Evaluation of Big Data Containers for Popular Storage, Retrieval, and Computation Primitives in Earth Science Analysis

Thomas Clune[$], Kamalika Das[§], Daniel Duffy[$], Ted Habermann[*], Thomas Huang[§§], Kwo-Sen Kuo[$], Chris Mattman[§§], Chaowei Phil Yang[$$]

AGU Fall Meeting, San Francisco, December 2015

[$]NASA Goddard Space Flight Center[$], [§]NASA Ames Research Center, [*]The HDF Group, [§§]NASA JPL , [$$]George Mason University, Contact: Kamalika.Das@nasa.gov

Abstract

- Data containers are infrastructures that facilitate storage, retrieval, and analysis of data sets. Big data applications in Earth Science require a mix of processing techniques, data sources and storage formats that are supported by different data containers. The data containers compared in this study are
  - AsterixDB,
  - RasDaMan,
  - SciDB
  - Hadoop
  - HDF
- These infrastructures optimize different aspects of the data processing pipeline and are, therefore, suitable for different types of applications. These containers are also all undergoing rapid evolution and the ability to re-test, as they evolve, is very important to our handling of the large volumes of observational data and model output. We have identified a selection of steps that are relevant to most data processing exercises in Earth Science applications and we evaluate these systems for optimal performance for each of these steps in the data processing pipeline. The steps evaluated in this study:
  - Hardware/software dependencies
  - Data ingestion
  - Data preparation/processing
  - Data analysis
  - Result reporting

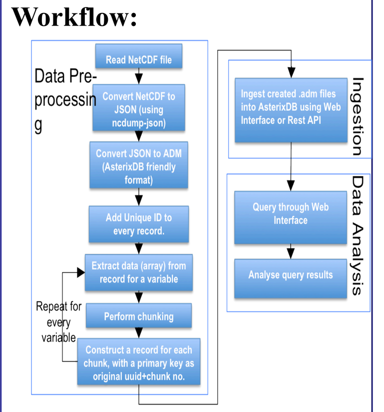
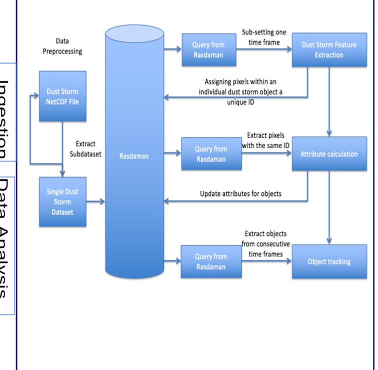
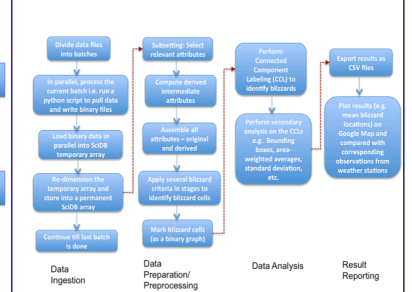
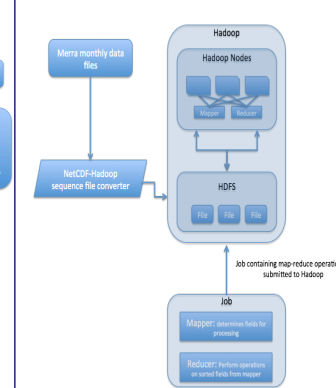
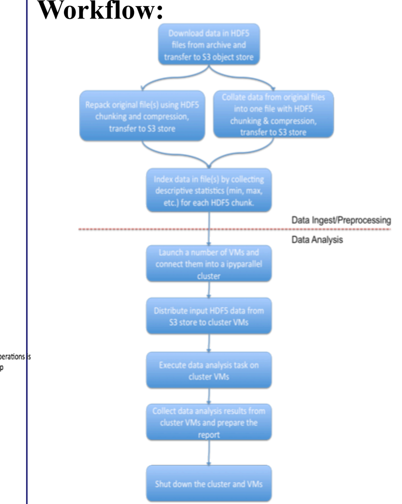
Software and Hardware Dependency

AsterixDB	Rasdaman	SciDB	Hadoop	HDF
Version AsterixDB 0.8.7-Snapshot **Hardware dependency** Share Nothing Architecture, uses controller nodes **Software dependency** JDK 1.7 Password-less SSH configuration	**Version** 9.1 **Hardware dependency** 3GHz, 8GB RAM, 400MB HDD for installation **Software dependency** Git, lib, Tomcat (or another suitable servlet container), Java Runtime Environment (JRE) 1.6 or higher, PostgreSQL 8.x	**Version** 14.12 **Hardware dependency** Distributed file system for data (Optional) shared file system for software **Software dependency** PostgreSQL, Apache Maven, Apache log4cxx, Fedora mock, Google protobuf, ScaLAPACK, Shim, SciDB-Py, SciDB-R, SciDB cluster	**Version** Cloudera 5.0 **Hardware dependency for datanode** 12-24 1-4TB hard disks, 2 quad-/hex-/octo-core CPUs, running at least 2-2.5GHz 64-512GB of RAM, 10Gigabit Ethernet **Software dependency** CentOS OpenVZ for RHEL 6 – LXC version 1.1.3 Inifiniband JDK 1.7.0_67, python, perl	**Version** HDF5 v1.8.15 **Hardware dependency** Xeon, Ethernet, Ephemeral file system, S3 **Software dependency** HDF5 library v1.8.15, h5dump, h5repack, Python 3, h5py, numpy, ipyparallel

Earth Science Application and Data

AsterixDB	Rasdaman	SciDB	Hadoop	HDF
Application: Dynamic data subsetting and statistics aggregation using selected oceanographic data **Data:** GHRSST Level 4 CMC 0.2° Global Foundation Sea Surface Temperature Analysis. Grid size: 1800x901 **Subset Used:** Spatial span: 50 x 50 grid Temporal span: 4 months Size: 2.43 GB	**Application:** Dust storm analysis framework consisting of dust storm feature identification, attribute calculation, and object tracking. **Data:** Non-hydrostatic Mesoscale Dust Model (NMM-dust) fromNCEP, simulating dust event in Phoenix, Arizona during 3rd and 4th of July 2014. Horizontal resolution: 3 km with 45 vertical levels in the vertical and the Vertical Resolution: Between 2.5 KM and ~5 KM Time Resolution: 3 hours. **Subset Used:** Full data set Size: ~1 TB	**Application:** Identify grid cells meeting blizzard conditions using (imprecise) NWS definitions. Identify blizzard events using spatio-temporal CCL and appropriate statistics. Compare results with observed data. **Data:** Modern Era Retrospective Analysis for Research and Analysis (MERRA) Spatial resolution: ⅔°x ½° Hourly resolution: Hourly **Subset Used:** Winter 2010 (DJF) time period with 16 attributes from MERRA (MAT1NXFLX, MAT1NXSLV, MAT1NXLND, MACONXASM) Size: ~25 GB	**Application:** Climatology research to enable simple canonical operations including subsetting, averaging, searching for minimum and maximum values, etc. **Data:** Modern Era Retrospective Analysis for Research and Analysis (MERRA) Spatial resolution: ⅔°x ½° Hourly resolution: Hourly **Subset Used:** MERRA data for northern India/Pakistan, North China Plain, California Central Valley,, and Nile Valley Size: ~ 132 GB	**Application:** Supporting multiple applications and various data sets **Data:** • NCEP/DOE Reanalysis II, for GSSTF, Daily Grid, V3 Spatial: 0.25°x0.25°, global Temporal: 1987-08, daily • NOAA Coral Reef Temperature Anomaly Database Spatial: ~4km global Temporal: 1982-12, weekly **Subset Used:** Full data set Size of NCEP/DOE Reanalysis2 ~ 17GB NOAA Coral Reef temperature data ~ 24MB

Data Ingestion and Workflow

AsterixDB	Rasdaman	SciDB	Hadoop	HDF
Time to convert to .adm file: ~7541 sec ~ 32 sec/ MB **Disk space required:** Raw data ~ 235 MB AsterixDB format ~2.43 GB 10 fold increase in disk space requirement **Workflow:** 	**Time to convert Geotiff file:** ~1 sec/MB Can be parallelized **Disk space required:** Raw data ~ 45 MB AsterixDB format ~77 MB Less than 2 fold increase in disk space requirement **Workflow:** 	**Time to convert file:** Pull data on the fly (OPenDAP); Write (1-D) binary data); Load 1-D and redimension): ~0.3 sec/MB Can be parallelized **Disk space required:** ~2.5 fold increase in disk space requirement **Workflow:** 	**Time to convert to HDFS file:** Involves sequencing, mapping, and then using Bloom filter for reducer ~0.4 sec/MB Can be parallelized Disk space required: ~2.5 fold increase in disk space requirement **Workflow:** 	**Time to convert HDF file:** Data sets are re-chunked and compressed ~ 0.2 sec/MB Can be parallelized **Disk space required:** 63% reduction in file size **Workflow:** 

Data Preparation/Preprocessing

AsterixDB	Rasdaman	SciDB	Hadoop	HDF
Operations • Subsetting (50x50 chunks) • Sorting (**bugs in current version) Operations are parallelizable **Bottlenecks** • For parallel sorting (e.g., parallel order by) final merge is done on one node, which does not scale. • The CC (master node) compiles all queries. To be changed in future versions. • Current design requires returning entire result of a query in a single object. Current max size allowed for objects may not suffice for large queries.	**Operations** None: Data ingested in Geotiff format and entire data set used. **Bottlenecks** • Physical memory and disk I/O are main performance bottlenecks • Performance can be really slow when subsetting portions of source images	**Operations** • Subsetting • Table join • Constructing alternative representation Operations are parallelizable **Bottlenecks** • Operations are local, little or no data exchange • Performance bottlenecks are being investigated	**Operations** • Write custom NetCDF to Hadoop convertor to keep files as sequence files • These files sent to Hadoop for storage • Hadoop splits and distributed the sequence files across HDFS; builds index for Hadoop access • Maintains NetCDF metadata for each file Operations are parallelizable **Bottlenecks** • Processing is offline mode – not useful for adhoc queries • Very large and skewed data causes memory issues both at mappers and reducers	**Operations** • HDF5 dataset chunks with all-missing data not stored during the data ingest stage. • No subsetting; entire temporal and spatial extent of data is used after ingestion • Sorting only in the temporal domain, if required, to ensure monotonic order of the temporal axis. • Data are indexed by calculating descriptive statistics for each HDF5 dataset chunk. • Initial data files are collated into a single file with optimized HDF5 dataset chunking/ compression. Operations are parallelizable **Bottlenecks** • File granularity (inefficient to copy same file to multiple nodes if number of nodes > number of files). • One processor performs aggregation of results, could result in bottleneck depending on data.

Acknowledgement: This research is supported by funding from the NASA ESTO-AIST Program.

Data Analysis

AsterixDB	Rasdaman	SciDB	Hadoop	HDF
Primitives Tested Standard statistics computation (mean, std deviation) • Calculations cannot be performed on 50x50 chunks due to unresolved bugs in software. Computing similarity or distance between every pair of records • Supports edit distance (on strings) and Jaccard coefficient (on sets) Largest connected subgraph search • Can be integrated with Pregelix for graph computation.	**Primitives Tested** Extract individual dust storm object (region-growing based algorithm) Mean computation • Done using queries Example queries -Select a single pixel from all images ~66msec -Select a subset from all images ~1sec Select mean value of each band of a single image ~0.3sec Select mean value of each band across all images ~4.5sec	**Primitives Tested** Finding all connected components in a graph ~0.688 (μsec.core)/ data_point Observations: Data exchange is expensive across nodes	**Primitives Tested** Use of Bloom filter to speed up Hadoop jobs by leveraging the probabilistic search capability. Speed up by 30-80% obtained Example performances: • 83.9% efficient for reading a single parameter (“T”) from a single sequenced monthly means file • 29% efficient for single MR job across 4 months of data seeking “T” (period = 2)	**Primitives Tested** Standard statistics computation (mean, std deviation) • Calculation performed on original data (as obtained from the archive) ~Single node: 5.4 sec/GB ~50 nodes: 0.05 sec/GB Clustering • Searching for points/ regions based on a set of temporal, spatial, data value conditions Data subsetting • Slicing; selectionbased on temporal, spatial, data value criteria

Result Reporting

AsterixDB	Rasdaman	SciDB	Hadoop	HDF
Graph Plotting: • Pregelix supports parallel graph computations using the Pregel programming model • Query results in json format can be used as input to visualizations (software or web visualizations) • Cannot be used for plotting figures with overlay for showing results on the Earth’s grid • Cannot be automated	**Graph Plotting:** • Provides several Open Geospatial Consortium (OGC) standard interfaces through its web services wrapper, Petascope. • Can be used to plot figures with geographic overlays • Plotting can be automated but using spatial/temporal indexing which would require Petascope to store the temporal and spatial metadata	**Graph Plotting:** • Plotting is possible using “shim” SciDB client to interact with external tools like SciDB-py and SciDB-R • Exporting data also possible (for other external tools) • Figures with overlay can also be plotted using SciDB-py or SciDB-R. • Automation can be done via scripting around SciDB-Py or SciDB-R	**Graph Plotting:** • Plotting is possible by exporting data to standard formats and using external plotting software • Visualization tool IDL can be used to visualize and diagnose data stored in the native Hadoop file format, HDFS • Process can be made faster by using parallel reader for data ingestion before visualization	**Graph Plotting:** • Hdf file formats allows storage of meta information that can be used for plotting results using overlays.

Observations

- AsterixDB is inefficient for big data applications because its storage format requires 10x more disk space than raw archive format. Current version has many bugs.
- Hadoop requires significant parameter tuning for optimal performance and has high bandwidth requirements.
- Most containers suffer from parallelization bottlenecks due to aggregation/merging of results at a single node
- HDF files can cause issues during concurrent read/copy in multicore architectures
- Rasdaman can be slow for large I/O operations and inefficient for big data applications. Also development support for Rasdaman is also low compared to some other containers
- SciDB data format is not compatible with other common big data processing frameworks thereby requiring duplicate data storage.

Additional Contributors		
John Thompson, NASA Goddard Namrata Malaraut, NASA JPL	Fei Hu, George Mason University John Ready, HDF Group	Aleksander Jelenak, HDF Group Amidu Oloso, NASA Goddard