Application of Open-source Spatio-Temporal Database Systems in Wide-Field Time-domain Astronomy

Seo-Won Chang[1] and Min-Su Shin[2]

[1]Yonsei University Observatory, Seoul, Korea

[2]Korea Astronomy and Space Science Institute, Daejeon, Korea

E-mail: seowony@galaxy.yonsei.ac.kr and msshin@kasi.re.kr

Abstract

- We present our on-going experience with open-source spatio-temporal database systems that are optimized to manage both spatial and time information for analyzing large volumes of astronomical data acquired by wide-field time-domain surveys, such as KMTNet (Korea Microlensing Telescope Network) or upcoming LSST.
- Considering performance, cost, and difficulty of the database systems, we conduct comparison studies of two spatio-temporal databases (GeoMesa and PostGIS) that are already being used for handling big geo-spatial data. Our experiments include ingesting, transforming, indexing, and querying millions or billions of astronomical spatio-temporal features using both systems.
- We discuss the performance and limitations of these spatio-temporal database systems to be utilized for astronomical applications: easiness of use, functionalities (e.g., indexing scheme, supported query functions), and speed of computation.

Open-source Spatio-Temporal Databases: GeoMesa and PostGIS

GeoMesa is an open-source, distributed spatio-temporal database that manages big geo-temporal data within the Accumulo key-value data store (i.e., NoSQL DB) so that those data can be indexed and queried at scale effectively. Meanwhile, as an extension to the PostgreSQL, PostGIS is also an open-source database which adds support for geospatial objects and queries.

- Set-up: We used the most recent version of each database system at the time of our testing; GeoMesa 1.2.6 and PostGIS 2.2.1. Since the test hardware consists of only single server, GeoMesa has no benefits in using Hadoop's distributed-computing framework. Our experiments are only for testing purpose.
- Prerequisite condition: GeoMesa is not a standalone system, making the whole configuration process complex and error-prone. Three core components and their related functions are required for running the GeoMesa Tools: (i) Hadoop Distributed File System (HDFS), (ii) Zookeeper's coordination system, and (iii) a highly scalable structured store (Accumulo).

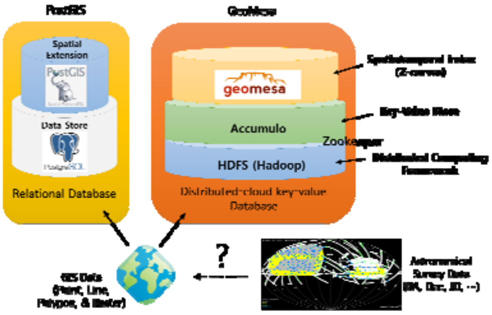
PostGIS
GeoMesa
Spatial Extension
PostGIS
Data Store
PostgreSQL
Relational Database
geomesa
Accumulo
HDFS (Hadoop)
Zookeeper
Distributed-cloud key-value Database
Spatiotemporal Index (Z-curve)
Key-Value Store
Distributed Computing Framework
?
GIS Data (Point, Line, Polygon, & Raster)
Astronomical Survey Data (RA, Dec, JD, ...)

Figure 1. *Simplified PostGIS (Left) and GeoMesa (Right) architectures.*

Model	Dell T610
CPU	Two HT 4-core Intel Xeon Processor E5640 2.66GHz
RAM	24 GB (1333MHz) Dual Rank LV RDIMM
OS kernel/system	Linux 4.4.0-38 x86_64/ Ubuntu 16.04 LTS

Table 1. *Experimental Hardware specification*

Building Data Stores

In order to ingest the spatio-temporal data sets, we should transform the data format such as delimited text or JSON, and then convert the data into the "SimpleFeatures". This data scheme specifies a common storage and access model of mostly two-dimensional geographical data (e.g., point, line, polygon).

- Limitation of Data Transformation: Unlike the converters for PostGIS, GeoMesa use a predefined spatial/temporal reference systems (Geometry: World Geodetic System 84; Date: Unix/Java-style timestamp). Thus, we reproject our data to that reference system before ingesting it into GeoMesa.
- Complex Indexing scheme: The uniqueness of GeoMesa's index is that it implements a space filling curve (Z-Curve) to combine three-dimensions of geometry and time (i.e., longitude, latitude, and time) into a single-dimension lexicographic key space.

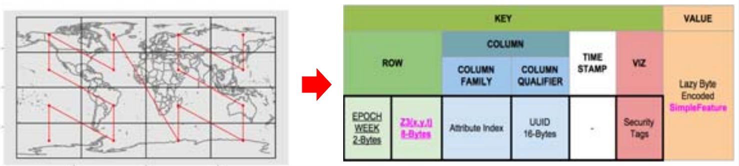

Figure 3. *GeoMesa's data indexing scheme (Left) and the real structure of index in Accumulo (Right). The red line shown in left panel is known as a Z-curve. The Z3 encoding (x,y,t) shown in right panel is used to efficiently answer queries of features with point geometry with both spatial and temporal components.*

Experimental Data: VVV DR4

We chose the public VVV (VISTA Variables in the Via Lactea) catalogs of billions measurements for hundreds of millions objects as the test data.

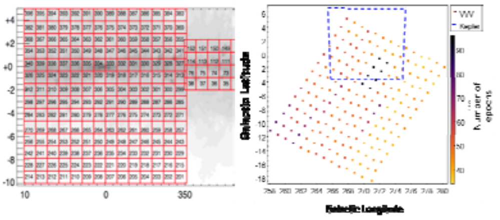

Figure 2. *VVV Survey area for Bulge (Left) and selected VVV tiles overlapped with Kepler K2 field 9 (Right; blue box).*

- Use K_s multi-epoch data for obtaining spatio-temporal data sets.
- Overlapped with Kepler K2 field 9: 7 tiles (0.04% to the total)
- Total 408,970,029 rows with 7 attributes (GlobalID, RA, DEC, MJD, etc).

Preliminary Results

We first examine the data ingestion performance with difference sizes of data (Fig 4) and then check the query execution times for varying conditions (Fig. 5 and Fig. 6).

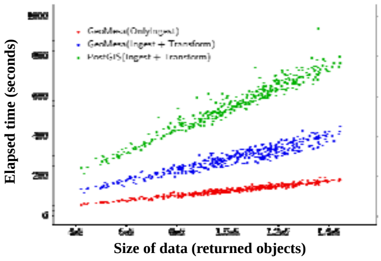

Figure 4. *Data ingest times for varying data sizes.*

- Successfully ingested ~4 billion features with no failures on both database systems.
- The total elapsed time of GeoMesa (14.26 hours) is much shorter than those of PostGIS (61.72 hours) → The maximum speed achievable in our tests is below 8~10k records/s per server.

We randomly generate (i) 300 pairs of spatial (RA, DEC) quires with different searching radius and (ii) 300 x 10 pairs of spatial-temporal (RA, DEC, $\mathrm{MJD}_{\mathrm{start}}$, & $\mathrm{MJD}_{\mathrm{end}}$) queries, respectively.

- In terms of spatial queries, GeoMesa returns more rapid query response compared to PostGIS.
- The number of object returned is a significant factor in query performance.

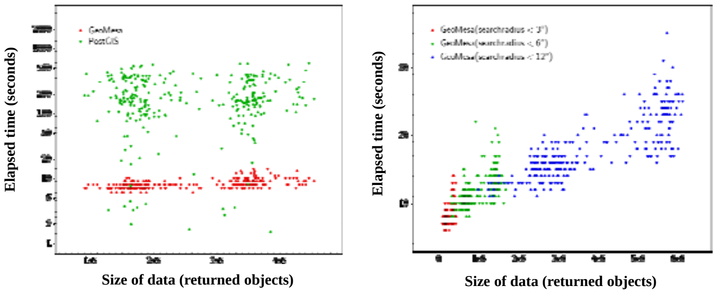

Figure 5. *Spatial query execution times for varying data sizes. (Left) GeoMesa vs. PostGIS with the same condition; (Right) Only GeoMesa's query performances for varying search radius (3", 6", and 12").*

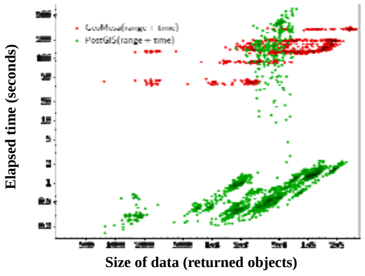

Figure 6. *Spatial-temporal query execution times for varying data sizes in GeoMesa and PostGIS.*

- For spatio-temporal queries, GeoMesa's query execution times were much slower than those of PostGIS. This is unexpected result as compared to spatial query test.
- Depending on the selected time intervals in GeoMesa, some queries with different time intervals can result in too many ranges. We suspect that this is a primary cause of query slowness → We are trying to ascertain the reasons.

Planned Experiments

- Tuning ingest/query performance of GeoMesa: Memory & Server side parameter setting.
- Measuring the ingest & query performances on the small clusters in KISTI (Korea).