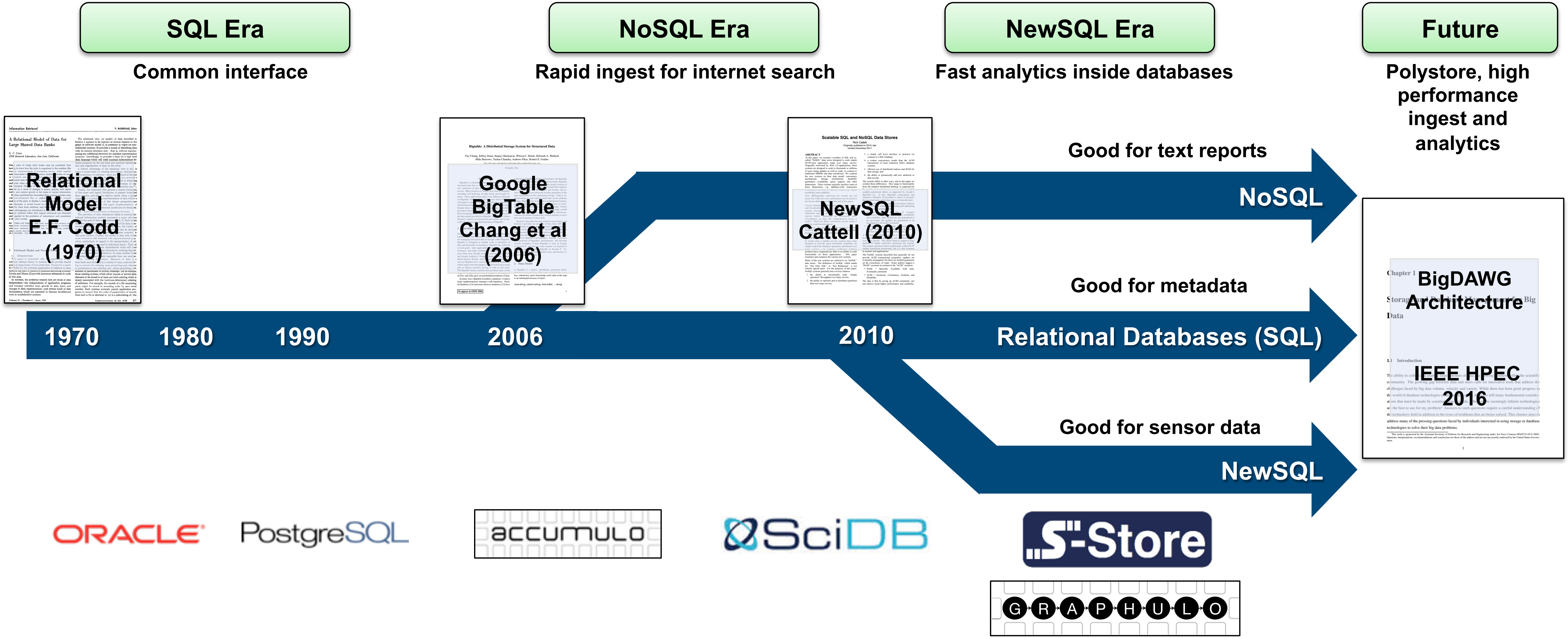
Government Database Challenges

- **Sustaining rapid database ingest**
- **Fast database analytics**
- **Querying multiple diverse databases**

LLSuperCloud: Sharing HPC Systems for Diverse Rapid Prototyping, Reuther et al, IEEE HPEC 2013

OSINT=Open Source Intelligence
HUMINT=Human Intelligence

Modern Database Paradigm Shifts



SQL = Structured Query Language
NoSQL = Not only SQL

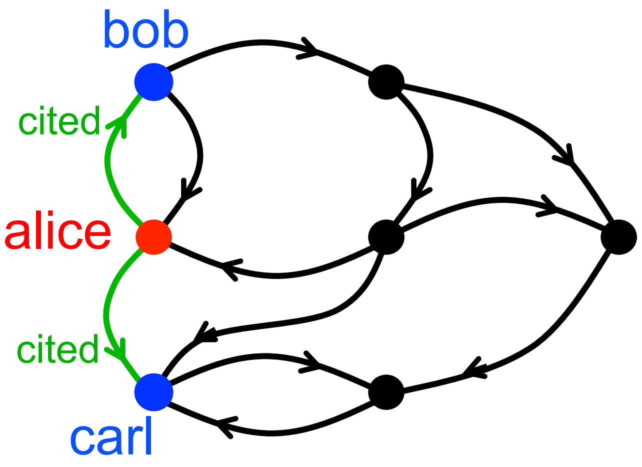
Polystore Database Challenge
-Providing a Common Mathematic Framework-

SQL
Set Operations

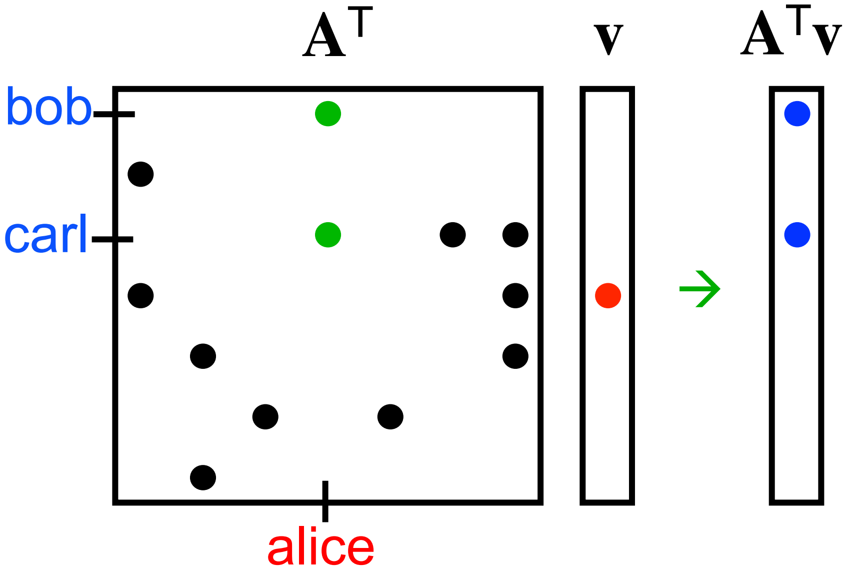
	from	link	to
001	alice	cited	bob
002	bob	cited	alice
003	alice	cited	carl

```
SELECT
WHERE from=alice
```

NoSQL
Graph Operations



NewSQL
Linear Algebra



Associative Array Algebra Provided a Unified Mathematics for SQL, NoSQL, NewSQL

$\mathbf{A} = \mathbb{S}^{N \mathrm{x} M}(\mathbf{i},\mathbf{j},\mathbf{v})$ $\quad (\mathbf{i},\mathbf{j},\mathbf{v}) = \mathbf{A}$ $\quad \mathbf{C} = \mathbf{A} \oplus \mathbf{B}$ $\quad \mathbf{C} = \mathbf{A} \otimes \mathbf{C}$ $\quad \mathbf{C} = \mathbf{A}\,\mathbf{B} = \mathbf{A} \oplus.\otimes \mathbf{B}$

Operations in All Representations are Equivalent

The World of Big Data

- **Data comes in all shapes and sizes**
  - **Unstructured data**
  - **Relational data**
  - **Images**
  - **Time series**

Why force all data to fit into a single data store?

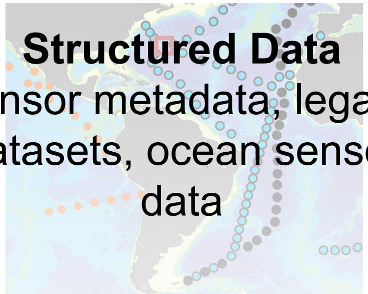
Leave data in the storage engine that matches the data …. A concept we call ***Polystore***

Exemplary Problems

	SQL	NoSQL	NewSQL
Ocean Genomics	**Structured Data** Sensor metadata, legacy datasets, ocean sensor data	**Reports** Field reports, analysts reports, memos, news articles, social media	**Sequence Data** Genomic sequences
MIMIC II test dataset*	**Structured Data** Demographic information, lab test results, hospital accounting records	**Freeform Text Data** Caregiver (doctor/nurse) notes and test reports	**Physiological Signals** Electrocardiogram (ECG) traces, arterial blood pressure monitoring, pulse oximeter readings

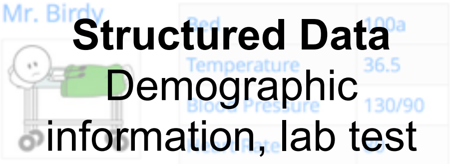
Ocean Genomics

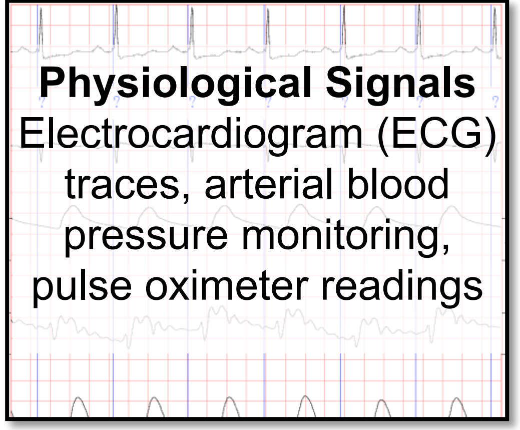
- Very large (multiple TB)
- Contains mix of different types of data from collected from 1000s of readings of ocean water samples



MIMIC II test dataset*

- >3 terabytes (TB) total
- 1000s of intensive care unit patients from 2001-08





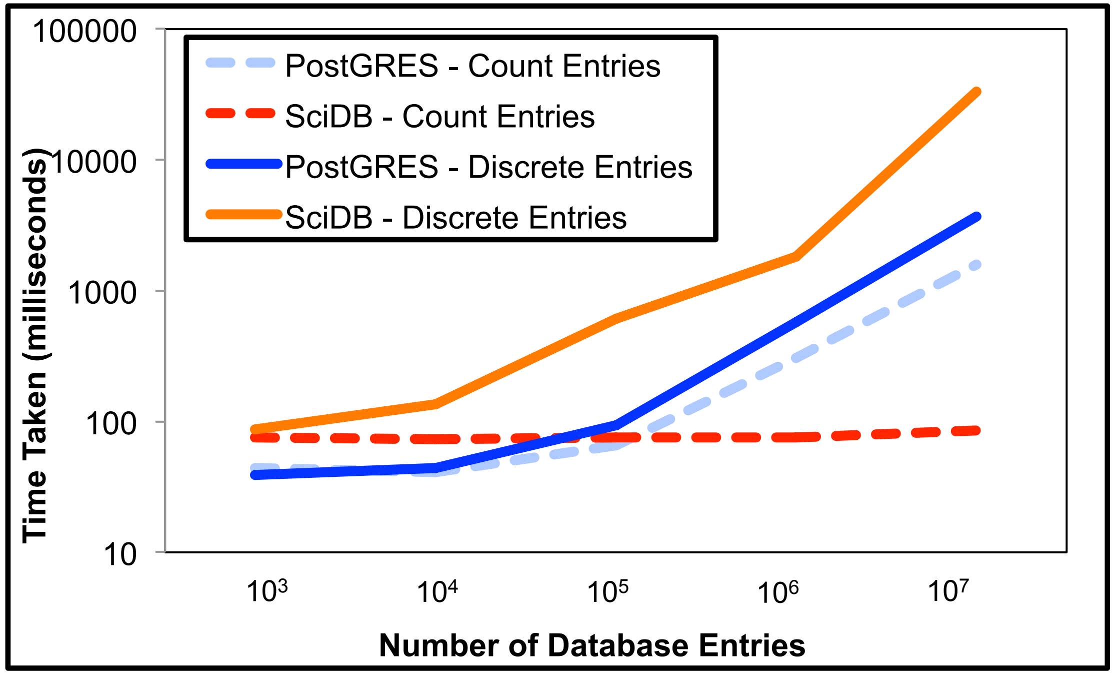
* *Multiparameter Intelligent Monitoring in Intensive Care II (MIMIC-II): A public-access intensive care unit database*, Saeed, et al., Crit Care Med. 2011

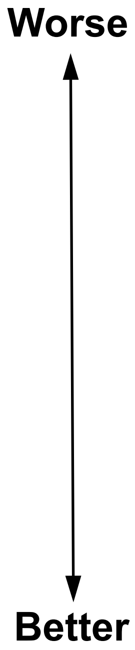
The Case for Polystores

- **Historically:**
  - Federated Databases
    - Mapping disparate database management systems via a single federated interface
    - Characteristics: Single query language (often SQL), single data model (often relational)
    - Examples: Garlic, R, IBM DB2
  - Parallel Databases
    - A single logical database or tables divided over multiple computing elements
    - Examples: SciDB (Array model), Teradata (Relational model)
- **Currently:**
  - Increasing need to support analysis of diverse data sources
  - "One size does not fit all" – no single database management system that supports high performance on all kinds of data
- **Polystore tenets:**
  - There is no single query language to rule them all
  - Complete functionality of underlying database management systems is required

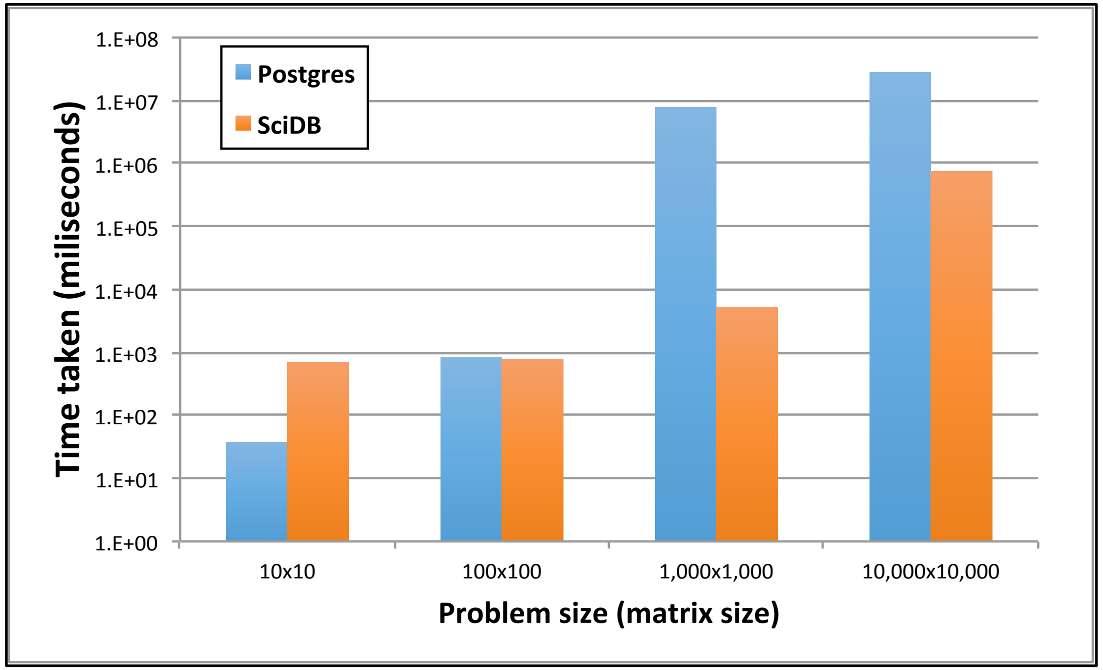
One Size Does Not Fit All
-Quantified for Common DB Operations-

Typical DB Operations





Matrix Multiplication Operations



Count and Find Operations

- **SQL database (PostgreSQL) better for some operations than Array database (SciDB)**

Matrix Multiplication

- **Array database (SciDB) faster than a SQL database (PostgreSQL)**

BigDAWG Architecture: VLDB’15, HPEC’16b

Outline

- **Introduction and Background**
- **BigDAWG**
  - **What it is**
  - **BigDAWG Initial Results**
  - **BigDAWG in Action: Ocean Genomics**
  - **S-Store Streaming Engine**
- **Summary and Future Work**

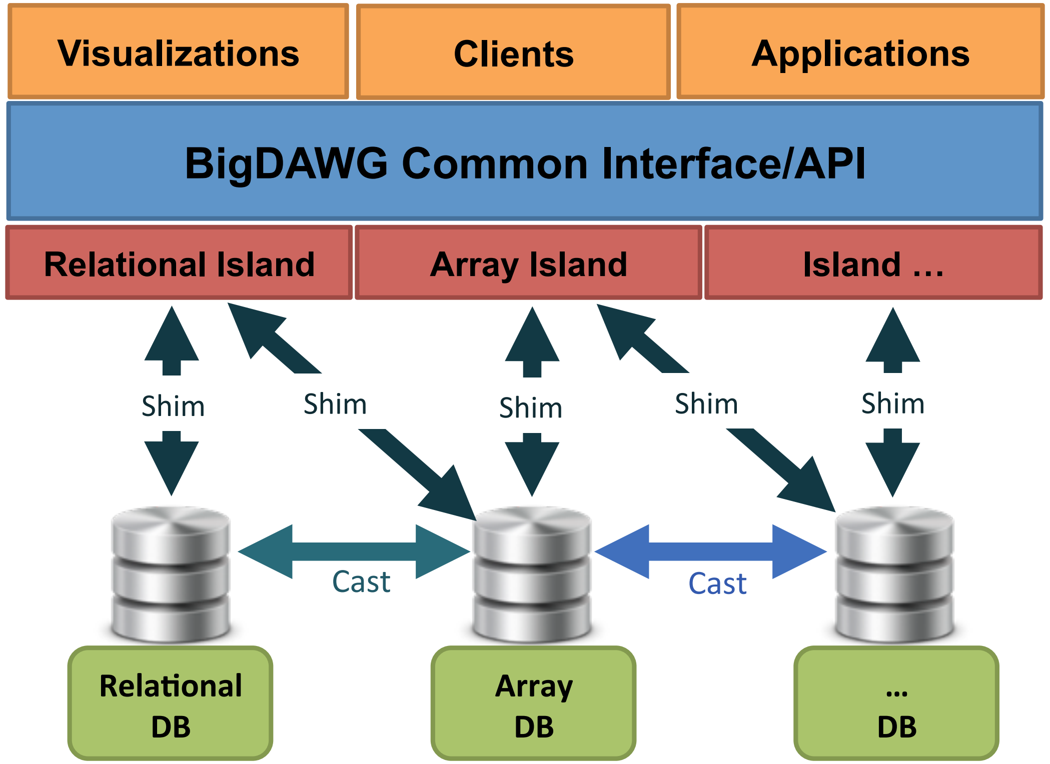
BigDAWG = Big Data Working Group

Our Approach: BigDAWG (the Big Data Working Group)

- Data Analytics & Processing Platforms
  - New platforms for storing and processing "big data"
- Scalable Math and Algorithms
  - Implementing parallel algorithms that scale to petabytes on thousands of machines
- Visualization
  - Presenting very large, high rate data sets
- Hardware architecture
  - Exploiting new advances in hardware
- Integration Across Multiple Data Processing Systems
- Benchmarks & Testbeds – Medical data, Oceanographic Data

BigDAWG is a large scale project meant to change the way we interact with very large datasets

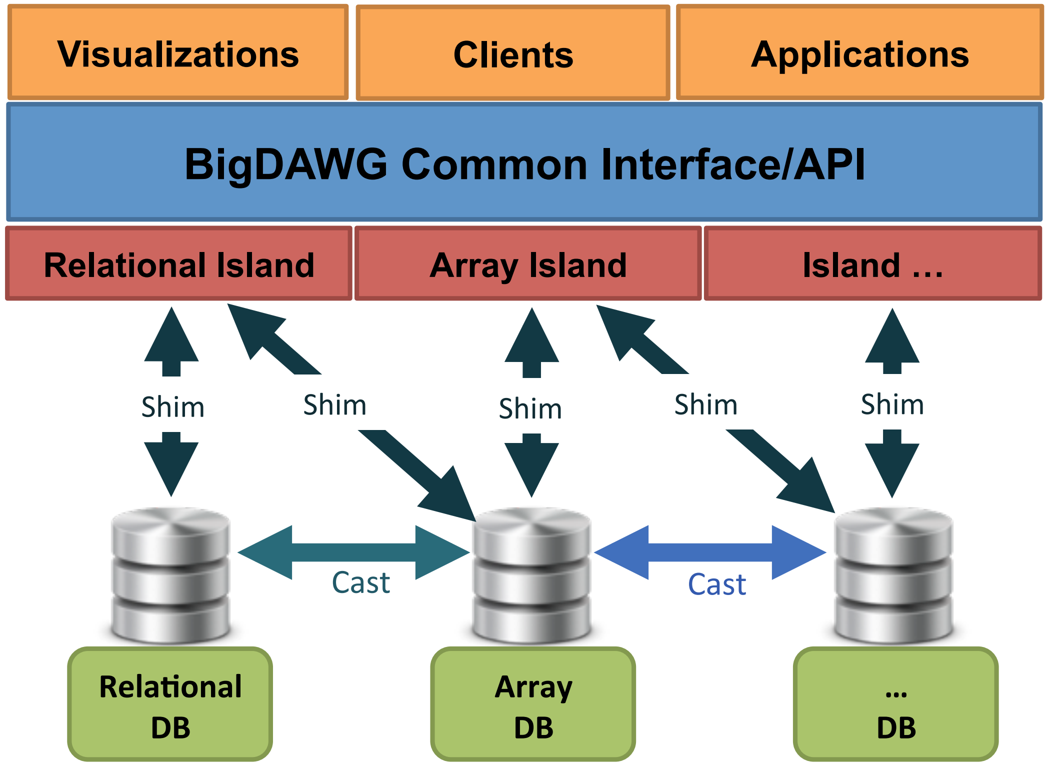
BigDAWG Polystore Reference Architecture



Goal: A single interface for ALL data

BigDAWG Overview

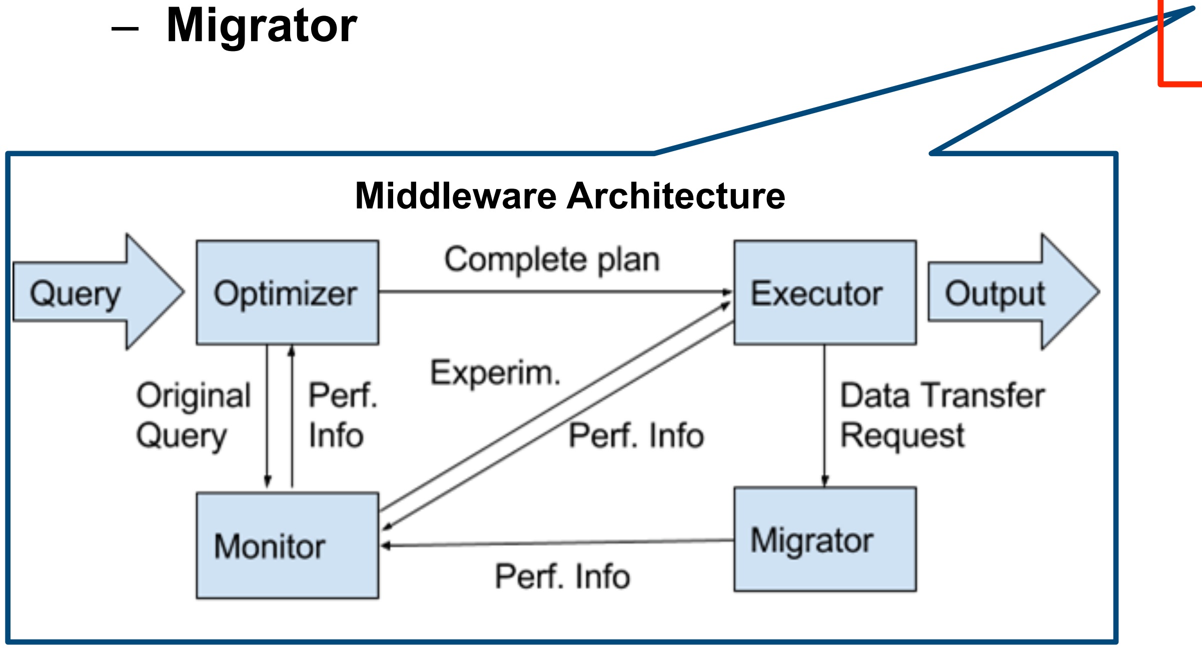
- ***Focus on solutions rather than technology***
- **Goal**
  - **Polystore reference architecture**
  - **Support:**
    - **High performance ingest and analytics**
    - **Points along the location transparency vs. semantic completeness spectrum**
- **Foundational Operations**
  - **BigDAWG Common interface (*interface*)**
  - **Common analytic translators (*shim*)**
  - **Common data translators (*cast*)**
  - **Multiple database support via *islands***

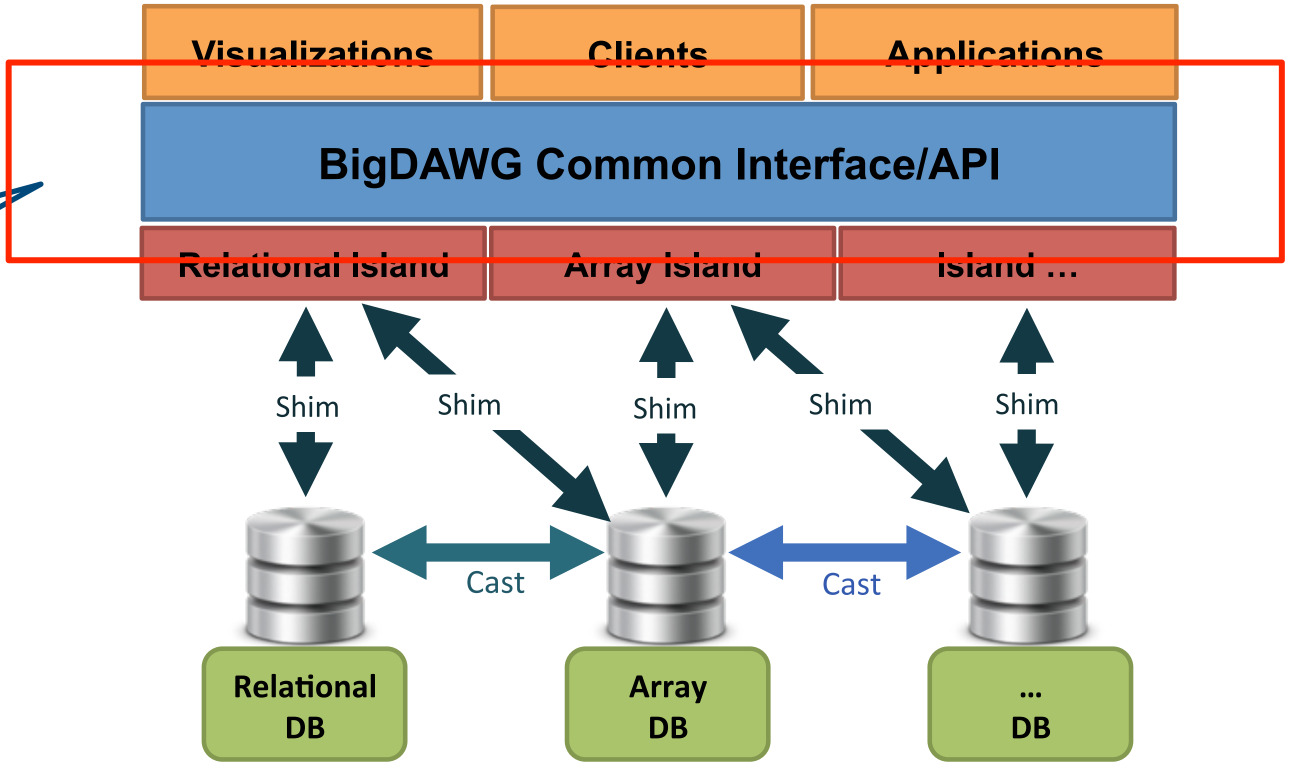


BigDAWG Architecture: VLDB’15, HPEC’16b

BigDAWG Middleware

- **BigDAWG Middleware**
  - **Optimizer**
  - **Monitor**
  - **Executor**
  - **Migrator**





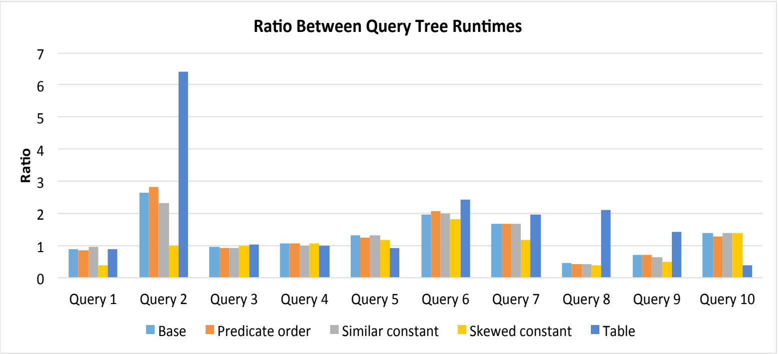
Middleware Details: HPEC’16c,HPEC’16d

BigDAWG Middleware

Early stages of optimizer

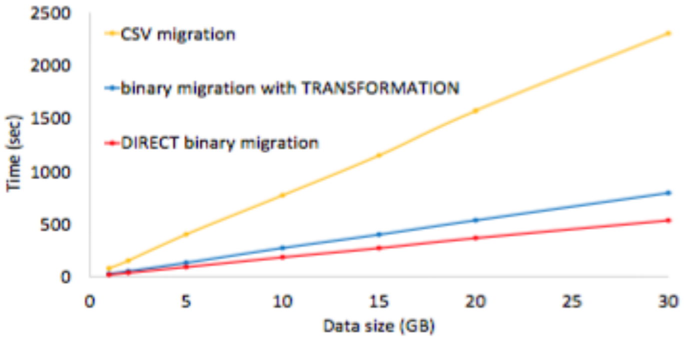
Monitor

- Responsible for determining the best execution strategy for a given query that is received
- Determines similarity with previous queries to determine best path
- Output query plan is sent to executor and migrator



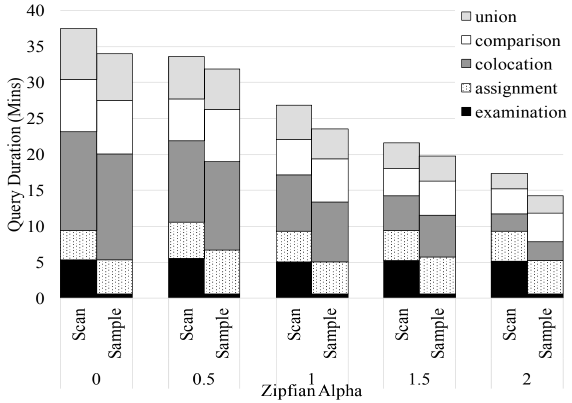
Migrator

- Responsible for moving data between engines or nodes as needed
- May be explicit (user defined) or implicit (based on query plan)
- Interacts with monitor and executor modules



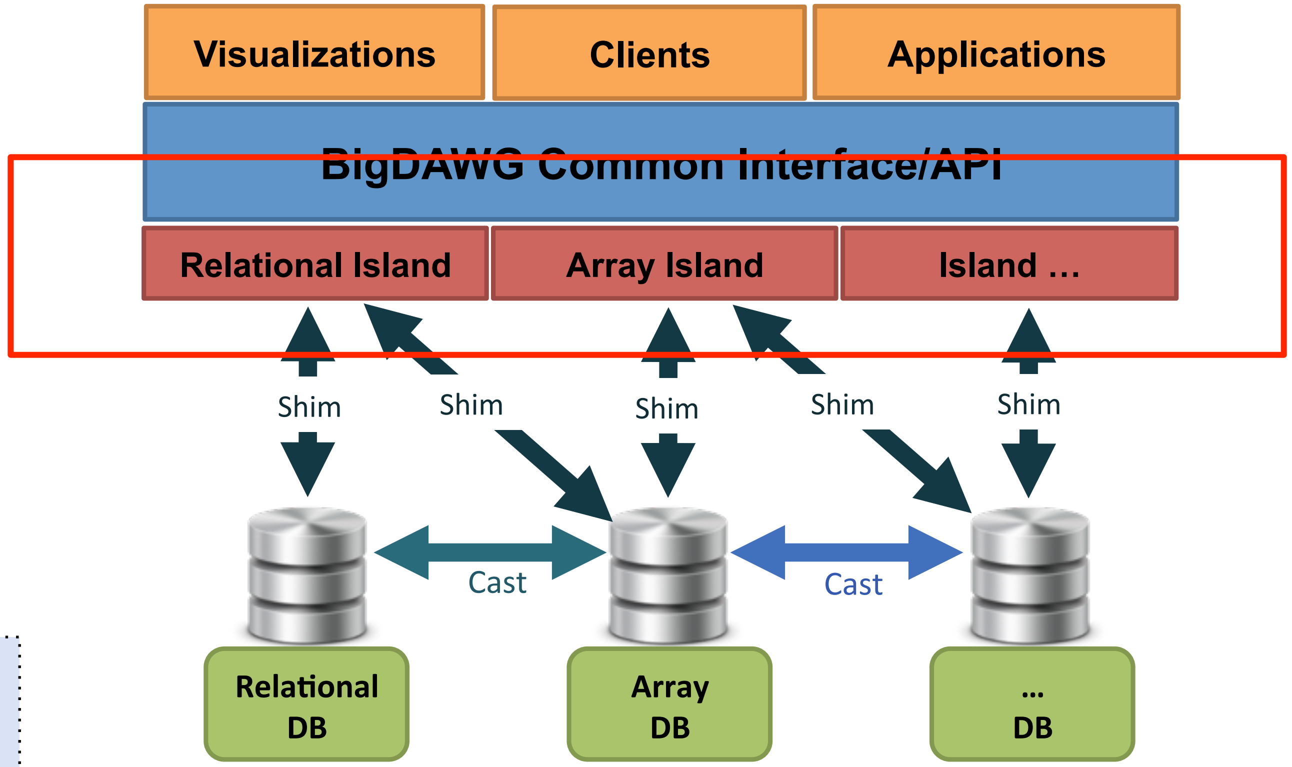
Executor

- Responsible for physical execution of query plan and recording results that are shared with monitor module
- Makes use of migrator as needed to complete execution



Semantic Islands as the Tradeoff

- **Islands are the trade-off between functionality and location transparency**
- **Islands have**
  - **A Data Model**
  - **A Language or Set of Operators**
  - **A Set of Candidate Database Engines**



User specifies the Island

```
RELATIONAL(select avg(temp) from device)
ARRAY(multiply(A,B))
```

* Islands do **Intersection** of engines
* BigDAWG does **Union** of Islands
* Islands are logical
* Also, degenerate islands for functionality

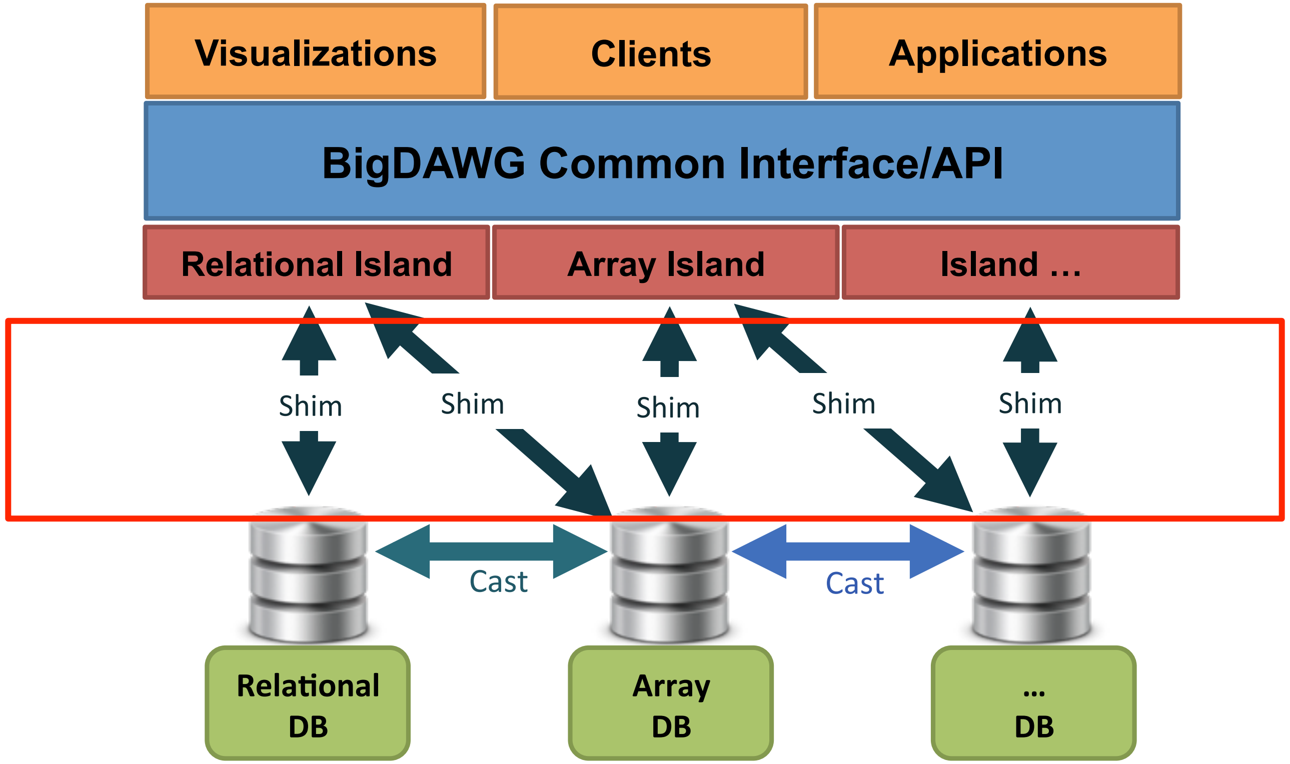
Details: HPEC'15

Key BigDAWG Operations
-Cast and Shim Operations-

- **Shim: Translates queries to native database language from island speak**

Shim Operation Example
(find number of non-zero entries)

```
nnz(T1) → SELECT COUNT(*) FROM T1
nnz(T2) → aggregate(T2, count(val));
nnz(T3) → scan –t T3 --np
```



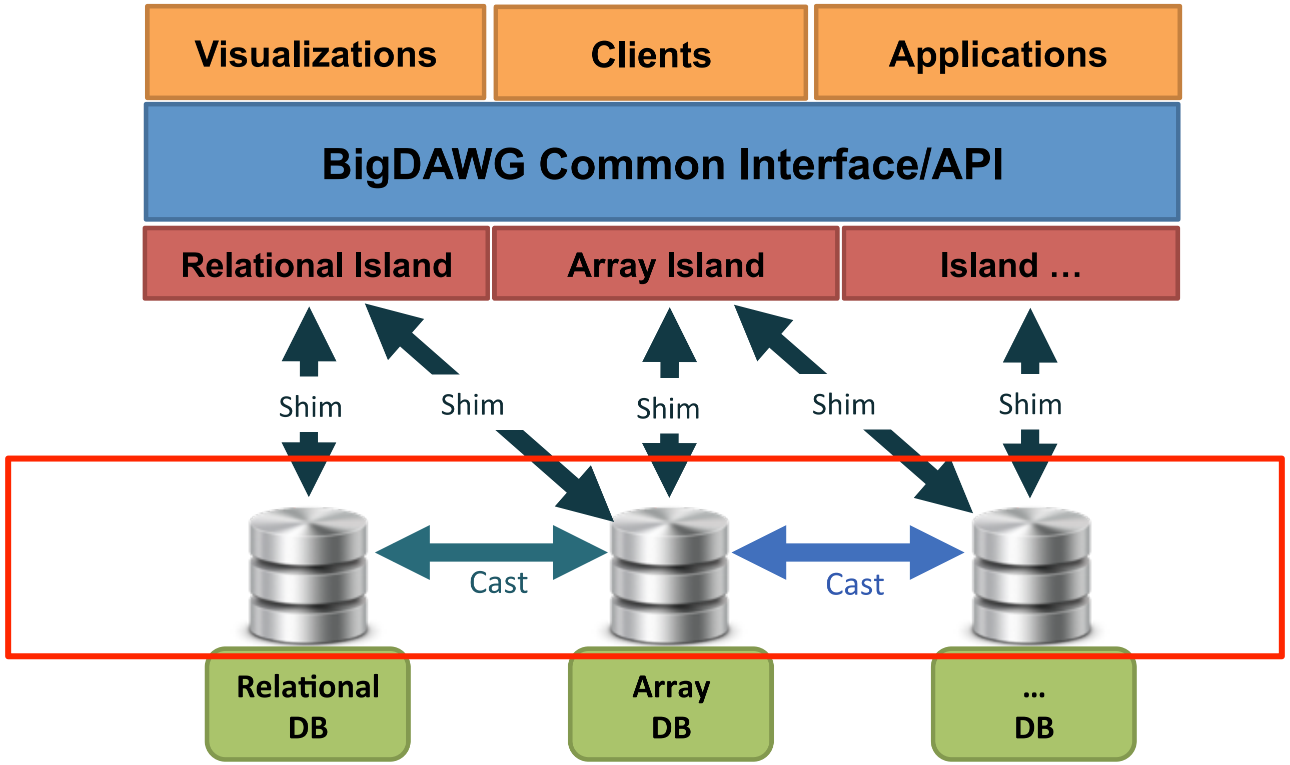
Details: HPEC'15

Key BigDAWG Operations
-Cast and Shim Operations-

- **Cast: Translates data between database engines (or islands)**

Cast Operation Example
(move data from one DB to another)

```
CAST nnz(T1)  →  ArrayDB
CAST nnz(T3)  →  ArrayDB
```



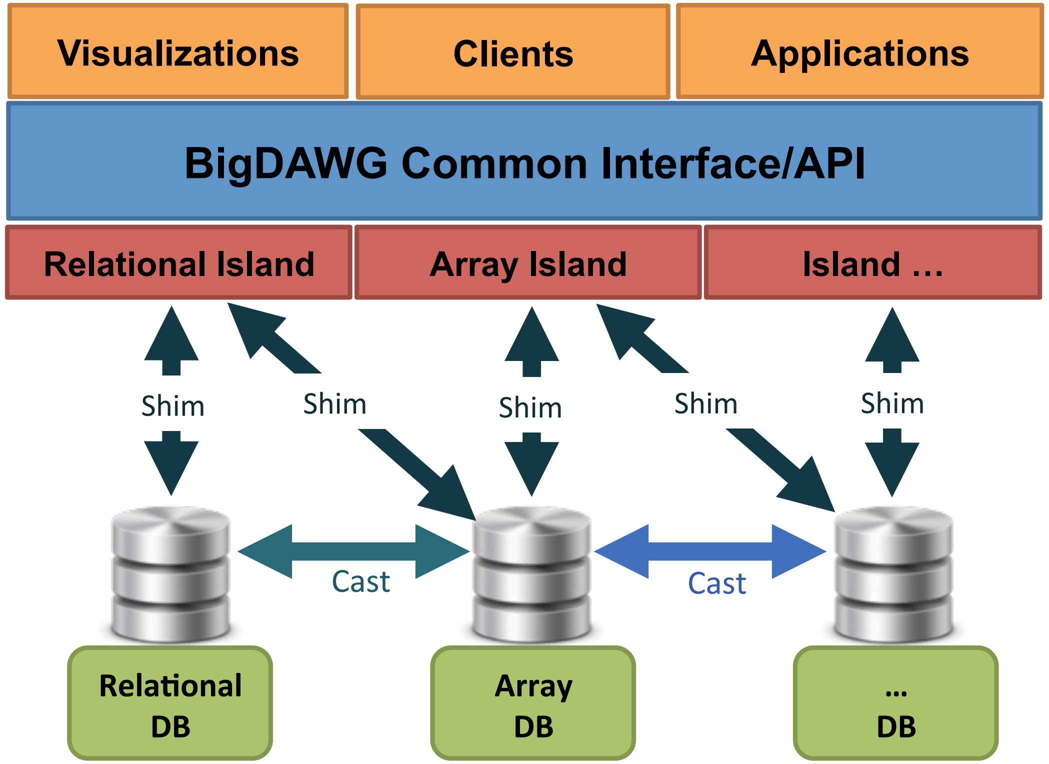
Details: HPEC'15

Outline

- **Introduction and Background**
- **BigDAWG**
  - **What it is**
  - **BigDAWG Initial Results**
  - **BigDAWG in Action: Ocean Genomics**
  - **S-Store Streaming Engine**
- **Summary and Future Work**

BigDAWG Prototype Implementation

- Support for 5 DB engines
- Island support
  - Support for islands based on arrays, relational, iteration and streaming
  - Support for degenerate islands (full semantic completeness at the cost of location transparency
- Cast operations based on associative arrays
- Developed a number of "apps" that can only be done via Polystore for medical and ocean genomics

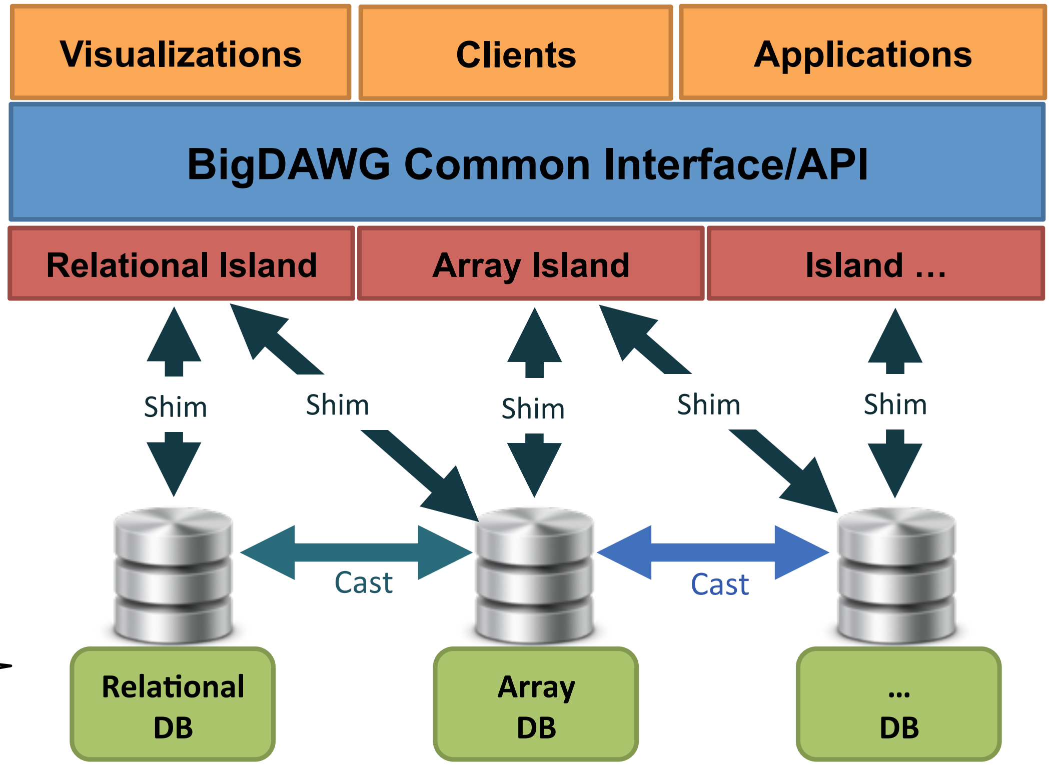


BigDAWG Prototype Implementation

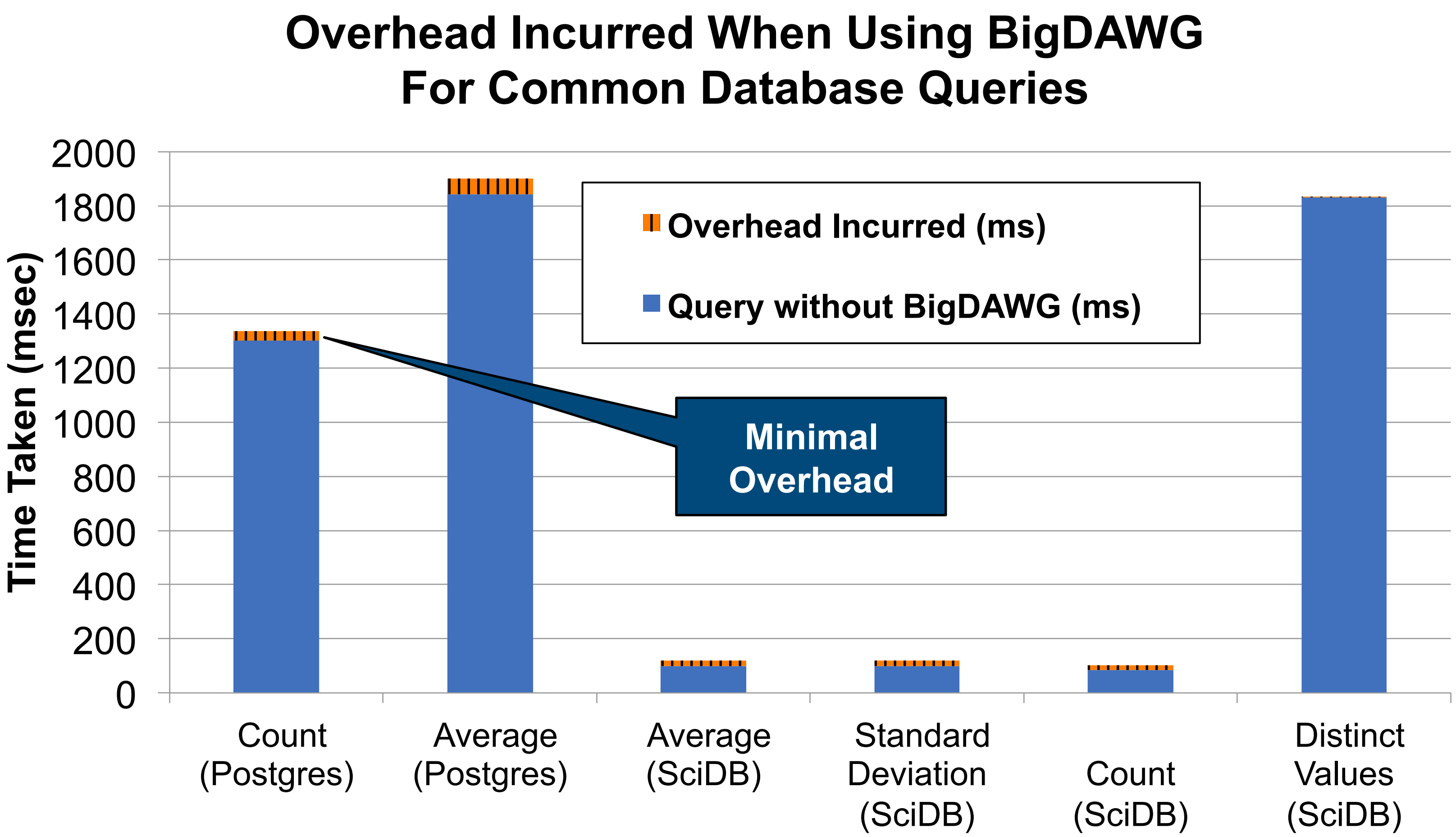
Applications
- Data Exploration
- Data Visualization
- Deep Analytics
- Streaming Analytics
- Extract-Transform-Load

Islands
- D4M (Associative Arrays)
- Myria (Iteration)
- Streams
- *Degenerate* Islands

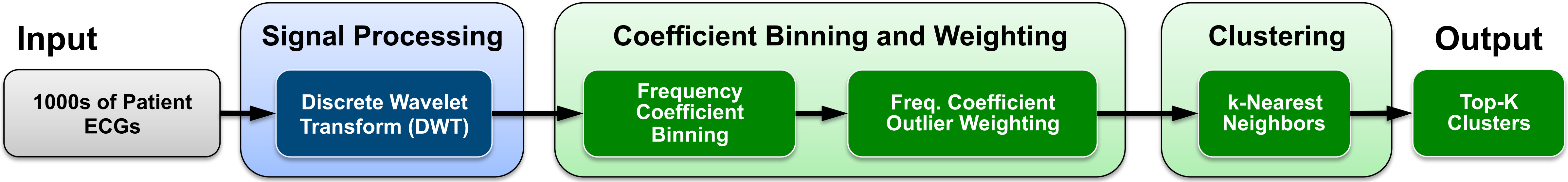
DB Engines
SciDB
accumulo
Myria



Prototype BigDAWG Overhead



BigDAWG Polystore Analytic Example

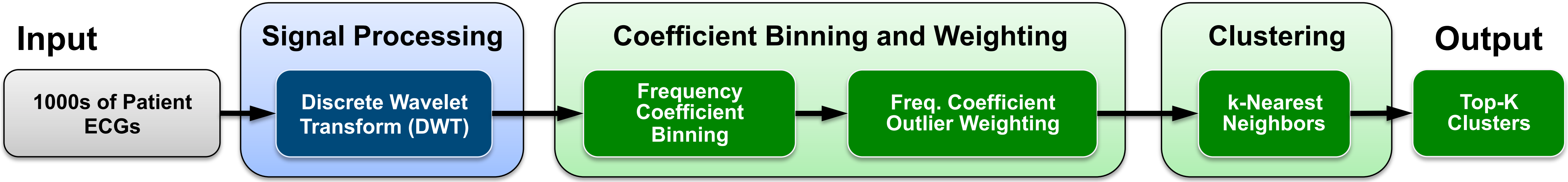


- **Goal: Find patients with similar ECG time-series***
- **Procedure**
  - **Perform Discrete Wavelet Transform of ECG**
  - **Generate wavelet coefficient histogram**
  - **TF-IDF waveform coefficients (weight rare changes higher)**
  - **Correlate against all other ECGs**

* *A novel method for the efficient retrieval of similar multiparameter physiologic time series using wavelet-based symbolic representations*, Saeed & Mark, AMIA 2006

TF-IDF=Term Frequency-Inverse Document Frequency

BigDAWG Polystore Analytic Example



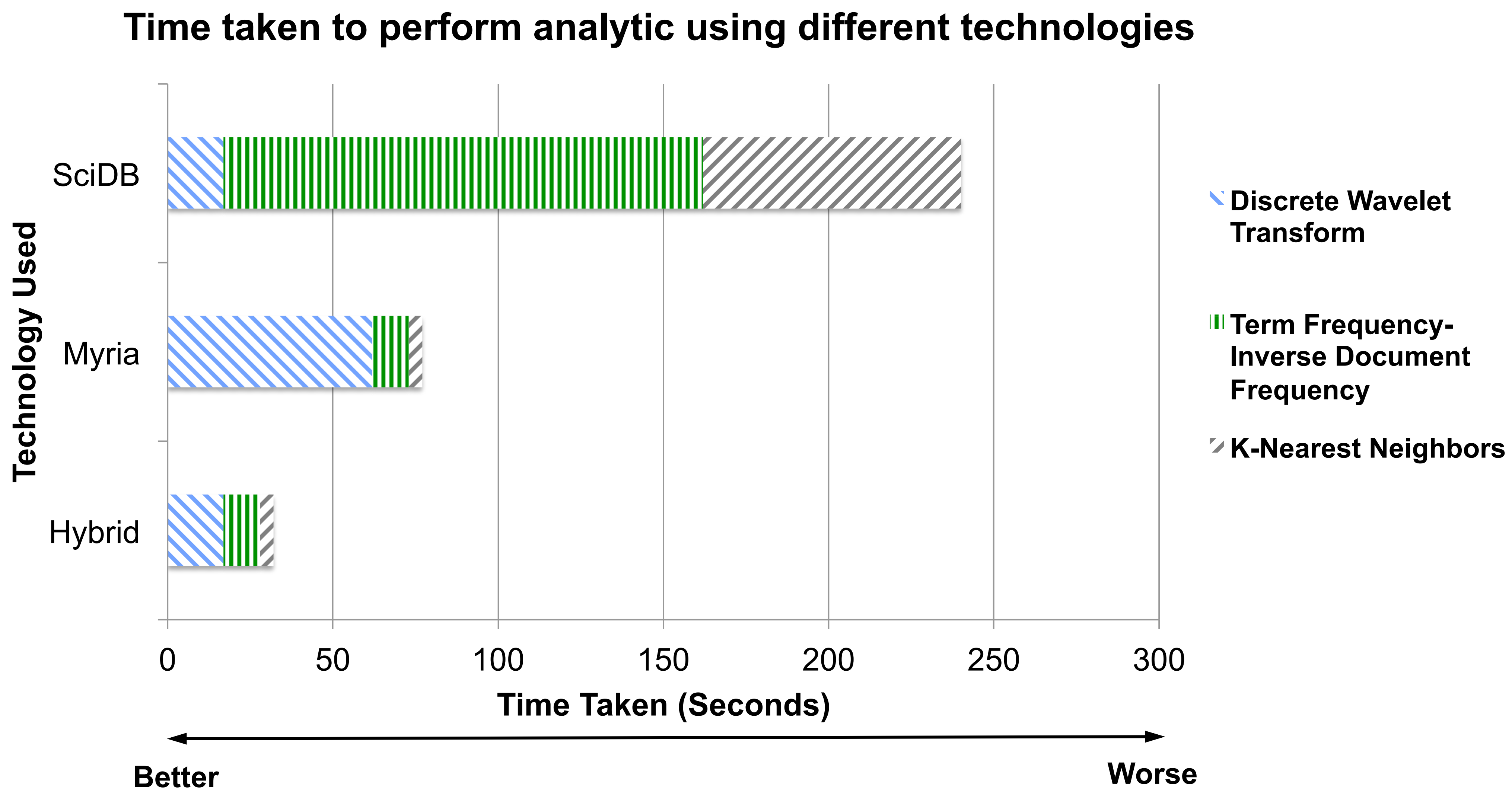
- Goal: Find patients with similar ECG time-series*
- Procedure
  - Perform Discrete Wavelet Transform of ECG
  - Generate wavelet coefficient histogram
  - TF-IDF waveform coefficients (weight rare changes higher)
  - Correlate against all other ECGs

- Show timings for individual pieces in two different types of databases
  - Option 1: Everything in a single system
  - Option 2: Polystore application

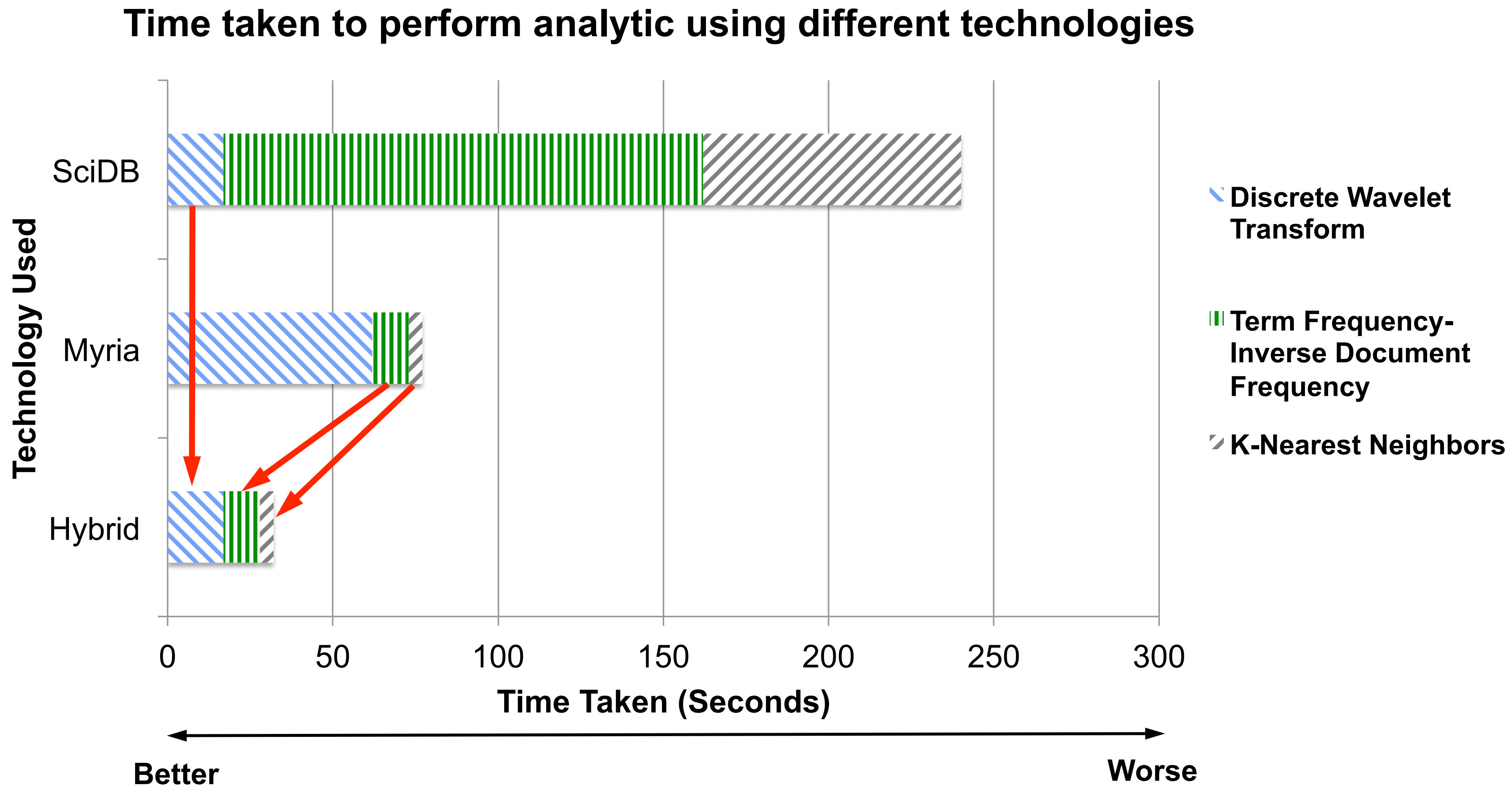
* *A novel method for the efficient retrieval of similar multiparameter physiologic time series using wavelet-based symbolic representations*, Saeed & Mark, AMIA 2006

TF-IDF=Term Frequency-Inverse Document Frequency

Polystore Analytic Performance



Polystore Analytic Performance



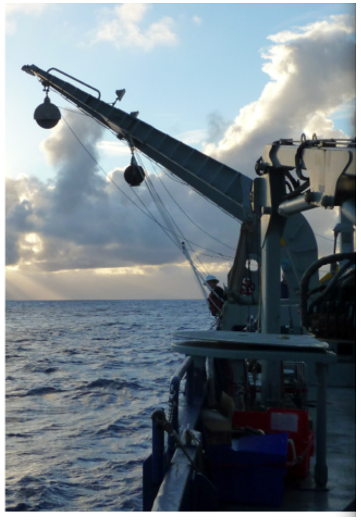
Outline

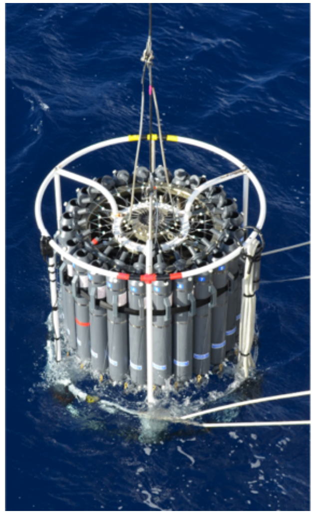
- **Introduction and Background**
- **BigDAWG**
  - **What it is**
  - **BigDAWG Initial Results**
  - **BigDAWG in Action: Ocean Genomics**
  - **S-Store Streaming Engine**
- **Summary and Future Work**

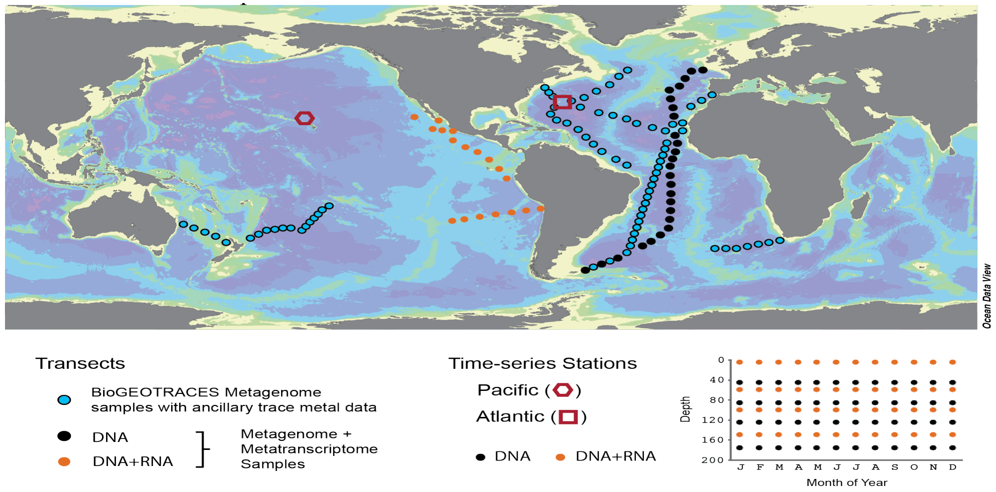
BigDAWG = Big Data Working Group

Ocean Genomic Data

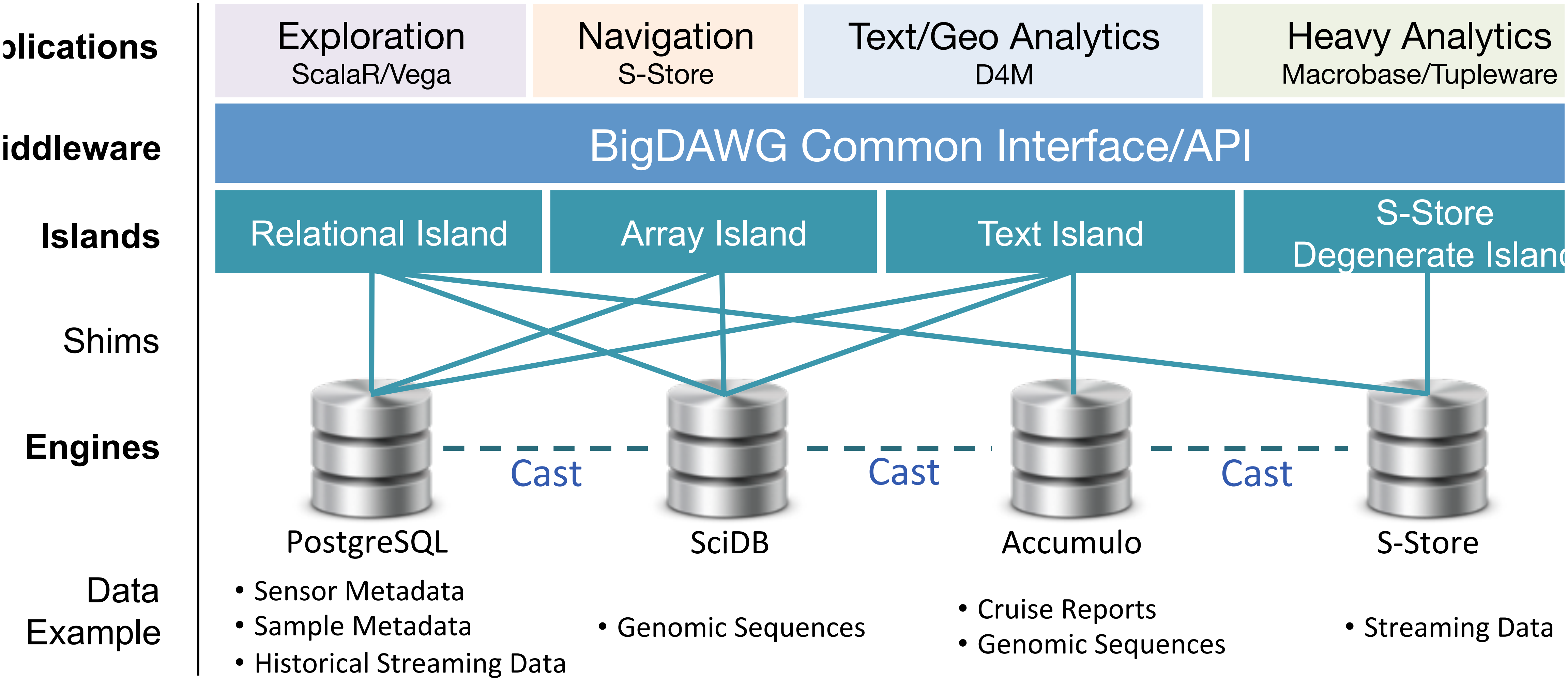
- **The Chisholm Lab (MIT) has been collecting seawater samples from the across the globe for many years**
  - **Currently a number of challenges that are faced by researchers.**
- **Contents:**
  - **Genome Sequence Data**
    - **For every individual sample, we quality controlled, trimmed and (sometimes) paired sequence data. Each sample contains many different DNA sequence reads from a particular sample corresponding to different DNA samples.**
  - **Discrete sample metadata**
    - **Recording of nearly 500 different entities for water samples (ocean chemistry)**
  - **Sensor Metadata**
    - **Information about recordings, where they took place**
  - **Cruise Reports**
    - **Free form text reports written as cruise logs**
  - **Streaming Data**
    - **Data collected from SeaFlow* system.**







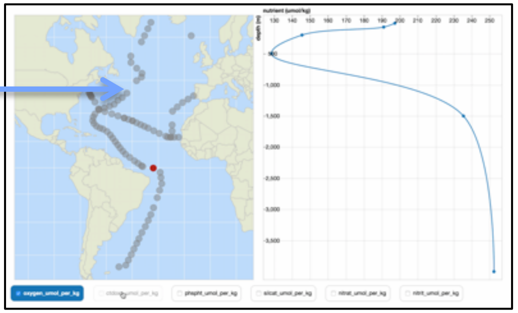
BigDAWG Ocean Genomics Architecture



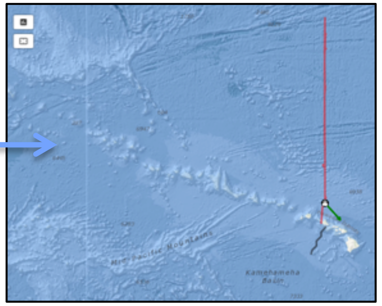
Ocean Metagenomic Analysis: CIDR’17 (to appear)

Ocean Genomics Polystore Applications

Exploration
(see the entire dataset)

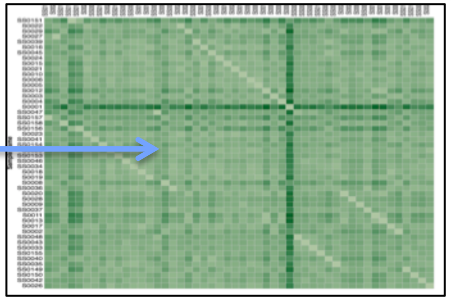


Navigation
(make cruises more efficient)

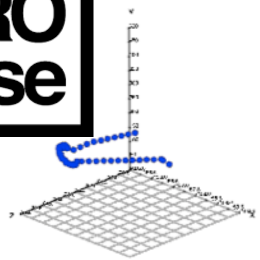


Geo-Analytics
(leverage the unstructured data)

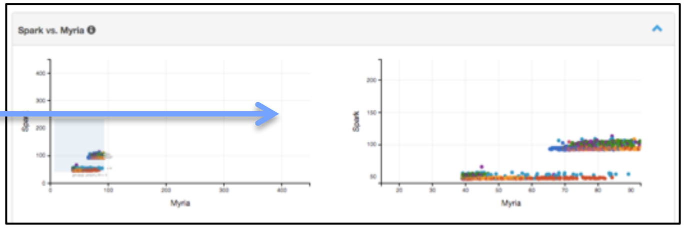
Genomic Processing
(look for interesting trends in genomic data)



Heavy Analytics
(cut across data set for deep analytics)



Performance Modeling
(see how well the system performs)



Ocean Metagenomic Analysis: CIDR’17 (to appear)

Outline

- **Introduction and Background**
- **BigDAWG**
  - **What it is**
  - **BigDAWG Initial Results**
  - **BigDAWG in Action: Ocean Genomics**
  - **S-Store Streaming Engine**
- **Summary and Future Work**

BigDAWG = Big Data Working Group

Streaming Databases

- Allow you to query data from a stream (rather than batching results and loading them into a traditional DB for querying)
  - Examples: Stock prices, kinematic data from autonomous vehicles, network data,…

Thanks to Nesime Tatbul (Intel/MIT), Stan Zdonik and John Meehan (Brown) for lending me their slides

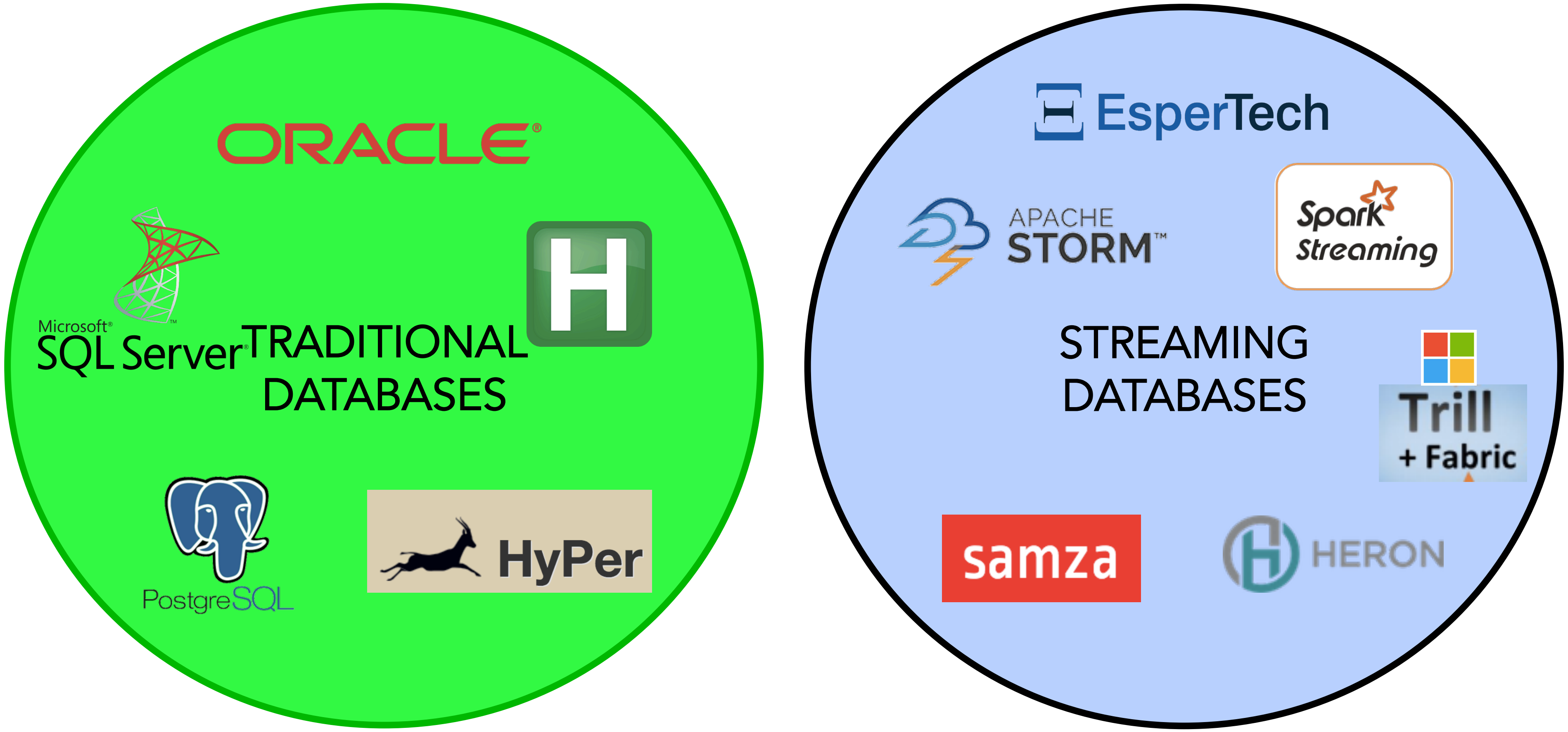
Streaming Databases

- **Allow you to query data from a stream (rather than batching results and loading them into a traditional DB for querying)**
  - **Examples: Stock prices, kinematic data from autonomous vehicles, network data,…**
- **Requires rethinking traditional systems:**

Traditional systems	**Streaming Systems**
State Management	Data Driven Processing
Pull operations	Push operations
Full queries	Partial Queries (transaction may not complete)
Multiple Passes	Single Pass
Higher latency (larger batch sizes)	Low Latency (small batch sizes)

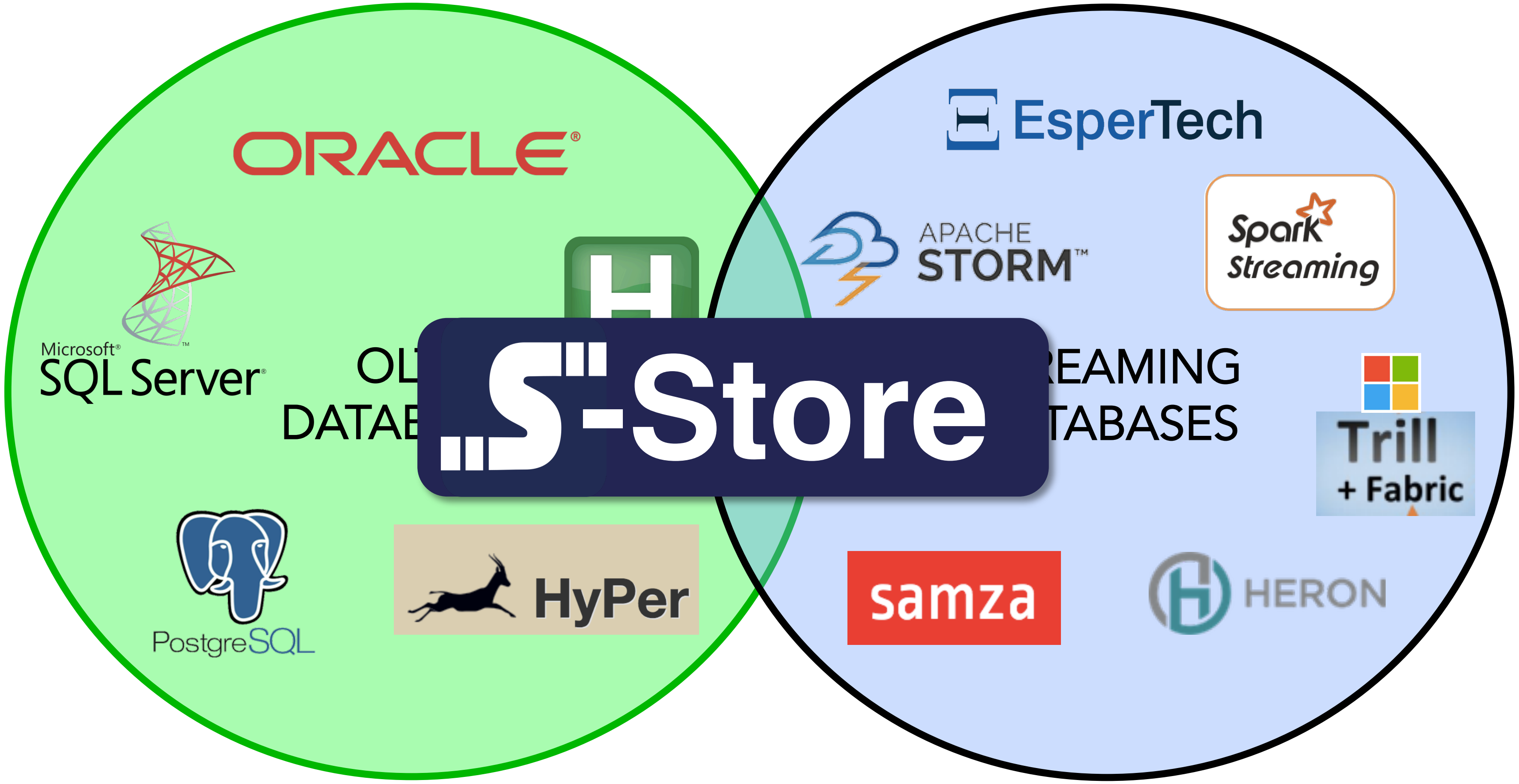
Goal: Develop next generation streaming engines to support stream transaction processing

Traditional vs. Streaming Databases



J. Meehan et al. S-Store: Streaming Meets Transaction Processing. PVLDB, 8(13), 2015.

S-Store: OLTP + Streaming Databases



J. Meehan et al. S-Store: Streaming Meets Transaction Processing. PVLDB, 8(13), 2015.

S-Store Features

ARCHITECTURE
extends H-Store NewSQL system

TRANSACTION MODEL
extends ACID to include streaming

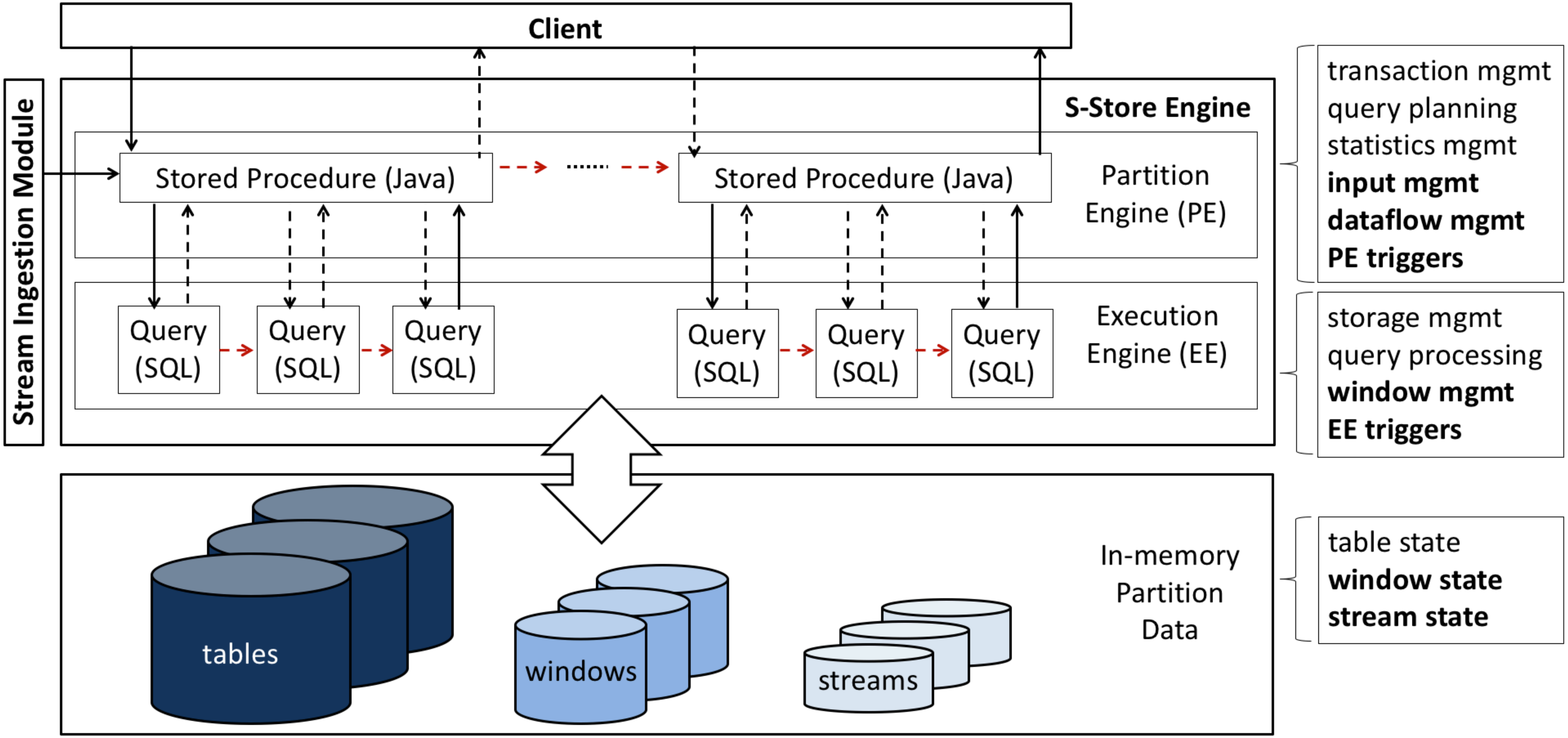
PERFORMANCE COMPARISON
to state-of-the-art streaming systems*

APPLICATIONS
support wide spectrum of workloads

***for workloads that include shared mutable state + streaming**

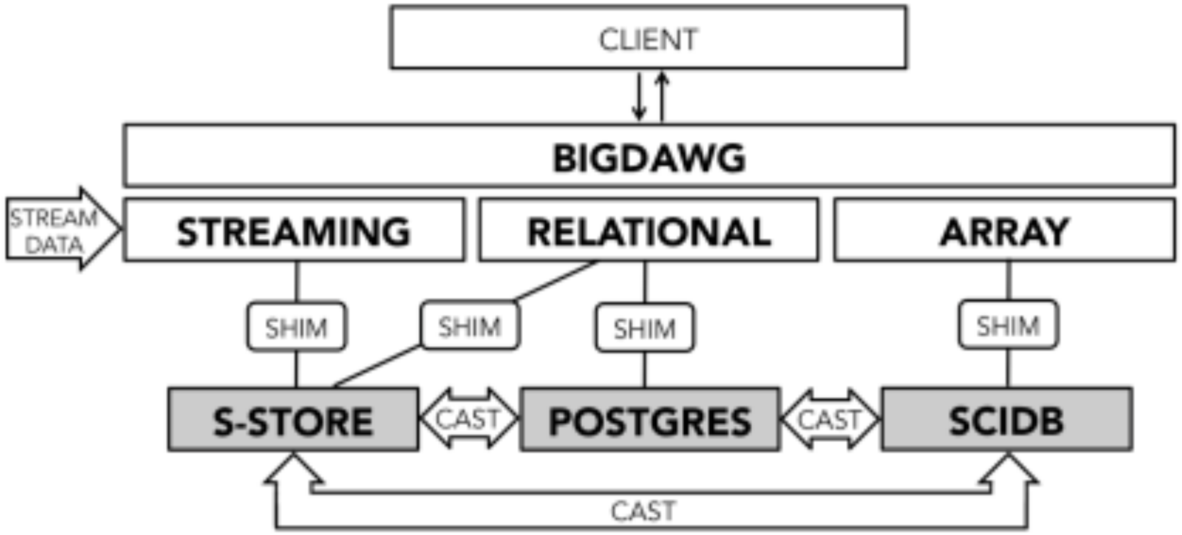
J. Meehan et al. S-Store: Streaming Meets Transaction Processing. PVLDB, 8(13), 2015.

S-Store Architecture



N. Tatbul et al. Handling Shared, Mutable State in Stream Processing with Correctness Guarantees. IEEE Data Engineering Bulletin 2015.

S-Store and BigDAWG

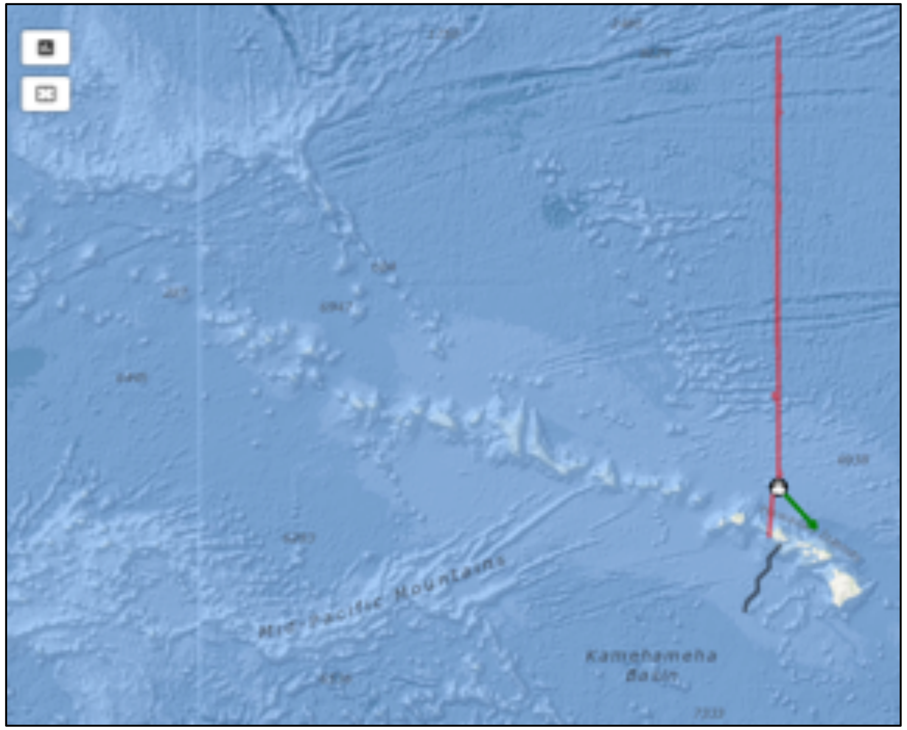


- S-Store is the only engine that (currently) lives under two different islands.
- Streaming island
  - captures common basic primitives of streaming in general
  - supports streaming and ETL operations
- Applications:
  - Applied to medical and oceanographic data

J. Meehan et al. Integrating Real-Time and Batch Processing in a Polystore. IEEE HPEC 2016 | BigDAWG Architecture: CIDR'17

Demonstration:
S-Store + BigDAWG for Ocean Genomics

Goal: To provide real-time navigation support for ships collecting water samples



S-Store Summary

- S-Store bridges the gap between traditional OLTP databases and streaming databases to provide high performance query processing with strong transactional guarantees

- For more information:
  - Nesime Tatbul: tatbul@csail.mit.edu
  - John Meehan: john@cs.brown.edu
  - Stan Zdonik: sbz@cs.brown.edu

Outline

- **Introduction and Background**
- **BigDAWG**
  - **What it is**
  - **BigDAWG Initial Results**
  - **BigDAWG in Action: Ocean Genomics**
  - **S-Store Streaming Engine**
- **Summary and Future Work**

BigDAWG = Big Data Working Group

Inaugural Workshop on Polystore Databases

Important Dates

~~October 10, 2016:~~
~~Full workshop papers submission deadline~~

November 1, 2016:
Notification of paper acceptance to authors

November 15, 2016:
Camera-ready of accepted papers

December 5-8, 2016:
Workshops Dates

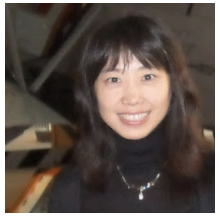
Website:
https://goo.gl/oLFR1F

Contact:
Vijay Gadepally
(vijayg@mit.edu)

Research topics included in workshop:

- New Computational Models for Big Data
- Languages/Models for integrating disparate data such as graphs, arrays, relations
- Query evaluation and optimization in federated or polystore systems
- High Performance/Parallel Computing Platforms for Big Data
- Integration of HPC and Big Data platforms
- Data Acquisition, Integration, Cleaning, and Best Practices
- Complex Big Data Applications in Science, Engineering, Medicine, Healthcare, Finance, Business, Transportation, Retailing, Telecommunication, Government and Defense applications
- Efficient data movement and scheduling, failures and recovery for analytics

Keynotes


Luna Dong


Fatma Ozcan

Summary

- Polystore architecture shows great promise to make impossible problem a bit easier
- We’ve applied the BigDAWG reference implementation to a number of data sets
- Leverages Big Data and HPC resources
- Promising performance, but lots do to!

- Many areas for future work!
  – Query optimization
  – Smarter query planning
  – More DBMSs
  – Better islands
  – …
  – ...
  – ...

Acknowledgements

ISTC Team

- Michael Stonebraker
- Brandon Haynes
- Sam Madden
- Peinan Chen
- Magdalena Balazinska
- Tim Mattson
- Adam Dziedzic
- Aaron Elmore
- Jennie Duggan
- Nesime Tatbul
- John Meehan
- Stand Zdonik

MIT Lincoln Laboratory

- Jeremy Kepner
- Albert Reuther
- Lauren Milechin
- Dylan Hutchison
- Braden Hancock
- David Martinez
- Roger Khazan
- Siddharth Samsi

LLSC Team

- Bill Arcand
- Bill Bergeron
- David Bestor
- Chansup Byun
- Matt Hubbell
- Mike Jones
- Pete Michaleas
- Julie Mullen
- Andy Prout
- Tony Rosa
- Charles Yee

Recent BigDAWG Publications

[CIDR’17] “Demonstrating the BigDAWG Polystore System for Ocean Metegenomic Analysis,” Tim Mattson, Vijay Gadepally, Zuohao She, Adam Dziedzic, Jeff Parkhurst, Conference on Innovations on Data Research (CIDR), 2017 (to appear).

[HPEC’16a] "Associative Array Model of SQL, NoSQL, and NewSQL Databases", Jeremy Kepner, Vijay Gadepally (MIT), Dylan Hutchison (University of Washington), Hayden Jananthan (MIT), Timothy Mattson (Intel), Siddharth Samsi, Albert Reuther (MIT), IEEE High Performance Extreme Computing (HPEC), 2016.

[HPEC’16b] "The BigDAWG Polystore System and Architecture", Vijay Gadepally, Peinan Chen (MIT), Jennie Duggan (Northwestern University), Aaron Elmore (University of Chicago), Brandon Haynes (University of Washington), Jeremy Kepner, Samuel Madden (MIT), Tim Mattson (Intel), Michael Stonebraker (MIT), IEEE High Performance Extreme Computing (HPEC), 2016.

[HPEC’16c] “The BigDawg Monitoring Framework", Peinan Chen, Vijay Gadepally, Michael Stonebraker, IEEE High Performance Extreme Computing (HPEC), 2016.

[HPEC’16d] "Cross-Engine Query Execution in Federated Database Systems", Ankush M. Gupta, Vijay Gadepally, Michael Stonebraker, IEEE High Performance Extreme Computing (HPEC), 2016.

[VLDB’15] "A Demonstration of the BigDawg Polystore System", Aaron Elmore, Jennie Duggan, Michael Stonebraker, Magda Balazinska, Ugur Cetintemel, Vijay Gadepally, Jeff Heer, Bill Howe, Jeremy Kepner, Tim Kraska, et al., Proceedings of the VLDB Endowment, 2015

[HPEC’15] "D4M: Bringing Associative Arrays to Database Engines", Vijay Gadepally, Jeremy Kepner, William Arcand, David Bestor, Bill Bergeron, Chansup Byun, Lauren Edwards, Matthew Hubbell, Peter Michaleas, Julie Mullen, Andrew Prout, Antonio Rosa, Charles Yee, Albert Reuther, IEEE High Performance Extreme Computing Conference (HPEC), 2015.

[IPDPS’15] "Graphulo: A Graph Library for NoSQL Databases", Vijay Gadepally et al., IEEE International Parallel and Distributed Processing Symposium (IPDPS) GABB, 2015.