Polystore, Julia, and productivity in a Big Data world

Tim Mattson, Intel labs
timothy.g.mattson@intel.com
Intel-PI for the Big Data “Intel Science and Technology Center”

People I stole content from for this talk: Stavros Papadopoulos, Jake Bolewski, Mike Stonebraker, Vijay Gadepally, Bill Howe, Magda Balazinska, Jennie Duggan, Aaron Elmore, Sam Madden, Jeremy Kepner, Kiran Pamnany, and Tatiana Shpeisman.
Legal Disclaimer & Optimization Notice

• INFORMATION IN THIS DOCUMENT IS PROVIDED “AS IS”. NO LICENSE, EXPRESS OR IMPLIED, BY ESTOPPEL OR OTHERWISE, TO ANY INTELLECTUAL PROPERTY RIGHTS IS GRANTED BY THIS DOCUMENT. INTEL ASSUMES NO LIABILITY WHATSOEVER AND INTEL DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY, RELATING TO THIS INFORMATION INCLUDING LIABILITY OR WARRANTIES RELATING TO FITNESS FOR A PARTICULAR PURPOSE, MERCHANTABILITY, OR INFRINGEMENT OF ANY PATENT, COPYRIGHT OR OTHER INTELLECTUAL PROPERTY RIGHT.

• Software and workloads used in performance tests may have been optimized for performance only on Intel microprocessors. Performance tests, such as SYSmark and MobileMark, are measured using specific computer systems, components, software, operations and functions. Any change to any of those factors may cause the results to vary. You should consult other information and performance tests to assist you in fully evaluating your contemplated purchases, including the performance of that product when combined with other products.

• Copyright © , Intel Corporation. All rights reserved. Intel, the Intel logo, Xeon, Xeon Phi, Core, VTune, and Cilk are trademarks of Intel Corporation in the U.S. and other countries.

Optimization Notice

Intel’s compilers may or may not optimize to the same degree for non-Intel microprocessors for optimizations that are not unique to Intel microprocessors. These optimizations include SSE2, SSE3, and SSSE3 instruction sets and other optimizations. Intel does not guarantee the availability, functionality, or effectiveness of any optimization on microprocessors not manufactured by Intel. Microprocessor-dependent optimizations in this product are intended for use with Intel microprocessors. Certain optimizations not specific to Intel microarchitecture are reserved for Intel microprocessors. Please refer to the applicable product User and Reference Guides for more information regarding the specific instruction sets covered by this notice.

Notice revision #20110804
Disclaimer

• The views expressed in this talk are those of the speaker and not his employer.

• If I say something “smart” or worthwhile:
  – Credit goes to the many smart people I work with.

• If I say something stupid...
  – It’s my own fault

I work in Intel’s research labs. I don’t build products. Instead, I get to poke into dark corners and think silly thoughts... just to make sure we don’t miss any great ideas.

Hence, my views are by design far “off the roadmap”.
Big Data Today
common assumption ... Big Data = Hadoop/Spark

Intel Analytics Toolkit

Unified UI’s across the workflow

Easier feature & model creation

Cloud & On-Prem

Python Libraries  Future Libraries  Query Interfaces  Viz Tools  3rd Party GUIs/SDKs  BI Connectors

“DATA SCIENCE” API

DATA WRANGLING

Useful String Manipulation  Useful Math Operators  Graph Construction Tools

MACHINE LEARNING AND STATISTICS

Graphical Algorithms  Classical Algorithms

APACHE HADOOP  APACHE SPARK

FILESYSTEMS AND NOSQL STORAGE

HW PLATFORM

Third party names are the property of their owners
What’s next for Big Data

- Hadoop/SPARK is great … it helped put Big Data on the map
- Comes from a time when we were just thrilled to “do” big data.

- Hadoop/Spark design did not emphasize:
  - Performance for complex analytics.
  - Efficient utilization of the hardware
  - Programmability for anything beyond “embarrassingly parallel” applications.

Challenge: What happens when Hadoop/Spark runs out of steam? What comes next?
Big Data in the real world

• Consider patient data in an Intensive Care Unit (e.g. MIMIC II data set*)

- EKG traces
- Blood oxygen
- Blood pressure
- EEG traces

- Demographic Caregiver notes
- Medical charts
- Lab test results
- Xray, MRI, etc.

The challenge … apply predictive analytics across all data … so we can show up to restart a heart before it stops beating!!!

Big Data in the real world
Messy, heterogeneous, complex, streaming ...

- Consider patient data in an Intensive Care Unit (e.g. MIMIC II data set*)


# MIMC doesn’t include images. We are talking to several groups to add an image database to our project
Analysis of published MIMICII papers

- Data in databases is used; data in files is not
  - Data in files is nearly equivalent to deleting the data

A disruptive idea: Match data to the data-store technology but present as a single Data Base Management system to the end-users … A disruptive idea we call Polystore.

Source: Vijay Gadepally of MIT Lincoln labs

*Based on PhysioNet MIMIC2 ICU data
BigDawg: An integrated **polystore** system

### Applications
- e.g., Medical data, astronomy, twitter, urban sensing, IoT

### Visualization & presentation
- e.g., ScalaR, imMens, SeeDB, Prefetching

### SW Development
- e.g., APIs for traditional languages, Julia, GraphMat, ML Base

---

**BigDAWG Query Language and Data Federation layer**

**Stream**

- Real Time DBMSs
  - S-Store
- Analytics DBMSs
  - SciDB
  - MyriaX
  - TupleWare
  - TileDB

---

**Analytics**
- e.g., PLASMA, ML algos, plsh, GraphBLAS, other analytics packages

---

**Hardware platforms**
- e.g., Cloud and cluster infrastructure, NVM simulator, 1000 core simulator, Xeon Phi, Xeon

---

Third party names are the property of their owners
BigDawg: An integrated *polystore* system

Let’s focus on this layer ... the heart of BigDAWG

BigDAWG Query Language and Data Federation layer

“Narrow Waist” Provides Portability

Third party names are the property of their owners
BigDAWG Data Federation

• Two Key Components:
  – BigDAWG Query Language or BQL:
    – the quest for “one query language to rule them all”
  – BigDAWG Data Federation API:
    – **Islands**: a collection of data stores that share a data model and query language
    – **Shims**: to translate queries between islands
    – **Casts**: to move data from one island to another

Based on ISTC research over the last 3 years, we think we know how to do this.

High risk transformative research ... many people think this is impossible.

RDBMS = relational DBMS
Our VLDB’2015 Demo

A Demonstration of the BigDAWG Multi-Database System

ABSTRACT
This paper presents BigDAWG, a reference implementation of a new architecture for “Big Data” applications. Such applications

BigDAWG stores MIMIC II in a mixture of backends, including Postgres, which stores the patient metadata, SciDB [4], which stores the historical waveform data in a time-series (array) database, S-Store [1] which stores a stream of device information, and Apache

1
Our VLDB’2015 Demo

A Demonstration of the BigDAWG Multi-Database System

ABSTRACT

This paper presents BigDAWG, a reference implementation of a new architecture for “Big Data” applications. Such applications not only call for large-scale analytics, but also for real-time stream analytics working on patient data. Hence, this application serves the needs of doctors and researchers and provides real-time support for streams of patient data. To demonstrate the challenges inherent in applications bringing together the needs of many users and data sources, this use case is based on the real intensive-care unit (ICU) dataset, MIMIC II [12], which contains waveform data (up to 125 Hz measurements from bedside ICU admissions at Boston’s Beth Israel Deaconess Hospital. It contains over 26,000 patient admissions and about 250 million data points. We have preliminary results targeting MyriaX, SciDB, S-store, D4M, graphulo and several visualization packages.

MyriaX

SciDB

Myria

SciDB

MyriaX

Array DBMS

RDBMS X

RDBMS Y

Streams

Clients

Visualizations

Streams

BigDAWG Query Language and Data Federation layer

Array Island

Relational

Island X

Shim

CAST

Shim

CAST

Shim

CAST

Shim

Shim

Hypotension predictor

Third party names are the property of their owners
The App: Hypotension Predictor

- Problem: blood pressure drops (hypotension) $\rightarrow$ shock $\rightarrow$ death. Early intervention is key for survival.
- Solution: Machine learning over heterogeneous data (from MIMIC II) to identify patients about to suffer from a severe drop in blood pressure?
- Algorithm (from Saeed and Mark*) build a classifier .. Haar transforms over MIMICII time series data, summarize as histograms, and performs a K nearest neighbor search. Correlate with patient data.

Hypotension Classifier
Runtime in seconds (lower is better)

Source: Magdalena Balazinska and Brandon Haynes, university of Washington.

Third party names are the property of their owners
BigDawg: An integrated *polystore* system

**Applications**
e.g., Medical data, astronomy, twitter, urban sensing, IoT

**Visualization & presentation**
e.g., ScalaR, imMens, SeeDB, Prefetching

**SW Development**
e.g, APIs for traditional languages, Julia, GraphMat, ML Base

BigDAWG Query Language and Data Federation layer

---

**Real Time DBMSs**
- S-Store

**Analytics DBMSs**
- SciDB
- MyriaX
- TupleWare
- TileDB

**Analytics**
e.g., PLASMA, ML algos, plsh, GraphBLAS, other analytics packages

---

**Hardware platforms**
e.g., Cloud and cluster infrastructure, NVM simulator, 1000 core simulator, Xeon Phi, Xeon

“Narrow Waist” Provides Portability

Third party names are the property of their owners
System Infrastructure for BigDAWG Research

• To explore BigDAWG, we need to:
  – Quickly load a variety of DataBase Management Systems (DBMS)
  – Mix HPC (MPI) jobs with traditional DBMS jobs
  – Manage everything through an end-user driven web interface

• Fortunately, we were able to work with the team behind the MIT SuperCloud.

Common Big Data Architecture

Users
- Operators
- Analysts
- Commanders

Web
- Ingest
- Databases

Scheduler
- Computing

Data
- OSINT
- Weather
- HUMINT
- C2
- Ground
- Maritime
- Air
- Space
- Cyber

Source: Jeremy Kepner, MIT Lincoln Labs
Why a Database Management System?

• Remove requirement for dedicated servers, while avoiding virtual machines (VMs)
  – In addition to performance concerns, VMs historically have caused problems for the timeout-based failure detection features of Accumulo

• Enable rapid creation of new databases

• Reduce waste of resources on idle databases

• Create a viable backup & restore strategy

• Ensure security concerns are appropriately addressed

• Empower the less IT savvy researchers & scientists with self-service commands for common requests

• Integration with HPCC scheduler

Database creation should be closer to a mkdir than a major IT project

Source: Jeremy Kepner, MIT Lincoln Labs
Database Lifecycle

Central Storage (Lustre)

Scheduler

Web Portal

Dynamic DNS

Database User

db_start testdb01

Cluster Switch

LAN Switch

Web Portal

Dynamic DNS

System Admin

status: stopped

db_create accumulo -n 4 testdb01 SecGroup

Source: Jeremy Kepner, MIT Lincoln Labs
The SuperCloud lets researchers load multiple databases and Analytics jobs onto physical hardware through a simple web based interface.

Vital for productivity when integrating results into BigDAWG from so many teams!
## BigDawg: An integrated polystore system

### Applications
- e.g., Medical data, astronomy, twitter, urban sensing, IoT

### Visualization & presentation
- e.g., ScalaR, imMens, SeeDB, Prefetching

### SW Development
- e.g, APIs for traditional languages, Julia, GraphMat, ML Base

### BigDAWG Query Language and Data Federation layer

### Stream
- **Real Time DBMSs**
  - S-Store
  - SciDB
- **Analytics DBMSs**
  - MyriaX
  - TupleWare
  - TileDB

### Analytics
- e.g., PLASMA, ML algos, psh, GraphBLAS, other analytics packages

### Hardware platforms
- e.g., Cloud and cluster infrastructure, NVM simulator, 1000 core simulator, Xeon Phi, Xeon

---

**Third party names are the property of their owners**
Arrays in Big Data problems

• Data is often naturally considered as an array:
  – An object with multiple dimensions (e.g. 2)
  – The dimensions define a logical coordinate space
  – A cell “exists” at each point in the coordinate space.
  – A cell has one or more attributes which collectively define the “value” at that cell.

• Data is usually sparse
  – E.G. the AIS data set showing ship locations as a function of time in and around U.S. waters
TileDB: a new array data storage manager: optimized for Sparse Arrays

**Logical representation**

- **attribute values**: \((a_1, a_2, ..., a_m)\)
- **cell**: An occupied cell in the logical representation.
- **empty cell**: An unoccupied cell in the logical representation.
- **dimensions**: The x and y coordinates of the grid.

**Physical representation**

- **coordinates**: The x, y coordinates of the physical representation.
- **Files**: Physical tiles are stored as separate files.
- **segment**: An atomic unit of I/O.
- **tile**: An atomic unit of processing.
- **cell**: A data entry in the physical representation.

**Tile**: Atomic unit of processing

**Segment**: Atomic unit of I/O

Manage array storage as tiles of different shape/size in the index space, but with \(~\)equal number of non-empty cells.
Joint Genotyping Benchmark*

*Benchmark jointly developed by Intel and the Broad Genomics Institute. Each sample is 10MB. Compute correlations across samples at ~5000 positions.

Intel® Xeon® E5 2697 v2 CPU, 12 cores, dual socket, 128 GB RAM, CentOS6.6, Western Digital 4 TB WD4000F9YZ-0 as a ZFS RAID0 pool. Single thread/core results.
BigDawg: An integrated **polystore** system

**Applications**
e.g., Medical data, astronomy, twitter, urban sensing, IoT

**Visualization & presentation**
e.g., ScalaR, imMens, SeeDB, Prefetching

**SW Development**
e.g, APIs for traditional languages, Julia, GraphMat, ML Base

---

BigDAWG Query Language and Data Federation layer

---

**Stream**

**Real Time DBMSs**
- S-Store

---

**Analytics DBMSs**
- SciDB
- MyriaX
- TupleWare
- TileDB

---

**Analytics**
e.g., PLASMA, ML algos, plsh, GraphBLAS, other analytics packages

---

**Hardware platforms**
e.g., Cloud and cluster infrastructure, NVM simulator, 1000 core simulator, Xeon Phi, Xeon

---

Third party names are the property of their owners
Programming languages in Academia

The academic world has moved into “high productivity” languages. Maybe HPC should give them a try?

Source: http://cacm.acm.org/blogs/blog-cacm/176450-python-is-now-the-most-popular-introductory-teaching-language-at-top-us-universities/fulltext
What is Julia?

• Julia is yet another new language!!
• Started in 2009 in Alan Edelman's group at MIT.

• The problem is ... do we really need a new language? ... computer scientists spend more time creating new languages than making existing languages actually work.

• But people I know and respect have convinced me to take a close look at Julia
  – It is relatively easy to learn
  – A non-viral open source license (MIT License) so corporate types can play with it.
  – A large and growing community ... over 350 contributors since it started in 2009
What is Julia?

- An excellent foundation for programming research.
  - Core functionality of Julia is written in Julia.
  - Benefits from large LLVM eco-system.
  - **Introspection**: Exposes transformations from high level code into native assembly code ... so you can manipulate them inside Julia

```
code_lowered(fib,(Int32))
code_typed(fib,(Int32))
code_llvm(fib,(Int32))
code_native(fib,(Int32))
```
The benefits of introspection

• Using the flexible Julia framework, a group at Intel created C backend integrated with Julia (Prospect ... Open Source release Q4’2015)

• Prospect compiler tool generates optimized C code from Julia Source:
  • Parallelization
  • Loop fusion
  • Domain specific optimizations
  • Vectorization

Prospect vs. Julia

PSE: our Julia based Problem Solving Environment ... of which Prospect is a key component.

Source: Tatiana Shpeisman of Intel

Running on a server with two Intel® Xeon® E5-2690v2 processors at 3 Ghz, 128GB RAM
The benefits of introspection

• Using the flexible Julia framework, a group at Intel created C backend integrated with Julia (Prospect ... Open Source release Q4’2015)

• Prospect compiler tool generates optimized C code from Julia Source:
  • Parallelization
  • Loop fusion
  • Domain specific optimizations
  • vectorization

Running on a server with two Intel® Xeon® E5-2690v2 processors at 3 Ghz, 128GB RAM
What is Julia?

• It is a dynamic scripting language (like python or perl)
• Syntax natural to people working with Math plus a rich set of built in, standard operations (Like Matlab or R).
What is Julia?

A general purpose programming language with sophisticated dataflow type discovery

```plaintext
function fib(n)
    if n < 2
        n
    else
        fib(n-1) + fib(n-2)
    end
end
```

no need to declare parameter types

implicit return value

With a very compact syntax.

```
fib(n) = n < 2 ? n : fib(n-1) + fib(n-2)
```

one line shorthand

time = O(1.618^n)

Source: Arch Robison of Intel
What is Julia?

```plaintext
function fibfast(n)
    @assert n>=1  # Macro
    a = zero(n)  # Set local variable a to zero of same type as n
    b = one(n)   # Set local variable b to one of same type as n
    for i=2:n    # Loop from 2 to n inclusive
        (a,b) = (b,a+b)  # tuple construction and destructuring
    end
    b
end
```

time = O(n)

Source: Arch Robison of Intel
What is Julia?

- Multiple dispatch
- Dynamic polymorphism

```julia
julia> *
  * (generic function with 115 methods)

julia> *(a::Number, g::Function) = x->a*g(x)
  * (generic function with 116 methods)

julia> *(f::Function, t::Number) = x->f(t*x)
  * (generic function with 117 methods)

julia> *(f::Function, g::Function) = x->f(g(x))
  * (generic function with 118 methods)

julia> x=pi/4
0.7853981633974483

julia> (sin^2)(x)
0.6496369390800624
```

Since ^ is generically represented in terms of *, this becomes the composition of functions (sin(sin(x))) ... which is what Gauss wanted it to be.
Distributed Memory Parallelism

```julia
function fibpar(n)
    if n < 30
        fib(n)
    else
        x = @spawn fibpar(n-1)  # Run in parallel
        y = fibpar(n-2)
        y + fetch(x)
    end
end
```

- Each process is single-threaded.
- Julia coroutine mechanism simplifies writing inter-process synchronization code.
- Also has distributed array objects.

Source: Arch Robison of Intel
Shared Memory Parallelism

Kiran Pamnany of Intel has been experimenting with adding threading to Julia ... with some exciting preliminary results

### Productive Performance - Julia

<table>
<thead>
<tr>
<th>Task</th>
<th>matlab</th>
<th>matlab (GPU)</th>
<th>julia</th>
<th>julia (threading)</th>
</tr>
</thead>
<tbody>
<tr>
<td>3D Laplace</td>
<td>685.82</td>
<td>28.6246</td>
<td>24.3793</td>
<td>6.2374</td>
</tr>
<tr>
<td>Monte Carlo</td>
<td>24.4652</td>
<td>24.4652</td>
<td>20.6858</td>
<td>3.012</td>
</tr>
<tr>
<td>3D LBM</td>
<td>72.0695</td>
<td>&gt;560</td>
<td>75.9252</td>
<td>39.1219</td>
</tr>
</tbody>
</table>

Third Party names are the property of their owners.
Julia and TileDB

Conjugate transpose (A*) with TileDB’s Arrays, TileDB’s generic Array Cell types, and flexible iterator framework.

Third Party names are the property of their owners

Source: Jake Bolewski and Stavros Papadopoulos
Summary

• If “One size does not fit all™” ... Then we need Polystore.

• Productivity demands a single interface to multiple stores:
  – BigDAWG API to tie Islands together
  – BQL: the Dream of one Algebra to rule them all

• TileDB: The advantage of a domain specific storage engine.

• Julia: maybe its time to consider a new language?

Trademark held by Mike Stonebraker