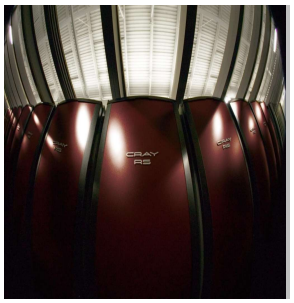

Sandia National Laboratories

Exceptional service in the national interest

Adventures in NoSQL for Scale-Up Computing

HPC-IODC Workshop

Jay Lofstead, Ashleigh Ryan, and Margaret Lawson

June 20, 2019

SAND2019-2166 PE

U.S. DEPARTMENT OF ENERGY

NNSA National Nuclear Security Administration

Sandia National Laboratories is a multimission laboratory managed and operated by National Technology and Engineering Solutions of Sandia, LLC, a wholly owned subsidiary of Honeywell International, Inc., for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA0003525.

Motivation

Sandia National Laboratories

- US DOE Exascale Computing Project Annual
- SIRIUS project

Q: How do we make it easier for scientists to select a data set for deep analysis once it is written?

A: Add user-defined tags and advanced searching capability based on the tags and the data itself.

How did I get here?

Sandia National Laboratories

- Margaret Lawson (UIUC/SNL) developed EMPRESS in 2017
  - SQLite database-based service to manage metadata, particularly for custom metadata
  - Accelerate data exploration based on searching for data tags generated from lightweight analysis during computation or output
  - Challenge: how to query against values for custom metadata tags?

- Insight: Columnar databases allow adding arbitrary additional columns with typed data you can query against (I think). Will this work with performance?

How did I get here?

Sandia National Laboratories

- Fall 2018 intern Ashleigh Ryan (GT) comes to the rescue
  - Agreed to explore this seemingly simple question
  - Did a tremendous job exploring the deadly terrain that is NoSQL databases-–particularly with the intersection of an HPC cluster on a restricted network.
- Primary database choice: Cassandra
  - Secondary: Hbase
- Other NoSQL databases are either key-value, document stores, or otherwise can't work
  - Software requirements that can't be met solely as an end user, it must be run as an OS service, or other non-starter requirements.

Goals and Restrictions

Sandia National Laboratories

- Goals:
  - Scale out tools easily deployable in scale up environment
  - Flexible schema data storage with rich querying capability
- Restrictions
  - Must be open source
  - Must be easily end-user installable without heavy lifting from the system administrators
  - Must work as an embedded engine rather than a permanent service

What else is comparable?

Sandia National Laboratories

- SoMeta
- MDHIM
- HDF5, ADIOS, PnetCDF, NetCDF

None of these address the desired flexibility and performance for managing metadata, and in particular, data tagging and searching.

What NoSQL databases

Sandia National Laboratories

- We (Ashleigh) investigated several options

Database	FOS	Columnar/ KeyValue	Fault Tolerance	General Language
Apache Accumulo	✓	Columnar	Write Ahead + HDFS	Java
Cassandra	✓	Columnar	Replication	Cassandra Query Language
Druid	✓	Columnar	Replication + HDFS	JSON over HTTP
Dynamo	✕	KV	S3	AWS API
Hbase	✓	Columnar	HDFS	Java API
Vertica	✕	Columnar	Varies	SQL

Let's Battle it Out

Sandia National Laboratories

- Four rounds of problems that have to be solved
  - What has the right features to be worth testing
  - What is it going to take to get it working at all
  - Can we make our queries work with any performance
  - Battle scars and lessons for our next battle against scale out computing tools

Round 1. Fight!

Sandia National Laboratories

- NoSQL assumption: running on a cluster with full Internet access from every node with no code deployment restrictions and full control over the storage infrastructure.
  - We fail this test miserably.

- HDFS requirement is a hard “no”
  - We have our storage infrastructure defined already with burst buffer and parallel storage deployed
- Full Internet access from the compute nodes is a hard “no”
  - Getting TCP/IP to work properly is hard enough
- Not getting full source code is a soft “no”
  - We’ll pay for support, but we need source to get it to work on our messy hardware and software.

Round 1. Result

Sandia National Laboratories

- Two potentially viable choices
  - Cassandra – no HDFS and a C-based query API available
  - Hbase – Java-based API and HDFS, but might work
- Chose Cassandra for the potential and active user community for support

Round 2. Fight!

Sandia National Laboratories

- Now to get Cassandra to work, just on a desktop on the restricted network
- Problem 1: What do you mean I need to install a bunch of packages from the Internet?
- Problem 2: What do you mean I have to run it either as a service or as a separate process I interact with?
- Problem 3: What do you mean I have replication? Can't I just do a single storage node?

Round 2. Problem 1 Result

Sandia National Laboratories

- Dependencies are a nightmare
  - This is not unique to Cassandra
- Let's spend a couple of weeks just generating a list of the dependencies and figuring out how to build them from source, in order, so I can build Cassandra.
- Ultimately it worked out, but it is a pain.

Other systems, like Ceph, can be dramatically worse for their list of dependencies. Open source packages are nice, but dozens to hundreds of dependencies is a recipe for weeks of work if you have to do it manually.

Round 2. Problem 2 Result

Sandia National Laboratories

- Cassandra wants to be your storage interface and therefore look like a storage service
- How do you put a TCP/IP-based service on an IB network when TCP/IP isn't the native protocol?
  - Yes, it can work, but configuration isn't straightforward
- You can run it in a window and access that service from another, but it isn't a general solution.
- That C-based API from DataStax works great!

Round 2. Problem 3 Result

Sandia National Laboratories

- Cassandra REALLY wants to use replication—even if you don't
- Running on a single node was easy enough
- Deploying to the cluster and suddenly it wouldn't run anymore
  - Even though the configuration set the replica count to '1'

Result: Even though the configuration said 1 replica, it had to be explicitly set in code to '1' again to make it work properly. The error messages were baffling making debugging a series of trial and error.

Round 3. Fight!

Sandia National Laboratories

- Now that we have something working-ish, how about those fancy queries we want to do?
- Problem 1: Ok, adding a column is straightforward. What do you mean querying has limited functionality against that new column?
- Problem 2: Wow, that is super limited, is there a way to avoid doing table scans regularly?

Round 3. Problem 1 Result

Sandia National Laboratories

- Cassandra's core setup allows only special columns to have an index on them. These are the only ones on which a query can execute efficiently.

- Result: This is completely against what we were trying to do. Is anything else (Hbase?) better?
  - Nope. NoSQL is REALLY limited to get the performance
  - Lots of Internet people say when asked about this, "why would you do that? That isn't how this is supposed to work."
  - Just do table scans for nearly all queries we want to perform.

This is a near fatal blow to the idea. Fortunately, there are still options.

Round 3. Problem 2 Result

Sandia National Laboratories

- Table scans. Really? What can be done?

- Enter: Map-Reduce
- It is possible to embed Spark into Cassandra
  - Wait, that is another set of software dependencies and assorted ugliness.
  - But it can work!
  - Should we switch to Hbase instead because of the built-in Hadoop?

Spark can cache Map-Reduce queries to mimic what we want, but then it is memory bound—and won't tell you when it throws away cached results. It just runs slower when it is out of memory.

Round 4. Fight!

Sandia National Laboratories

- Battle scars and lessons learned
- Problem 1: Can this work at all with performance
- Problem 2: Can this software stack co-exist on scale up platforms
- Problem 3: Is there a way to get what we want?

Round 4. Problem 1 Result

Sandia National Laboratories

- Is there a performant option?
- Bottom line: NO
- Issue: NoSQL is set to query against a limited set of pre-known parameters with the additional columns coming along for the ride.
  - Corollary: Doing a query for the presence or absence of a value for a column is impossible (except for using Map-Reduce).

Rigid relational models are just a better fit for performance, but it is hard to be the right kind of flexible.

Round 4. Problem 2 Result

Sandia National Laboratories

- What about installing this software?
- Bottom Line: Maybe to yes.
- Issue: Lots of dependencies including a heavy assumption of TCP/IP generates terrible network performance and a big software footprint. Getting machine admins to agree to install the behemoth will be difficult.
- This is solvable, but will take a lot of effort. The constant talking to the Internet requires pre-caching things or intercepting version checking. Not impossible, but annoying.

Round 4. Problem 3 Result

Sandia National Laboratories

- Can we make this work?
- Bottom Line: Not using any existing software.
- Issue: relational databases are rigid for performance scalability with arbitrary queries against a fixed schema.
- Issue: NoSQL databases are rigid for performance against a small set of know values to then check if some other attributes have been added. Looking based on attribute isn't a supported model.

Decision

Sandia National Laboratories

- NoSQL has interesting ideas, but we can't use the tools as is on restricted clusters
  - There are caveats that make it possible, but an end-user will find it difficult to impossible making a general library infeasible.
- We have to start over if we want this to work. There is no payoff for industry (or it would be done already) and it is a high-risk prospect for research.
  - HDF5 is the de-facto standard with NetCDF and ADIOS filling in the gaps. Replacing any of these will require solid buy-in from a community (Klasky, as a physicist promoting the tool, was the key for ADIOS to make that jump).
  - But that doesn't mean I won't try ☺

Questions?

gflofst@sandia.gov