SciDB for High Performance Array-Structured Science Data at NERSC

Yushu Yao,* Benjamin P. Bowen,* Dalya Baron,† Dovi Poznanski,†

*Lawrence Berkeley National Laboratory, Berkeley, CA, USA.
†School of Physics and Astronomy, Tel-Aviv University, Tel Aviv 69978, Israel.

***Abstract*—We discuss some common difficulties when dealing with large amounts of structured data and NERSC's approach to using SciDB as a unified data management, analysis, and sharing framework. We detail two use cases and their basic operations and carry out performance analysis of different workloads and different scales.**

I. Motivation

Current U.S. Department of Energy facilities are facing a data deluge. In addition to simulation results in traditional areas such as climate and nuclear physics, experimental facilities, such as light sources, telescopes, sequencing centers, etc., are producing data in increasingly larger and faster quantities. Some of these facilities are reasonably well prepared for this onslaught of data while others are just beginning to grapple with the challenge. However, none of them are in a position to fully exploit the scientific potential of these data sets. Three major problems researchers on these science projects face are **management**, **analysis** and the ability to **share** the data (across the collaboration and publicly). As the variety and volume of the data grow, a number of challenges emerge:

- Traditional file-system based solutions can no longer efficiently store and access a large number of small files given random I/O patterns. As a result, efficiently filtering large data sets is a difficult task.
- As the analytic algorithms used on big data have become more and more complex, it has become more difficult to implement them in a highly efficient way with respect to parallel I/O and computation.
- Sharing data is no longer feasible in many cases due to the data volume. Web-based portals exist that facilitate sharing, but portals that enable "smart queries" on large data sets are critically needed.

Moreover, with the advance of detector technologies, a small experimental group can generate an extremely large amount of data. The number of groups with access to such large data sets is growing quickly as well. Scientists are burdened with both data management tasks (such as filtering and querying) in addition to complex analytic tasks (such as modeling, machine learning, visualization, etc.). Currently, many emerging projects are developing their data management and analysis workflows, which often produce sub-optimal results as the wheel is reinvented repeatedly and poorly.

A unified, scalable, easy to use, extendable system is needed to tackle these extreme data challenges. This paper will describe such a solution that is being implemented at NERSC, the National Energy Research Scientific Computing Center. NERSC is especially well suited for a new extreme data solution because of critical existing infrastructure such as the NERSC Global Filesystem [1] and because of the large number of data-intensive projects at NERSC. Thus, enabling extreme-scale data intensive computing via deployment of pre-exascale hardware and software and rich services with big data capabilities is a major NERSC strategic thrust.

II. SciDB as a Solution for Array-Structured Data

Scientific data contains two major parts: structured data (e.g., an image), and unstructured metadata (e.g., parameters of the detector when image was taken). Some solutions (such as MongoDB) are available to manage and query unstructured metadata at scale, where the metadata is less well organized but smaller in size. The key to scientific data analysis is to efficiently query, filter, aggregate or perform more complicated operations on the structured data. Solutions providing these capabilities must also be able to take advantage of the hardware at current and future High-Performance Computing (HPC) facilities. While not all structured data has the same form, many experimental facilities are based on detectors that generate array-based structured data (CCD imagers). Many analysis techniques also naturally fit into arrays (such as comparative genomic hybridization). In this project, we evaluate SciDB as a unified, scalable and easy-to-use solution to manage, analyze and share array-based structured scientific data at extreme scale.

SciDB [2] is an open-source database system designed to store and analyze extremely large array-structured data. Some examples of array-structured data include:

- Imaging data: digital pictures from light sources or telescopes
- Time series data collected from sensors
- Spectral data produced by spectrometers and spectrographs
- Graph-like structures that represent relations between entities (sparse matrix)

The benefits of SciDB include scaling on hundreds of nodes, ease of deployment on commodity hardware or large DOE HPC systems, efficient parallel I/O, a large variety of built-in generic analysis tools, and ease of integrating new analytical

Digital Object Indentifier 10.1109/MCSE.2015.43 1070-9924/$26.00 2015 IEEE

algorithms that can transparently access efficient I/O and parallelization.

A. Features of SciDB

Details of SciDB are presented in [2]. A comparison of SciDB with other relational databases is detailed in [3]. The following features of SciDB are most attractive in solving the science problems in which we are interested:

- **Array-like Data Structured** : Most scientific data has natural characteristics such as spacial and temporal values. This is a natural fit to the array-like data structure of SciDB. For example, in climate simulation output, the array dimensions can be time, longitude and latitude, while each cell of the array contains a tuple of values such as temperature, humidity, wind speed, and other computed fields.
- **Abstracted Parallelism and I/O**: SciDB uses high-level query languages to define operations on the arrays. Operations may include sub-array, filter, slice, aggregate, and more. When used collectively, these few operations can provide the majority of functionalities most typical science project need. Using the high-level abstraction layer also hides the parallelism and I/O from the algorithm designer. This enables expanding the infrastructure when the data size grows: the same set of queries will work on the same arrays even if the size of the SciDB cluster grows.
- **Python and R Binding**: A significant amount of software has been developed in R and Python to solve a large variety of science problems. Python is a popular language in physics and astronomy for data analytics, and R is popular in life sciences. SciDB provides language bindings for both Python and R. In addition to allowing SciDB queries inside Python and R, SciDB can be used as the storage back-end for a Python NumPy array, and it is compatible with most of the operations on NumPy arrays. This allows most of the existing software to work out-of-the-box after adding the SciDB backend.
- **Extensibility**: For complex operations that are hard to express with "canned" SciDB functionalities, SciDB provides a way to create new types, functions or operators. The extensions are implemented in the C++ language. It is very easy to create user-defined types and functions; however, some engineering effort is necessary in creating new operators.

III. SciDB Testbed Project at NERSC

In 2013 we started the SciDB testbed project with a 20-node SciDB cluster. Our approach was to create partnerships in which we paired one to two domain scientists with NERSC staff to work on real data problems. To date, NERSC has initiated more than ten such partnerships across a broad range of science topics, from astrophysics to biology. In each partnership, we helped the scientists port their data onto the SciDB testbed at NERSC, and provided help with initial data analysis. Our aim was to help the science projects build SciDB into their normal science workflows. The assumption was that lessons learned from each case study would provide insight into how to create new technologies and environments for other data intensive computing projects at NERSC. For each science partnership, we focused on the three aspects of science workflow and tried to answer the following questions:

Data Management: Can SciDB be used as an efficient way to store and organize terabytes of raw and derived data? Can SciDB help to efficiently filter the data in parallel or find the "interesting" portion of the data?

Data Analysis: Can the project implement complex modeling algorithms in SciDB and take advantage of the automatic parallelism? How much analysis can be done with the built-in SciDB functionalities and how much needs custom SciDB extensions? How difficult is it to extend SciDB?

Data Sharing: Can SciDB efficiently allow multiple users to share the same dataset and perform different analysis? Can SciDB be integrated with a science gateway (web portal) and allow collaborators to submit smart queries?

Usability is at the core of all three aspects. The question of how easily a non-expert user could become proficient with SciDB was of the utmost importance. In the following sections, we present two use cases where SciDB played a revolutionary role in the data analysis workflow.

IV. Metabolite Atlases

A. The Science Story

Liquid chromatography electrospray ionization mass spectrometry (LC/MS)-based metabolomics is a technique used to profile the small-molecule composition of a biological sample containing intermediate and final products of metabolism [4]. This is a widely used method to study a variety of biological systems in many different science areas such as biofuels, clinical research, and other areas.

The challenge for LC/MS metabolomics is in the complexity of data. Modern LC/MS equipment can produce multiple gigabytes of data every day. The data often include many experimental artifacts. In addition, the huge chemical diversity of metabolites makes interpretation of the data extremely complicated.

"Metabolite Atlases" is a DOE-funded project to provide a platform to efficiently interpret LC/MS data through automatic data reduction and feature finding. The project aims to host multi-terabytes of LC/MS experimental data and enable the sharing of identification methods across different experiments.

B. The Data Structure

Below is a simplified version of the data structure for Metabolite Atlases. In each LC/MS experiment, a scientist prepares a set of samples and takes multiple "runs" of data. Each "run" consists of loading a sample, configuring the LC and MS operating parameters, and scheduling the injection of the sample along with the acquisition of data. In each run, the molecules in the sample are released little by little in a controlled way. At each time step (which we call retention time), some molecules are released, and a mass spectrum is

Digital Object Indentifier 10.1109/MCSE.2015.43 1070-9924/$26.00 2015 IEEE

recorded. An intensity reading is recorded for each each value of the mass divided by the charge of the molecule (M/Z). In the end, each run produces a 2D spectrum of Retention Time vs. M/Z. There are normally thousands of different retention time steps during a run, and each mass spectrum can contain millions of readings. After compression, each run will produce a file up to several hundred megabytes.

Considering multiple runs, we can use a 3-D array to represent the data structure in SciDB, with Retention time, M/Z and Run Number as the three dimensions. As the data are collected, new runs are assigned higher run number values, and the array will keep growing along that dimension. Each run is also associated with a quantity of metadata, such as experiment type and metabolites identified. Due to the unstructured nature of the metadata, we used a MongoDB database to store the metadata for each run. On top of MongoDB and SciDB, we built a set of RESTful queries [5] to encapsulate different operations and authentication/authorization. Then we built a web service on top of the RESTful queries to provide an easy-to-use interface for users.

C. The Operations

One of the most import use cases in LC/MS is where the user selects one run and wants to return different kinds of plots about that run. This corresponds to two types of operations in SciDB: subselection and aggregation. With a given "run number," the "slice" can be obtained with a simple subselection operation in SciDB. Since SciDB uses hash-table-like techniques, locating values along a dimension is close to constant time complexity much faster than indexing technologies in traditional databases (that go as O(logN)). Figure 1 illustrates selecting one slice of the data cube and performing aggregate operations on the slice. One benefit of SciDB is the ease of use. We are showing some of the queries for the operations we discuss. As an example, the query below will return a "slice" (sub-array) from the INPUT array with run number n:

SUBSELECTION="between(INPUT, n,null,null,n,null,null)"

Two types of plots about a run are most commonly used: the Total Ion Current Intensity Plot (TIC) and the RT/MZ Heat Map (See Figure 1). TIC is a plot showing the total intensity for each retention time step. That is, for each retention time, sum all available intensity values (Figure 2 left). This SciDB query looks like:

aggregate(SUBSELECTION,sum(Intensity),Retention Time)

Note that as a functional language, we can use the output of one query (SUBSELECTION) as an input to another query:

regrid(SUBSELECTION,M/Z Grid Size, Retention Time Grid Size, sum(Intensity))

Heat map is another commonly used plot. It divides the 2-D slice into coarser grids and returns the sum of the available intensity values for each grid (see Figure 2, right). The query reads:

regrid(SUBSELECTION,M/Z Grid Size, Retention Time Grid Size, sum(Intensity))

D. Discussion/Findings

As more experimental data are collected, we expect the size of the array to grow. It is important that 1) time to retrieve data for one run stays constant regardless of the total size of the data, and 2) time to retrieve values for all runs grows linearly with the data size. A benchmark test of this would consist of loading portions of the test data set (total 150GB) and performing the above two operations on arrays with different sizes. We performed the tests on a 16-node SciDB cluster (with one instance per node). As shown in Figure 3, the retrieving time for a single run stays constant as the data size grows, and the retrieving time for all runs grows linearly with data size.

Being a distributed system, SciDB distributes both the data and calculations to a number of compute nodes. So it is also important that the same problem will run faster as the size of the available compute resource grows. To benchmark this, we set up several SciDB clusters with different sizes, loaded the same amount of data (25% of the full test set), and performed the same two operations. The results are shown in Figure 4. For the "retrieve all" operation, strong scaling behavior is observed; i.e., the time to perform the same operation is inversely proportional to the total number of SciDB instances. As for the "retrieve one" operation, since most of the query time is spent in the database overhead, the scaling is less obvious.

This use case shows that SciDB is a good solution for hosting a growing amount of data and performing fast queries as the data size grows.

V. Studying the Milky Way with Massive Spectroscopic Samples

A. The Science Story

Diffuse interstellar bands (DIBs) are a long-standing mystery in astronomical spectroscopy - absorption features found in the spectra of stars, with an unknown origin [6]. The first DIBs were discovered in 1919 [7], but by now, hundreds have been discovered [8] [9]. It is clear that DIBs originate from the diffuse gas that permeates the Galaxy - the interstellar medium (ISM) [10]. It is also likely that they are created by some molecules. But identifying these molecules remains an unanswered question [11].

An important method for discovering which species are the carriers of DIBs involves correlating their properties with other known constituents of the ISM (See [6], [11] for reviews). Most studies so far have been based on just a few sightlines or a few absorption lines (However, see [12], [13] , [14]). However, with modern data and adequate analysis methods, one can study hundreds of lines in millions of spectra.

Our study uses spectra from the Sloan Digital Sky Survey (SDSS) [15], which has created the most detailed three-dimensional maps of the Universe made to date, with 1.5 million spectra spread over about a quarter of the sky. The signal-to-noise ratio (S/N) of a single spectrum is not sufficient for DIB detection and therefore the spectra must be stacked in large numbers (typically more than 3,000) to reach a

Digital Object Indentifier 10.1109/MCSE.2015.43 1070-9924/$26.00 2015 IEEE

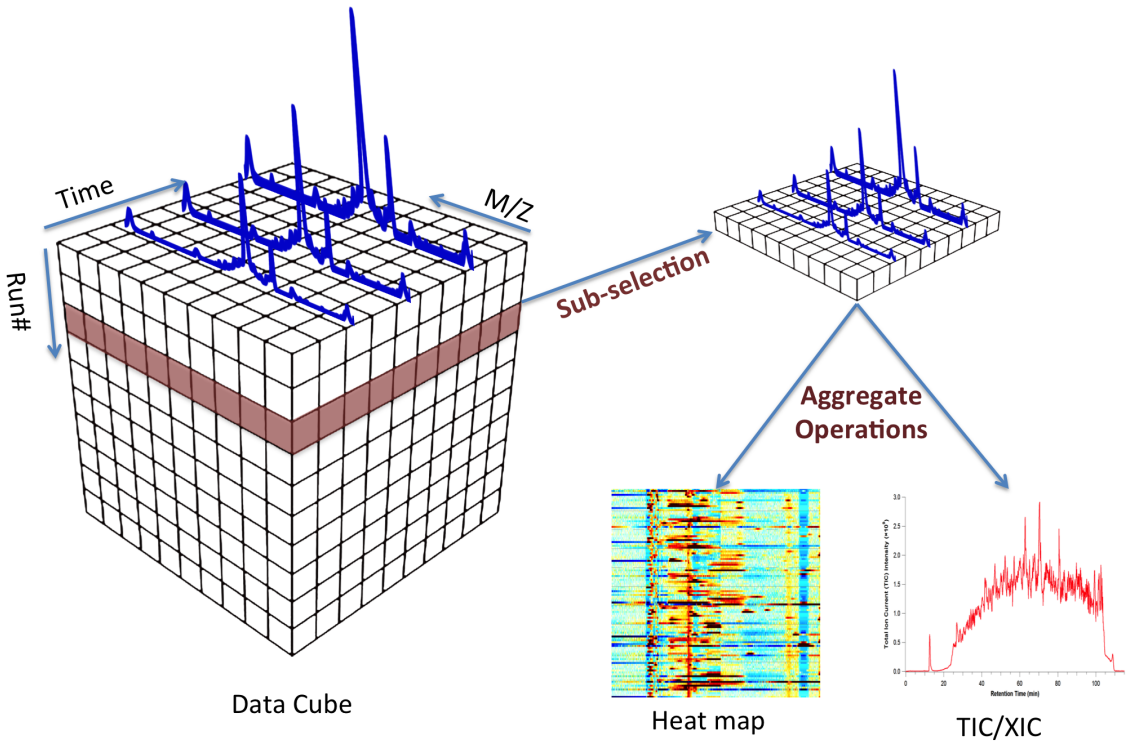

Fig. 1. Simplified workflow for Metabolite Atlas.

sufficiently sensitive measurement. To examine DIBs strengths (DIBs were taken from the catalogs: [16], [8] and [9]) in different ISM conditions we stack the spectra after binning them by various parameters. Since this is an exploratory process, it has to be repeated many times along various scenarios.

B. The Data Structure

The science team used 1.5 million spectra, each spectrum containing 11,000 grid elements and more than 40 metadata fields. They modeled the data as a two-dimensional array, each cell of which contained the flux in given spectrum at a given wavelength. Additional important attributes include smoothed versions of the spectra and the ratio between the spectra and their smoothed versions, showing only high-frequency features that are relevant to this study. Additional metadata fields were spread over several tables, cross-joined via a unique spectrum identifier.

With existing data sets, this method can trivially scale to two million objects, and soon, surveys (such as the Dark Energy Spectroscopic Instrument; DESI) will generate an increase by an order of magnitude.

C. The Operations

The science team subdivided the data based on the values or different metadata fields or random sampling, typically producing groups with tens of thousands of spectra. They tested different methods for the stacking process. While averages are computationally cheap, they are too sensitive to extreme outliers that often exist in physical sciences. Sigma clipping could, in principle, solve this issue, but we found that it was still less robust than calculating medians, the method ultimately used.

As illustrated in Figure**??**, the operation can be divided into three steps. In the first step, in the metadata array, the ”filter” operator finds all spectra ID’s based on some criterion or criteria. In the second step, a ”cross_join” operator selects the needed spectra out of the data array. In the third step, an aggregate operator calculates the median value at each wavelength. In the end, a one-dimensional array is returned with Wavelength as the dimension.

The whole process can be coded with one simple SciDB query below:

aggregate(cross_join(DataArray, filter(MetaArray, MetaData1=100), Spectrum ID), median(Intensity), Wavelength)

Another benefit of SciDB is that the coder doesn’t need to know the details about the SciDB setup (e.g., the number of nodes over which the database arrays are spread). All parallel I/O and parallel computation are handled automatically and transparently. One can run the same query on SciDB clusters with different sizes.

Digital Object Indentifier 10.1109/MCSE.2015.43 1070-9924/$26.00 2015 IEEE

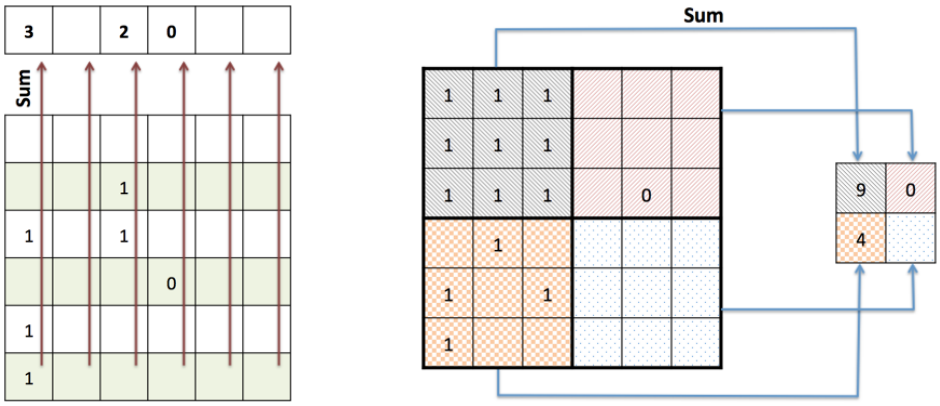

Fig. 2. Aggregate operation calculating the sum along one dimension (left); Regrid operation calculating the sum for each 3x3 grid.

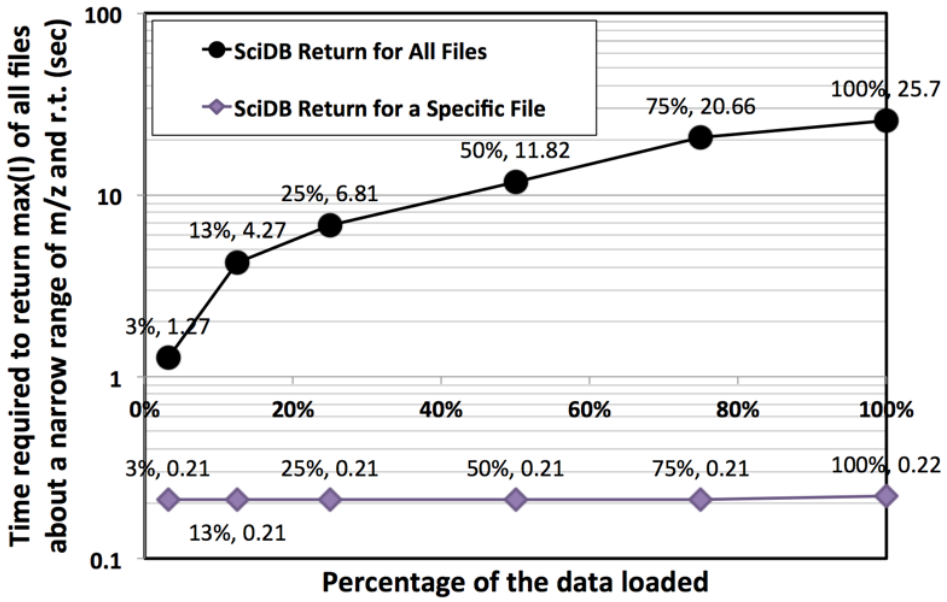

Fig. 3. Time to obtain a fixed amount of data from 1) a single run and 2) all runs, at different total data size.

D. Discussion/Findings

There are two classical alternatives to using SciDB for this project. One would be to read the relevant data from disk whenever needed, and construct the stacked spectra. This is severely I/O limited, and can take several hours for a single iteration. Another would be to hold the entire data in RAM. While this would be faster, it is possible only with a dedicated machine with of order 1 TB of RAM. This solution is expensive and does not scale to larger jobs.

To test the scaling of SciDB we loaded the same data into five different SciDB clusters from four instances to 64 instances. Then we performed the two operations above on each SciDB cluster, selecting 50,000 and 300,000 spectra, respectively, and calculated the median. As illustrated in Figure 6, strong scaling behavior is observed for both operations. It is important to note that increasing the number of spectra from 50K to 300K only uses 30% more time regardless of the cluster size.

Using SciDB, we binned the data using cross-join operations with the metadata tables and managed to compute median spectra in 1 to 3 minutes (combining 50,000 ? 300,000 spectra, respectively) per set of spectra (we call it bins). For 25 bins, the whole procedure takes about 30 minutes. This is fast enough to not delay the research process.

This project has already culminated in a submitted publication (Baron et al. 2014a), another in final editing stages (Baron et al. 2014b), and a few more planned.

VI. Conclusions and Future Work

The pilot projects start to help us answer the questions we posted earlier in this paper.

Data Management: SciDB can efficiently store and index terabytes of raw and derived data., providing very efficient ways to filter and subselect the “interesting” portion of the

Digital Object Indentifier 10.1109/MCSE.2015.43 1070-9924/$26.00 2015 IEEE

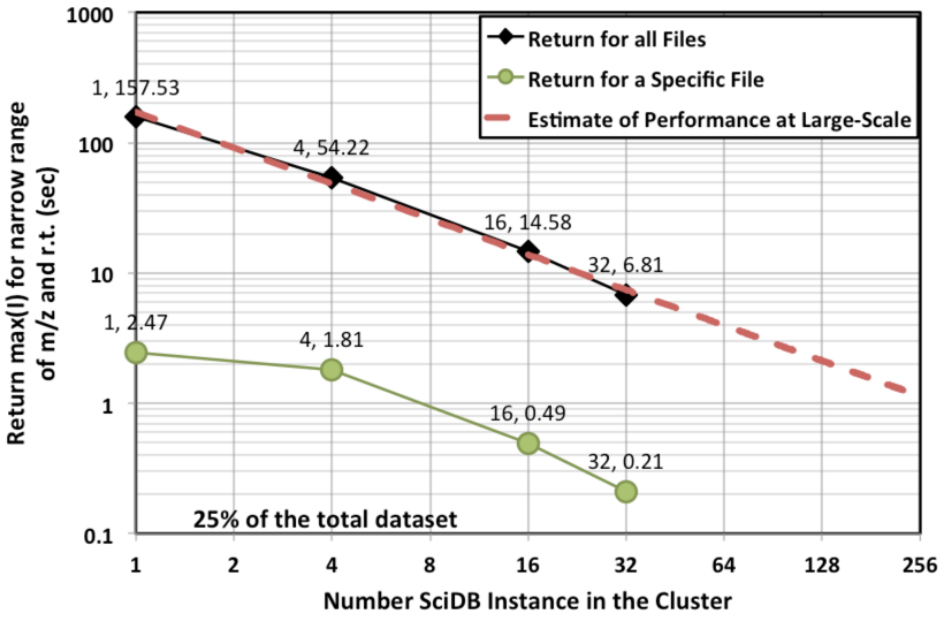

Fig. 4. Returning a fixed amount of data, from 1) a single run and 2) all runs, with different SciDB cluster size.

data. It is most efficient in subselecting based on the dimensions. It is also relatively easy to "redimension" an existing array to create new dimensions for efficient subselection. One drawback of SciDB is that one needs to import the raw data into SciDB, and SciDB's data format is not easily readable from other analytic tools. For projects with a large amount of existing data in HDF5 or NetCDF format, to get the benefits of SciDB one needs to load the data into SciDB; therefore, more than one copy of the data is stored.

Data Analysis: SciDB provides a powerful set of primitive operations on the arrays. From our experience, most of the operations can be implemented with the provided primitives. Also, most of the primitive operations are scalable. Therefore, the parallelism (both the algorithm and I/O) is hidden from the user. Extending SciDB to add user defined types and functions is easy. Creating a highly optimized new operator is less easy, but a well defined interface exists and we provided some examples. In some cases, the analytics require partitioning the array and processing each partition in an external application (i.e., map-reduce like operations with a black-box executable). Currently, SciDB does not provide a mechanism for this kind of operation.

Data Sharing: Projects such as the Metabolite Atlas are building a science gateway and RESTful API to allow collaborators to submit smart queries to SciDB. SciDB does not provide user control. The SHIM interface (with Python and R) supports primitive user authentication, but the "iquery" interface does not. Therefoe, additional firewall configurations are necessary to secure SciDB from unauthorized queries, and the access control for the science gateway application needs to be handled inside the web application.

Usability: We observed that it was very easy for beginners with limited computing knowledge to start using SciDB and writing queries. Most of the pilot projects start with a graduate student in a science field (not computer science), and significant progress can be made within weeks.

After several pilot projects we confirmed that SciDB is a scalable, yet easy to use data analytic framework, ideal for handling mid-to-large amounts of array based data. It can help solve the management, analysis and sharing aspects of the data problem for a number of science areas. As many of the pilot projects move from the testing stage to the production stage, requirements for reliability and support also increase. To meet this demand, NERSC is moving forward to provide a production SciDB service for its users.

Until now, all SciDB services have been provided using commodity hardware in a Linux environment. For large computing centers such as NERSC, the most efficient way to provide a SciDB service is to use hardware provided by the workhorse computing systems, which at NERSC, are Cray systems, such as the Hopper Cray XE6. In addition, as processor technology evolves, massively parallel architecture may prevail as the primary architecture in most supercomputing systems. Many operations within SciDB (such as matrix calculations) can benefit from parallelism. To address these problems, we are currently working on two exploratory projects. One project involves finding efficient ways to run large SciDB clusters on Edison, the NERSC Cray XC30 system with a peak performance of 2.57 petaflops/sec, 133,824 compute cores, and 357 terabytes of memory. Another project involves using Intel Xeon Phi systems for acceleration of computations in SciDB. We also aim to run SciDB at large scale on future NERSC systems such as the "Cori" Intel Knights Landing-based supercomputer that also includes novel features to

Digital Object Indentifier 10.1109/MCSE.2015.43 1070-9924/$26.00 2015 IEEE

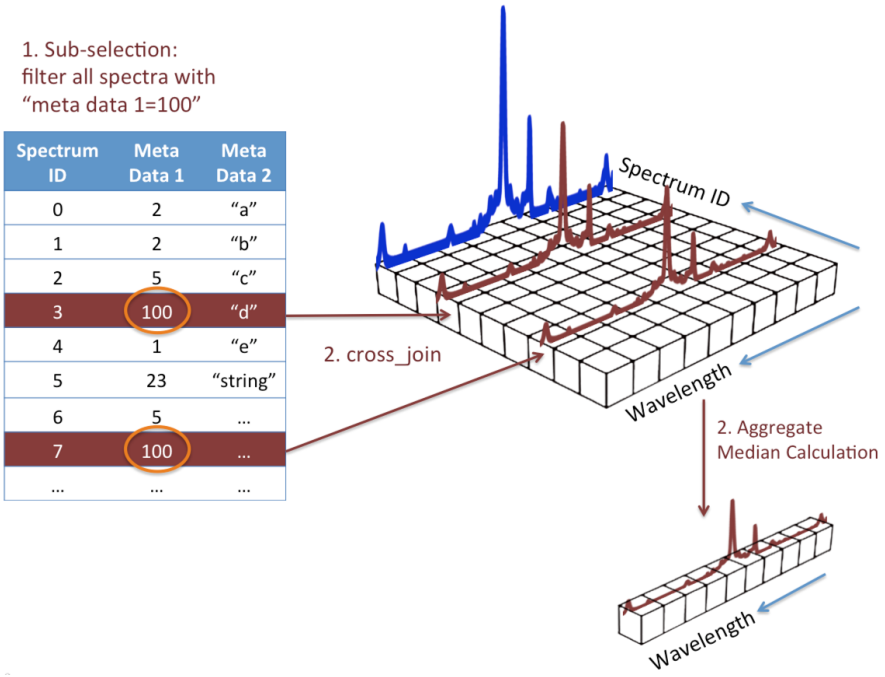

Fig. 5. Simplified workflow for DustOff.

vastly improve I/O performance. These efforts will go well along the way to providing single supercomputer systems supporting both usable exascale computing and extreme scale data processing to improve science discovery and impact.

References

[1] Nersc global file systems. [Online]. Available: https://www.nersc.gov/users/data-and-file-systems/file-systems/

[2] P. G. Brown, "Overview of scidb: Large scale array storage, processing and analysis," in *Proceedings of the 2010 ACM SIGMOD International Conference on Management of Data*, ser. SIGMOD '10. New York, NY, USA: ACM, 2010, pp. 963–968. [Online]. Available: http://doi.acm.org/10.1145/1807167.1807271

[3] D. Malon, P. van Gemmeren, and J. Weinstein, "An exploration of scidb in the context of emerging technologies for data stores in particle physics and cosmology," *Journal of Physics: Conference Series*, vol. 368, no. 1, p. 012021, 2012. [Online]. Available: http://stacks.iop.org/1742-6596/368/i=1/a=012021

[4] B. P. Bowen and T. R. Northen, "Dealing with the unknown: Metabolomics and metabolite atlases," *Journal of the American Society for Mass Spectrometry*, vol. 21, no. 9, pp. 1471 – 1476, 2010. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1044030510002692

[5] O. F. F. Filho and M. A. G. V. Ferreira, "Semantic Web Services: A RESTful Approach," in *IADIS International Conference WWWInternet 2009*. IADIS, 2009, pp. 169–180. [Online]. Available: http://fullsemanticweb.com/paper/ICWI.pdf

[6] G. H. Herbig, "The Diffuse Interstellar Bands," *Annual Review of Astronomy and Astrophysics*, vol. 33, pp. 19–74, 1995.

[7] M. L. Heger, "Further study of the sodium lines in class B stars ," *Lick Observatory Bulletin*, vol. 10, pp. 141–145, 1922.

[8] L. M. Hobbs, D. G. York, T. P. Snow, T. Oka, J. A. Thorburn, M. Bishof, S. D. Friedman, B. J. McCall, B. Rachford, P. Sonnentrucker, and D. E. Welty, "A Catalog of Diffuse Interstellar Bands in the Spectrum of HD 204827," *The Astrophysical Journal*, vol. 680, pp. 1256–1270, Jun. 2008.

[9] L. M. Hobbs, D. G. York, J. A. Thorburn, T. P. Snow, M. Bishof, S. D. Friedman, B. J. McCall, T. Oka, B. Rachford, P. Sonnentrucker, and D. E. Welty, "Studies of the Diffuse Interstellar Bands. III. HD 183143," *The Astrophysical Journal*, vol. 705, pp. 32–45, Nov. 2009.

[10] P. W. Merrill, "No. 536. Stationary lines in the spectrum of the binary star Boss 6142." *Contributions from the Mount Wilson Observatory / Carnegie Institution of Washington*, vol. 536, pp. 1–3, 1936.

[11] P. J. Sarre, "The diffuse interstellar bands: A major problem in astronomical spectroscopy," *Journal of Molecular Spectroscopy*, vol. 238, pp. 1–10, Jul. 2006.

[12] J. Kos, T. Zwitter, E. K. Grebel, O. Bienayme, J. Binney, J. Bland-Hawthorn, K. C. Freeman, B. K. Gibson, G. Gilmore, G. Kordopatis, J. F. Navarro, Q. Parker, W. A. Reid, G. Seabroke, A. Siebert, A. Siviero, M. Steinmetz, F. Watson, and R. F. G. Wyse, "Diffuse Interstellar Band at 8620 Å in RAVE: A New Method for Detecting the Diffuse Interstellar Band in Spectra of Cool Stars," *The Astrophysical Journal*, vol. 778, p. 86, Dec. 2013.

[13] T. Lan, B. Ménard, and G. Zhu, "Dibs in sdss," *ArXiv:1406.XXX*, 2014.

[14] J. Kos and T. Zwitter, "Properties of Diffuse Interstellar Bands at Different Physical Conditions of the Interstellar Medium," *The Astrophysical Journal*, vol. 774, p. 72, Sep. 2013.

[15] Sloan digital sky survey. [Online]. Available: http://www.sdss.org

Digital Object Indentifier 10.1109/MCSE.2015.43 1070-9924/$26.00 2015 IEEE

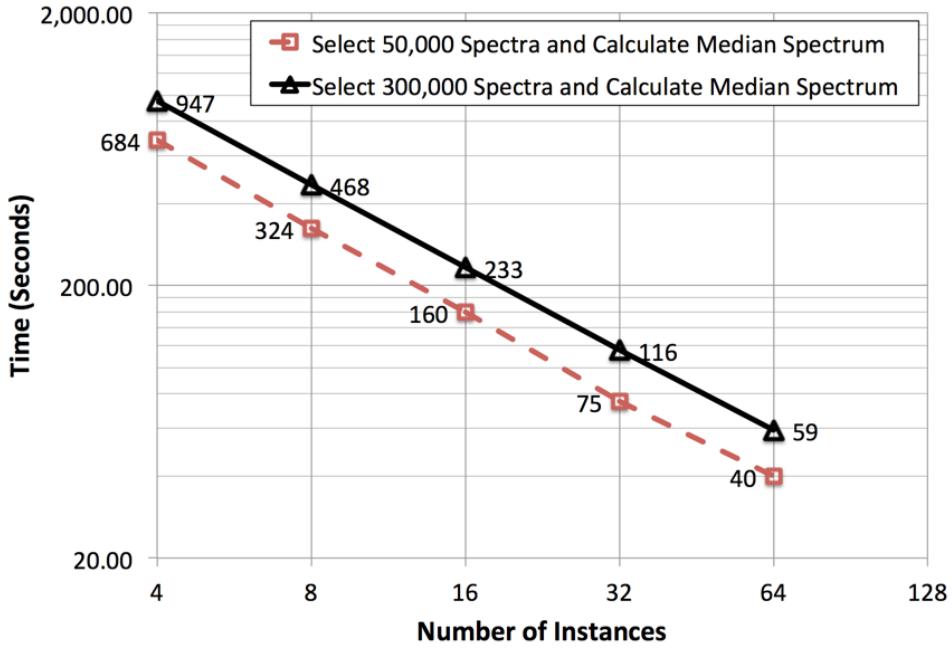

Fig. 6. Time to obtain a fixed amount of data from 1) a single run and 2) all runs, at different total data size.

[16] P. Jenniskens and F.-X. Desert, "A survey of diffuse interstellar bands (3800-8680 A)," *Astronomy and Astrophysics, Supplement*, vol. 106, pp. 39–78, Jul. 1994.

Digital Object Indentifier 10.1109/MCSE.2015.43 1070-9924/$26.00 2015 IEEE