Scientific big data analytics challenges at large scale

G. Aloisio[a,b], S. Fiore[a,b], Ian Foster[c], D. Williams[d]

[a]*Euro-Mediterranean Center on Climate Change, Italy*
[b]*University of Salento, Italy*
[c]*Computation Institute, University of Chicago and Argonne National Laboratory, Chicago, IL, USA*
[d]*Lawrence Livermore National Laboratory, Livermore, California, USA*

1. Introduction

Data intensive computing will play a major role in exascale scientific discovery [1,2]. Data volume, variety, velocity, and complexity all present challenges that must be faced to efficiently address data analysis at large scale [3,4].

In many domains such as life sciences, climate, and astrophysics, scientific data is often *n-dimensional* and requires tools that support specialized data types and primitives if it is to be properly stored, accessed, analyzed and visualized [5]. The *n-dimensionality* of scientific datasets, and their data cube abstraction, leads to a need for *On-Line Analytical Processing* (OLAP)-like primitives such as slicing, dicing, pivoting, drill-down, and roll-up. These primitives have long been supported in data warehouse systems and used to perform complex data analysis, mining and visualization tasks. Unfortunately, current OLAP systems fail at large scale—different storage models and data management strategies are needed to fully address scalability. Yet the analysis of scientific datasets has a higher computing demand with regard to current OLAP systems, which definitely leads to the need of having parallel/distributed solutions to meet the (near) real-time requirement. Finally OLAP systems are domain agnostic, so they do not provide domain-based support, functions and primitives that are essential to fully address scientific analysis.

Today, scientific data analytics relies on domain-specific software and libraries providing a huge set of operators and functionalities. This approach will fail at the large scale, because most of these software: (i) are desktop based, rely on local computing capabilities and need the data locally; (ii) cannot benefit from available multicore/parallel machines since they are based on sequential codes; (iii) do not provide declarative languages to express scientific data analysis tasks, and (iv) do not provide newer or more scalable storage models to better support the data multidimensionality.

Apart from these *technological* aspects, other concerns more *methodological* like the current scientific workflow should be revisited and additional ones more *social* should be considered.

In the remainder of this work, the most relevant challenges towards scientific big data analytics at exascale are categorized and discussed in Section 2, whereas a related work in this area named Ophidia is presented in Section 3.

2. Main challenges

We review some of the main challenges to be faced at peta-exascale to address scientific data analytics. In particular we discuss the need to identify new *scientific workflows* from a methodological point of view, and highlight *storage and data management challenges* as well as *analytics platforms and parallel/distributed paradigms* for data intensive science. We also discuss the difference between *declarative* and *procedural approaches*, the need to manage *metadata information* to describe data analytics tasks and the relevance of *social challenges*. Finally, we note the differences between *revolutionary* and *evolutionary* approaches.

2.1 Methodological approach: the current scientific workflow

The workflows commonly used for scientific discovery today are based on a *search, locate, download and analyze* steps, typically performed on a researcher's desktop. However, this workflow will not be feasible at peta-exascale. The model will fail for several reasons including: (i) ever-larger scientific datasets, (ii) time- and resource- consuming data downloads, and (iii) increased problem size and complexity requiring bigger computing facilities.

Peta-exascale data requires a different workflow based on data-intensive facilities close to data storage and server-side analysis capabilities. Only the final results of an analysis (e.g., images, maps, reports and summaries typically megabytes or even kilobytes) will need to be downloaded—and even those data may be server-side managed (it could be even stored in a cloud-based environment). Such an approach will reduce (i) the downloaded data, (ii) the makespan for the analysis task, and (iii) the complexity related to the analysis software to be installed on client machines. Moreover, server-side services will spur development of new "client" software (like visualization tools) strongly decoupling the front-end aspects (focusing on presentation and visualization), from back-end ones (targeting more on running the analytics primitives on the scientific datasets). Finally, to address interoperability, a strong emphasis should be devoted to the interfaces provided by the analytics services, which should implement and exploit well-known standards and protocols.

2.2 Storage and data management challenges

New storage models are essential if we are scale large-scale data analysis. New data organizations are needed that better fit the intrinsic *data cube* model of n-dimensional data. Partitioning and distribution could enable parallelism whereas indexing, replication, and data management hierarchies could improve execution efficiency and throughput. In the overall design *data independence* (to address *a clear separation of programs from data* [6]) should be a goal. While in the relational model data independence comes in the form of *physical data independence* and *logical data independence*, in a multidimensional data cube it should also encompass *dimensional data independence*, where the storage model should be independent of the number of dimensions.

As part of the co-design aspect, active storage management research would allow moving the analytics data kernels at the storage layer, drastically reducing I/O transfers via in-storage computations. This research will need a strong involvement of both hardware and software competencies linked with interdisciplinary groups, to efficiently address n-dimensional array data structures on modern storage devices.

2.3 Analytics platforms and parallel/distributed paradigms

As stated before, ad hoc data intensive facilities close to data storage will be needed to address scalable scientific data management. Data analytics primitives are mostly distributive functions, which means they [7] *can be computed for a given dataset by partitioning the data into smaller subsets, computing the operator for each subset, and then merging the results in order to arrive at the measure's value for the original (entire) dataset*. High Performance Computing and commodity-oriented computers will represent two completely different target environments and infrastructures to run data analytics software. In the former case, *tightly coupled* approaches mostly relying on MPI and OpenMP could represent viable solutions on HPC architectures. In the latter case, *loosely coupled* approaches mainly based on MapReduce [8]-like paradigms could help to manage and process large datasets in parallel on shared-nothing architectures.

2.4 Declarative vs. procedural approaches

A server-side paradigm like that described in the *methodological approach* section may use either a procedural or declarative language to express and define data analysis tasks. While a procedural approach may provide the programmer with fine-grained control over implementation, a declarative approach can allow the data analytics system to select implementation strategies. Regardless of approach, a data analysis system may support domain-specific statements to address specific scientific requirements and use cases. A data analytics language should include both a *data definition* and a *data manipulation* language to deal with (i) the definition of data cube structures in the back-end storage system and (ii) the primitives for data cube manipulation and analysis, respectively. A community-driven standardization body could work on the definition of a *scientific data analytics language* to provide a comprehensive reference model. Domain-based extensions could enrich the first level and general-purpose set of statements.

2.5 Metadata management and provenance

Metadata represents a valuable source of information for data discovery and data description. In a data intensive context it will be important to: (i) provide server-side metadata management capabilities, (ii) describe a dataset with provenance metadata information in terms of applied data analytics primitives (to help reproduce analyses and products); (iii) enrich this information with descriptive metadata and links to cross-related digital objects, that could be indexed as well, to improve the data search and discovery process, (iv) build new community-oriented tools to enrich metadata and provide, at the same time, a way to move this process towards much more open, multi-level and collaborative forms.

2.6 Social aspects

Social challenges are becoming more important as scientific efforts in various disciplines move towards large-scale experiments/environments (e.g., the Large Hadron Collider at CERN for High Energy Physics and the Coupled Model Intercomparison Project, phase 5 (CMIP5) [9] for the Climate Science). In this regard, new collaborative environments devoted to ultra-scale data analytics and connected with social networks could change the way scientists interact each other both inside (for research purposes) and outside (for dissemination purposes) the scientific community. Finally, training and academic courses on exascale computing and data intensive science, both from a software and a hardware perspective will be needed to prepare the next generation of scientists and fill the skills gap.

2.7 Revolutionary vs. evolutionary approaches for scientific data analytics

We suggest that in the 2020 timeframe the scientific community should take the chance to exploit *revolutionary* strategies for large-scale scientific data analytics. There are strong domain-based I/O and real-time analysis requirements, new n-dimensional array-based data structures, primitives and parallel/distributed operators to be addressed, and big datasets to be managed, compared and visualized. At the same time, simulations, observations, and reanalyses are producing new, richer and more complex data. *Evolutionary* approaches may require less short-term effort but are unlikely to succeed in all cases and at large scales. On the contrary, *revolutionary* approaches would require a *from-scratch* design and long-term implementation strategy. They may need larger investments (not always!), but would definitely target and focus on large-scale data intensive issues and requirements. They will permit hardware-software co-design strategies to be considered from the beginning, as part of the overall design.

3. The Ophidia Project

A related work in this area is the Ophidia project [10], a research effort on big data analytics facing scientific data analysis challenges in the climate change domain. It provides parallel (server-side) data analysis, an internal storage model and a hierarchical data organization to manage large amount of multidimensional scientific data.

3.1 Ophidia storage model and hierarchical data management

Even though most of the available climate change datasets are in NetCDF, the Ophidia storage model does not rely on the NetCDF file format. More specifically, the internal storage model designed in Ophidia exploits a *relational* approach in which two-column based relational tables are used to store n-dimensional data cubes. A relational table in Ophidia manages a set of (key, value) tuples, where the *key* is a numeric identifier associated to *n-m* dimensions (called "explicit") and the *value* is the binary array intrinsically associated to the remaining *m* dimensions (called "implicit"). To address scalability and to enable parallelism, from a physical point of view, a data cube in Ophidia is horizontally split into several relational tables called fragments that are distributed across multiple I/O servers. Each I/O server is a MySQL RDBMS supporting, at both the data type and primitives levels, the management of *n*-dimensional array data structures. The array-based extensions are provided as a set of User Defined Function (UDF) developed in C.

The entire set of fragments is mapped onto a hierarchical storage implementation composed of four different levels:

- *Level0*: multiple I/O nodes;
- *Level1*: multiple instances of DBMS on the same I/O node;
- *Level2*: multiple instances of physical databases on the same DBMS;
- *Level3*: multiple fragments on the same physical database.

As it can be easily argued, changing these settings can strongly affect performance. For a specific data cube, the higher is the product of the 4 levels, the smaller is the size of each fragment. A high number of fragments is key to enable parallelism. However, for a specific platform configuration (*Level0* x *Level1*) and data cube, increasing too much the number of fragments, can negatively impact on performance as a consequence of the overhead introduced by the higher level of fragmentation and the lower number of elements associated to each fragment.

3.2 Array-based primitives and parallel operators

As mentioned before, the Ophidia framework provides *array-based primitives* as Structured Query Language (SQL) extensions relying on the UDF approach. So far, about 50 primitives have been implemented. Among others, the available array-based functions allow to perform data sub-setting, data aggregation (i.e. max, min, avg), array concatenation, algebraic expressions and predicate evaluation. We note that multiple plugins can be nested to implement a single more complex task (e.g., aggregating by *sum* a subset of the entire array). *Bit*-oriented plugins have also been implemented to manage binary data cubes.

The Ophidia analytics platform provides several MPI-based *parallel operators* to manipulate (as a whole) the entire set of fragments associated to a data cube. Some relevant examples include: (i) data sub-setting (slicing and dicing), (ii) data aggregation, (iii) array-based primitives (the same operator applies to all the implemented UDF extensions), (iv) data cube duplication, (v) data cube pivoting, (vi) NetCDF-import and export.

4. Conclusions

The Ophidia framework is currently being tested at the Euro-Mediterranean Center on Climate Change (CMCC), on CMIP5 data in NetCDF format, Climate and Forecast (CF) convention compliant. Preliminary experimental results are extremely encouraging and demonstrate the scalability of the framework. Moreover, a comprehensive set of domain-based use cases is being defined with climate scientists to provide a scientific validation of the system.

References

[1] J. Dongarra, P. Beckman, et al., The International Exascale Software Project roadmap. International *J. High Performance Computing Apps.* 25, no. 1, 3-60 (2011), ISSN 1094-3420 doi: 10.1177/1094342010391989.

[2] S. Fiore and G. Aloisio, Special section: Data management for eScience. *Future Generation Computer System* 27(3): 290-291 (2011).

[3] J. Chen, A. Choudhary, S. Feldman, B. Hendrickson, C.R. Johnson, R. Mount, V. Sarkar, V. White, D. Williams. "Synergistic Challenges in Data- Intensive Science and Exascale Computing," DOE ASCAC Data Subcommittee Report, Department of Energy Office of Science, March, 2013.

[4] G. Aloisio and S. Fiore, Towards exascale distributed data management, International *J. of High Performance Computing Apps.* 23, no. 4, 398-400 (2009) doi: 10.1177/1094342009347702.

[5] D. Williams, C. Doutriaux, J. Patchett, S. Williams, G. Shipman, R. Miller, C. Steed, H. Krishnan, C. Silva, A. Chaudhary, P. Bremer, D. Pugmire, W. Bethel, H. Childs, M. Prabhat, B. Geveci, A. Bauer, A. Pletzer, J. Poco, T. Ellqvist, E. Santos, G. Potter, B. Smith, T. Maxwell, D. Kindig, D. Koop, "The Ultra-scale Visualization Climate Data Analysis Tools (UV-CDAT): Data Analysis and Visualization for Geoscience Data," *Computer* , vol.PP, no.99, pp.1,1, 0. doi: 10.1109/MC.2013.119.

[6] J. Gray, D. T. Liu, M. Nieto-Santisteban, A. Szalay, D. J. DeWitt, and G. Heber. 2005. Scientific data management in the coming decade. *SIGMOD Rec.* 34, 4 (December 2005), 34-41. DOI=10.1145/1107499.1107503 http://doi.acm.org/10.1145/1107499.1107503

[7] J. Han and M. Kamber, *Data mining: Concepts and Techniques*, Morgan Kaufmann Publishers, 2005.

[8] J. Dean and S. Ghemawat. 2008. MapReduce: simplified data processing on large clusters. *Commun. ACM* 51, 1 (January 2008), 107-113. http://doi.acm.org/10.1145/1327452.1327492.

[9] K.E. Taylor, R. J. Stouffer, and G. A. Meehl 2012 An overview of CMIP5 and the experiment design, Bulletin of the American Meteorological Society 93, no. 4, 485-498 (2012), doi:10.1175/BAMS-D-11-00094.1, http://journals.ametsoc.org/doi/abs/10.1175/BAMS-D-11-00094.1.

[10]S. Fiore, A. D'Anca, C. Palazzo, I. Foster, D. N. Williams, G. Aloisio, "Ophidia: toward big data analytics for eScience", to appear in the Proceedings of the International Conference on Computational Science ICCS 2013, Procedia Elsevier, Barcelona, June 5-7, 2013.