In-situ processing of big raster data with command line tools[*]

R.A. Rodriges Zalipynis

National Research University Higher School of Economics, Moscow, Russia

Explosive growth of raster data volumes in numerical simulations, remote sensing and other fields stimulate the development of new efficient data processing techniques. For example, in-situ approach queries data in diverse file formats avoiding time-consuming import phase. However, after data are read from file, their further processing always takes place with code developed almost from scratch. Standalone command line tools are one of the most popular ways for in-situ processing of raster files. Decades of development and feedback resulted in numerous feature-rich, elaborate, free and quality-assured tools mostly for a single machine. The paper reports current development state of ChronosServer – distributed system partially delegating in-situ raster processing to external tools. The new delegation approach is anticipated to readily provide rich collection of raster data operations at scale.

Keywords: big raster data, distributed processing, command line tools, delegation approach.

1. Introduction

Raster is the primary data type in a broad range of subject domains including Earth science, astronomy, geology, remote sensing and other fields experiencing tremendous growth of data volumes. For example, DigitalGlobe – the largest commercial satellite imagery provider, collects 70 terabytes of imagery on an average day with their constellation of six large satellites [1].

Traditionally raster data are stored in files, not in databases. The European Centre for Medium-Range Weather Forecasts (ECMWF) has alone accumulated 137.5 million files sized 52.7 petabytes in total [2]. This file-centric model resulted in a broad set of raster file formats highly optimized for a particular purpose and subject domain. For example, GeoTIFF represents an effort by over 160 different remote sensing, GIS (Geographic Information System), cartographic, and surveying related companies and organizations to establish interchange format for georeferenced raster imagery [3].

The corresponding software has long being developed to process raster data in those file formats. Many of them are free, popular and have large user communities that are very accustomed to them. For example, ImageMagic is under development since 1987 [4], NCO (NetCDF common operators, a tool for multidimensional arrays), since about 1995 [5]; Orfeo ToolBox – remote sensing imagery processor now represents over 464,000 lines of code made by 43 contributors [6]. Many tools take advantage of multicore CPUs (e.g. via OpenMP), but most of them work on a single machine.

In-situ distributed raster data processing has recently gained increased attention due to explosive growth of raster data volumes in diverse file formats. However, already existing stable and multifunctional tools are largely ignored in this research trend. Thus, raster operations are re-implemented almost from scratch delaying emergence of a mature in-situ distributed raster DBMS.

This paper describes the prototype extension of ChronosServer [7, 8] leveraging existing command line tools for in-situ raster data processing on a computer cluster of commodity hardware. Unlike current systems, it is easier and faster to equip ChronosServer with wide variety of raster operations due to new delegation approach. Thus, it is anticipated that it is possible to quickly develop new distributed file-based raster DBMS with rich functionality and quite decent performance.

2. In-situ raster data processing

In-database data storage (in-db, import-then-query) requires data to be converted (imported) to internal database format before any queries on the data are possible. Out-of-database (out-db, in-situ, file-based, native) approach operates on data in their original (native) file formats residing in a standard filesystem without any prior format conversions.

[*] This work was partially supported by Russian Foundation for Basic Research (grant #16-37-00416).

2.1 State-of-the-art

PostgreSQL extensions PostGIS [9] and RasDaMan [10] work on single machine and allow registering out-database raster data in file system in their native formats. SAGA [11] performs only distributed aggregation queries over data in HDF format. SWAMP [12] accepts shell scripts with NCO and parallelizes their execution. Hadoop extensions SciHadoop [13] and SciMATE [14] implement drivers reading Hadoop DFS chunks as if they are in HDF or NetCDF formats. Galileo [15] indexes geospatial data with distributed geo-hash. Only PostGIS is available in production with poor performance on multidimensional arrays (e.g. NetCDF, HDF or Grib formats [16]). SWAMP launches command line tools but focuses on NCO and requires scripts looping over files with explicitly specified file names. The proposed approach is universally applicable to any tool and abstracts from "file" notion at all.

Commercial ArcGIS ImageServer [17] claims in-situ raster processing with custom implementation of raster operations. However, in a clustered deployment scenario all cluster nodes are recommended to hold copies of the same data or fetch data from a centralized storage upon request what negatively impacts scalability. Commercial Oracle Spatial [18] does not provide in-situ raster processing [19]. Open source SciDB is specially designed for distributed processing of multidimensional arrays [20]. However, it does not operate in-situ and imports raster data only converted to CSV format – very time-consuming and complex undertaking. Moreover, SciDB lacks even core raster operations like interpolation which makes it an immature and not widely used product [21]. Open source TileDB has been released on 04 Apr. 2016 and it is yet neither distributed nor in-situ enabled [22].

Hadoop [23] and experimental SciDB streaming [24] allow launching a command line tool, feed text or binary data into its standard input and ingesting its standard output. Note two time-consuming data conversion phases in this case: data import into internal database format and their conversion to other representation to be able to feed to external software. The proposed approach directly submits files to external executables without additional data conversion steps.

2.2 In-situ approach benefits

This section collects in one place advantages and challenges of in-situ approach that are quite scattered in the published literature.

- *Avoid inefficient neighborhood*. Traditionally, BLOB (Binary Large OBject) data type served for in-db raster storage (PostGIS, RasDaMan). Physical layouts where raster data are close to other data types are quite inefficient since the former are generally much larger than the latter.
- *Leverage powerful storage capabilities*. Some raster file formats support chunking, compression, multidimensional arrays, bands, diverse data types, hierarchical namespaces and metadata. These techniques are fundamental for raster storage; their implementation for an emerging in-db storage engine results in yet another raster file format.
- *Avoid conversion bottleneck*. In mission-critical applications it is important to be able to analyze the data before their new portion arrives or a certain event happens. In some cases the conversion time may take longer than the analysis itself. The data arrival rate and their large volumes may introduce prohibitively high conversion overheads and, thus, operational failure.
- *Avoid additional space usage*. Most data owners never delete source files after any kind of format conversions including database import. There are numerous reasons for this including unanticipated tasks that may arise in future that are more convenient, faster, easier or possible to perform on the original files rather than their converted counterparts. Storing both source data and their in-db copies requires additional space that may be saved by in-situ approach.
- *Reduce DBMS dependence.* It is easier to migrate to other DBMS keeping data in a widely adopted storage format independent from a DBMS vendor.
- *Leverage other software tools (this paper)*. Out-db raster data in their native formats remain accessible by any other software which was inherently designed to process file-based data.

Key difficulties lie in the ability to perform the same set of operations on data in different file formats. Three data models are most widely used that allow abstracting from file format: Unidata CDM, GDAL Data Model, ISO 19123 (not cited due to space constraints). Most existing command line tools use those models and, thus, are capable to handle data in diverse raster file formats.

3. ChronosServer architecture

3.1 Raster data model: abstracting from files, their locations and formats

This paper focuses on climate reanalysis and Earth remote sensing global gridded raster data represented as multidimensional arrays and usually stored in NetCDF, Grib and HDF file formats. For example, AMIP/DOE Reanalysis 2 (R2) spans several decades, from 01.01.1979 to current date with 6 hour time interval and contains over 80 variables [25]. The grid resolution is 2.5° × 2.5°. Global grids for each variable are stored in a sequence of separate files partitioned by time. File names contain variable codename, e.g. files with surface pressure are named pres.sfc.1979.nc, pres.sfc.1980.nc, …, pres.sfc.2015.nc. Where "pres.sfc" denotes surface pressure, 1979 is year, ".nc" is NetCDF file extension. Usually file naming is much more complex (e.g., compare to AIRS/AMSU daily file name for CO_2 satellite data: AIRS.2004.08.01.L3.CO2Std001.v5.4.12.70.X09264193058.hdf). Note, that the data are usually already split by files by data providers.

ChronosServer distributes files among cluster nodes without changing their names and formats. Any file is always located as a whole on a machine in contrast to parallel or distributed file systems. It introduces a data model to work with grids, not files to abstract from "file" notion, file naming, their locations, formats and other details that are unique to every dataset and not relevant for data analysis. ChronosServer dataset namespace is hierarchical. For example, "r2.pressure.surface" refers to surface pressure of R2 reanalysis. ChronosServer provides SQL-like syntax for subsetting grids. For example, "SELECT DATA FROM r2.pressure.surface WHERE TIME_INTERVAL = 01.01.2004 00:00 - 01.01.2006 00:00 AND REGION = (45, 60, 50, 70)" returns time series of R2 surface pressure in the specified time interval and region between 45°S – 50°N and 60°W – 70°E. The query execution may involve several cluster nodes. Any grid or time series from any dataset may be extracted with the same syntax regardless of original file format, file split policy and other details.

3.2 Cluster orchestration

ChronosServer cluster consists of workers launched at each node and a single gate at a dedicated machine. Gate receives client queries and coordinates workers responsible for data storage and processing. All workers have the same hierarchy of data directories on their local filesystems. A worker stores only a subset of all dataset files and only a portion of the whole namespace relevant to the data it possesses. A file may be replicated on several workers for fault tolerance and load balancing. It is not required to keep all workers up and running for the whole system to be operational.

The gate is unaware of file locations until a worker itself reports them to it. This is done for better scalability and fault tolerance. Upon startup workers connect to gate and receive the list of all available datasets and their file naming rules. Workers scan their local filesystems to find out what datasets and time periods do they hold by parsing dataset file names. Workers transmit the list of time intervals for each dataset they store to the gate. Gate keeps this information in worker pool – in-memory data structure used during query execution that maps time intervals to their respective owners (workers).

4. New delegation approach

4.1 ChronosServer raster data processing commands and their distributed execution

ChronosServer syntax of a raster data processing command is the same as launching a tool from a command line. Command names coincide with names of existing command line tools. ChronosServer command options have the same meaning and names as for the tool but without options related to file names or paths. Commands and tools also support options with long names having the same meaning.

For example, NCO consists of several standalone command line tools: ncap2 (*A*rithmetic *P*rocessor v.2), ncwa (*W*eighted *A*verager), ncatted (*Att*ribute *Ed*itor – metadata manager, Fig. 1), etc. [5]. Metadata are crucial component of any raster data. NetCDF and many other formats store metadata as attributes (key-value pairs). For example, attribute named "_FillValue" holds a constant used to mark raster cells with missing values (e.g., -9999).

```
ncatted [-a ...] [--bfr sz] [-D nco_dbg_lvl] [--glb ...] [-h] [--hdr_pad nbr]
        [-l path] [-O] [-o out.nc] [-p path] [-R] [-r] [-t] in.nc [[out.nc]]

-a, --attribute attribute_name,variable_name,mode,attribute_type,attribute_value
      mode = a,c,d,m,o (append, create, delete, modify, overwrite)
      att_typ = f,d,l/i,s,c,b (float, double, long/int, short, char, byte)
    --bfr_sz, --buffer_size sz    Buffer size to open files with
-D, --dbg_lvl, --debug-level lvl  Debug-level is lvl
    --glb nm=val                  Global attribute to add
-h, --hst, --history              Do not append to "history" global attribute
    --hdr_pad, --header_pad       Pad output header with nbr bytes
-l, --lcl, --local path           Local storage path for remotely-retrieved files
-o, --output, --fl_out out.nc     Output file name (or use last argument)
-O, --ovr, --overwrite            Overwrite existing output file, if any
-p, --pth, --path path            Path prefix for all input filenames
-R, --rtn, --retain               Retain remotely-retrieved files after use
-r, --revision, --version         Compile-time configuration and program version
-t, --typ_mch, --type_match       Type-match attribute edits
in.nc [[out.nc]]                  Input file name [[Output file name]] (or use -o)
```

Fig. 1. Parameters of the NCO ncatted tool (all listed) and Chronos ncatted command (marked bold)

By default, command is applied to the whole available dataset time interval and spatial coverage. They may be restricted by "select" query with alias dataset name specification. New virtual dataset will contain subset of the original dataset. Its name (alias) may be used in the subsequent commands. It is helpful to test a series of commands on a dataset sample to check hypotheses about the anticipated results before submitting large-scale query involving large data volumes to save time.

For example, ChronosServer command for "_FillValue" attribute deletion from dataset "r2.pressure.surface" is "ncatted -a _FillValue,r2.pressure.surface,d,,". Instead of "variable_name" – the term specific for NetCDF format, ChronosServer ncatted accepts a dataset name to be independent of a concrete format. Usually the same attributes are duplicated in all files of a dataset.

The gate receives and parses command line options, verifies their correctness and absence of malicious instructions since they are passed to operating system shell. The dataset or its subset is locked for reading/writing depending on the command. Several commands may work concurrently if they do not block each other. Gate selects workers on which dataset files with the required time/space intervals are located and sends them the modified command (see below). Workers complement command line with full paths to dataset files according to time and space limitations and launch the tool on each file.

In the simple case above, ChronosServer invokes several instances of the NCO ncatted tool on the cluster nodes where at least one dataset file is located (pres.sfc.1979.nc, …, pres.sfc.2015.nc). The execution command line for file pres.sfc.1979.nc is "<path to ncatted.exe> -a missing_value,pres,d,, <data path>\ pres.sfc.1979.nc". The file path and "pres" were automatically put by worker and gate correspondingly. The latter is the NetCDF variable name that stores R2 surface pressure (NetCDF3 format does not have hierarchical namespace and stores data in structures called "variables").

Workers also collect standard output of the tool which is sent to gate after its completion. Running tool on a different cluster node in case of a hardware failure is under development. Gate reports to the user once it receives success messages from all workers involved in the command execution. Report contains the merged standard outputs from each run of the tool and total elapsed time.

4.2 Distributed apply-combine-finally execution scheme (under development)

Raster operations can be broadly classified as global (involve all data), local (pixel-wise), focal (cell values from a rectangular window are required to compute new cell value), zonal (same as focal but spatial region is defined by a function) [26]. Thus, some operations cannot be completed autonomously using data on a single cluster node. For example, R2 data interpolation for 1980 year from 6 to 3 hour time step requires grids for December 1979 and January 1981. Also, computation of maximum mean winter summer pressure involves all files potentially located on different cluster nodes.

In this case, user specifies commands: APPLY command1 INTERVALS intervals COMBINE command2 FINALLY command3. ChronosServer ensures that files on each involved node contain data in given temporal and spatial intervals (e.g., in case of winter means for 1980-1990, there should be nodes with data for all winter months for at least two consecutive years: 1980-1981, 1981-1982, 1982-1983, etc. to be able to compute the mean). This may require data movement between cluster nodes. After the intervals requirement is met, command1 is executed autonomously on the data intervals on corresponding cluster nodes. Since some nodes may have several disjoint intervals, their intermediate results may be combined on the same node to reduce network traffic with command2 if it is possible (e.g. compute maximum of the means of intervals 1980-1981 and 1982-1983 that happened to reside on the same node). All results are gathered on a single node and command3 is applied to obtain final result (e.g., find maximum of means or maximums of maximums if combine phase was applied).

Unlike existing schemes [27, 28], the proposed distributed execution scheme takes into account peculiarities of raster operations, geospatial data and ChronosServer file-based storage model. For example, respective intervals must be specified to guarantee the raster operation (command1) is possible to accomplish within a single node. It is widely recognized that numerous data processing tasks are much easier to parallelize once actions are expressed in functional style, not “for” loops. Note, that to date there are no in-situ distributed raster DBMS.

4.3 Benefits of the proposed delegation approach

While in-situ approach leverages benefits of already existing sophisticated file formats, delegation approach leverages benefits of already existing standalone command line tools.

- *Avoid learning new language.* ChronosServer provides command line syntax that is well-known to every console user instead of a new SQL dialect.
- *Steep learning curve*. Users work with ChronosServer as if with console tools they have accustomed to with only minor changes to already familiar command line options.
- *Documentation reuse*. Most of the tool’s documentation is applicable to the corresponding ChronosServer command due to exactly the same meaning and behavior of the parameters.
- *Output conformance*. Output files are formatted as if a tool was launched manually.
- *Language independence*. ChronosServer may use tools written in any programming language.
- *Community support*. Bugs in tools are fixed by their developers as well as new functionality added, usage suggestions via mail lists are obtained regardless of ChronosServer context.
- *Zero-knowledge development (0-know dev.)*. Developers of existing and emerging tools do not have to know anything about ChronosServer in order the tool could be used in ChronosServer.

The main difficulty of the proposed approach lies in the correct specification of the “intervals”, “combine” and “finally” clauses. Meta-commands are a possible simplification of the problem. They consist of a single command line which translates to predefined apply-combine-finally clauses.

5. Conclusion

The paper presented new approach with numerous benefits and delegating in-situ raster data processing to existing command line tools. The approach is under development as an extension to ChronosServer – inherently distributed, file-based system for high performance raster data dissemination [8]. To date, the only available distributed raster DBMS are SciDB and Oracle Spatial that do not operate in-situ. The former lacks even core raster operations and the latter is commercial. Thus, only the performance of arithmetical raster operations of ChronosServer and SciDB may be compared in future. The ChronosServer raster processing extension is scheduled for release at the end of 2016.

References

1. Launching DigitalGlobe's Maps API | Mapbox. URL: https://www.mapbox.com/blog/digitalglobe-maps-api/ (accessed 25.05.2016).
2. M. Grawinkel et al., Analysis of the ECMWF Storage Landscape, 13th USENIX Conference on File and Storage Technologies, P. 2, February 16–19, 2015, Santa Clara, CA. URL: https://usenix.org/system/files/login/articles/login_june_18_reports.pdf (accessed 26.05.2016).
3. GeoTIFF. URL: http://trac.osgeo.org/geotiff/ (accessed 20.05.2016).
4. ImageMagic: History. URL: http://imagemagick.org/script/history.php (accessed 25.05.2016).

5. NCO 4.6.1-alpha01 User Guide. URL: http://nco.sourceforge.net/nco.html (accessed 13.06.2016).
6. The Orfeo ToolBox on Open Hub. URL: https://www.openhub.net/p/otb (accessed 25.05.2016).
7. Rodriges Zalipynis R.A., ChronosServer: real-time access to "native" multi-terabyte retrospective data warehouse by thousands of concurrent clients, Informatics, cybernetics and computer engineering, P. 151–161. Vol. 14 (188), 2011.
8. ChronosServer, URL: http://www.wikience.org/chronosserver/ (accessed 13.06.2016).
9. Chapter 5. Raster Data Management, Queries, and Applications, URL: http://postgis.net/docs/manual-2.2/using_raster_dataman.html (accessed 31.05.2016).
10. P. Baumann, A. Dumitru, V. Merticariu, The Array Database That Is Not a Database: File Based Array Query Answering in rasdaman // SSTD 2013, LNCS 8098, P. 478–483, 2013.
11. Y. Wang, A. Nandi, G. Agrawal, SAGA: Array Storage as a DB with Support for Structural Aggregations // SSDBM'14, June 30 - July 02, 2014.
12. L. Wang et al., Clustered workflow execution of retargeted data analysis scripts // CCGRID, 2008.
13. J.B. Buck, N. Watkins, J. LeFevre, K. Ioannidou, C. Maltzahn, N. Polyzotis, and S. Brandt. SciHadoop: Array-based Query Processing in Hadoop // In Proceedings of SC, 2011.
14. Y. Wang, W. Jiang, and G. Agrawal. SciMATE: A Novel MapReduce-Like Framework for Multiple Scientific Data Formats // In Proceedings of CCGRID, P. 443–450, May 2012.
15. M. Malensek, S. Pallickara, Galileo: A Framework for Distributed Storage of High-Throughput Data Streams, Proc. of the 4th IEEE/ACM Intl. Conf. on Utility and Cloud Computing, 2011.
16. NetCDF. URL: http://www.unidata.ucar.edu/software/netcdf/docs/ (accessed 13.06.2016).
17. ArcGIS for Server | Image Extension, URL: http://www.esri.com/software/arcgis/arcgisserver/extensions/image-extension (accessed 13.06.2016).
18. Oracle Spatial and Graph, URL: http://www.oracle.com/technetwork/database/options /spatialandgraph/overview/index.html (accessed 13.06.2016).
19. Georaster: Import very large images with sdo_ge... | Oracle Community, URL: https://community.oracle.com/thread/3820691?start=0&tstart=0 (accessed 31.05.2016).
20. Paradigm4: Creators of SciDB, URL: http://scidb.org/ (accessed 13.06.2016).
21. Interpolation - SciDB usage - SciDB Forum, URL: http://forum.paradigm4.com/t/interpolation/1283 (accessed 13.06.2016).
22. TileDB - Scientific data management made fast and easy, URL: http://istc-bigdata.org/tiledb/index.html (accessed 15.06.2016).
23. Hadoop Streaming. URL: wiki.apache.org/hadoop/HadoopStreaming (accessed 19.07.2016).
24. GitHub - Paradigm4/streaming: Prototype Hadoop streaming-like SciDB API, URL: https://github.com/Paradigm4/streaming (accessed 31.05.2016).
25. NCEP-DOE AMIP-II Reanalysis, URL: http://www.esrl.noaa.gov/psd/data/gridded/data.ncep.reanalysis2.html (accessed 13.06.2016).
26. Geospatial raster data processing. URL: http://rgeo.wikience.org/pdf/slides/rgeo-course-04-raster_processing.pdf (accessed 25.05.2016).
27. H. Wickham, The split-apply-combine strategy for data analysis // Journal of Statistical Software, Vol. 40, P. 1–29, 2011.
28. H. C. Yang, A. Dasdan, R. L. Hsiao, D. S. Parker., Map-reduce-merge: Simplified relational data processing on large clusters, ACM SIGMOD, June 12–14, Beijing, China, 2007.