Accelerating Scientific Analysis with SciDB

To cite this article: L. Gerhardt et al 2015 J. Phys.: Conf. Ser. 664 072019

View the article online for updates and enhancements.

Related content
- Comparing NetCDF and SciDB on managing and querying 5D hydrologic dataset
  Haicheng Liu and Xiao Xiao
- An exploration of SciDB in the context of emerging technologies for data stores in particle physics and cosmology
  D Malon, P van Gemmeren and J Weinstein
- Emerging Database Technologies and Their Applicability to High Energy Physics: A First Look at SciDB
  D Malon, J Cranshaw, P van Gemmeren et al.
Accelerating Scientific Analysis with SciDB

L. Gerhardt, C.H. Faham, and Y. Yao
Lawrence Berkeley National Laboratory
1 Cyclotron Road Mailstop 943R0256
Berkeley, CA 94720 USA

E-mail: lgerhardt@lbl.gov

Abstract. SciDB is an open-source analytical database for scalable complex analytics on very large array or multi-structured data from a variety of sources, programmable from Python and R. It runs on HPC, commodity hardware grids, or in a cloud and can manage and analyze terabytes of array-structured data and do complex analytics in-database. We present an overall description of the SciDB framework and describe its implementation at NERSC at Lawrence Berkeley National Laboratory. A case study using SciDB to analyze data from the LUX dark matter detector is described and future plans for a large SciDB array at NERSC are described.

1. Introduction
Scientists are facing an ever-escalating amount of data. With new instruments coming online that will deliver in excess of 1 PB / day of data, the need for fast analysis of large volumes of data is stronger than ever. The vast majority of this data is in the form of arrays. Common examples of this include time series data from sensors, image data from light sources with an intensity value for each x, y, and z position over time, or climate data with various values for each latitude and longitude coordinate. This data is filtered, aggregated, rebinned, and analyzed. Detectors are often multi-purpose instruments, and their data sets are subject to many different filtering and analysis requirements, resulting in data that is written once and read and analyzed many times.

SciDB is an open-source analytical database that is built for exactly these kinds of analyses on large sets of array data. It runs on commodity hardware and offers a flexible and scalable analytical platform. In this paper we will give a brief description of how SciDB works, present a case study using data from the LUX dark matter detector, and discuss future plans for implementing SciDB at NERSC.

2. SciDB
SciDB is an open-source array database management system and analytical platform, created by Paradigm4. SciDB uses a shared-nothing architecture that can be scaled out to hundreds to thousands of processors. A single managing instance, called the coordinator, serves as the interface to the worker nodes. Communication occurs between coordinator and worker instances and between the worker instances. Each worker node stores a portion of the data in a separate repository that can be located on local disk or on a shared file system. Data is handled in units called “chunks”; we found the optimal chunk size to be around 1-2 million entries. Data in chunks can also be overlapped, for optimizing nearest-neighbour type calculations. Chunks are apportioned out to the worker instances, and the coordinator runs a small PostgreSQL database that tracks part of the metadata of the cluster and the
arrays. When queries are run, data is scanned from all instances and streamed into and out of each
operator one chunk at a time. If a chunk is repeatedly accessed, it is kept in memory for faster access.

Each SciDB array is made up of two classes of data: attributes and dimensions. Attributes are the
data that is stored in the array; there can be multiple attributes in an array. Dimensions are also data,
but they are also used as index values for each cell. SciDB functions act most efficiently on
dimensions, so arrays should be structured with commonly used selection data as a dimension. Array
creation is relatively easy, and reshaping arrays to optimize analysis is fairly common.

SciDB accepts queries in various ways. There are two SciDB query languages, Array Query
Language (AQL) and Array Functional Language (AFL). AQL is a high-level declarative language
that is similar to the SQL language for relational databases, while AFL is a slightly friendlier language
used to compose queries or statements. SciDB also supports the shim interface, with robust support for
python and R interfaces.

This SciDB AQL command creates an array named “zoo”:

```sql
create array ZOO <monkey:int64 zebra:int64 elk:int64> 
[cagenum=0:100,10,0, distemper=0:*,2,0]
```

Monkey, zebra, and elk are attributes, while cagenum and distemper are the dimensions.
Dimensions can be integers or floats, and support for string dimensions is forthcoming. The range of
the dimensions can either be declared at array creation (as for cagenum) or left unbounded
(distemper). The range of each dimension stored in a chunk as well as the overlap is also declared in
the array creation line. This mechanism is somewhat awkward, since the optimal chunk size can vary
greatly depending on the sparseness of your data. Functionality to automatically calculate an optimal
chunk size would help analyses greatly.

This SciDB AFL query selects entries in cages 1 to 20 that contain zebras:

```sql
filter (between (ZOO, 1, null, 20, null), zebra=1))
```

and the same query in R:

```r
subset (ZOO[1:20], “zebra == 1”)
```

Nested commands are lazily evaluated. For a given query, the coordinator node divides the
execution plan into fragments and sends them to each worker instance. When possible the coordinator
node will send a complete plan to worker nodes, but for more complex queries it uses adaptive
methods where sub-plans are sent to each node and evaluated in parallel. These sub-plans can be
thought of as new, temporary arrays. Arrays are not overwritten, so previous versions can be recovered
if necessary. Additionally, SciDB keeps a record of each command that is performed on a given array.

SciDB scales well. Tests indicate that the time to retrieve a subset of data stays constant regardless
of the total size of the data and that the time to retrieve all data grows linearly with data size.
Additionally the response time of SciDB shows strong scaling as the number of worker instances
increase [1].

SciDB has been used for analysis of climate data, bioinformatics, astronomy, and high energy
physics.

3. High Energy Physics Case Study
To evaluate the usefulness of SciDB for high energy physics analyses, we used a test instance at the
NERSC LBNL facility. The SciDB instance had 32 workers spread over 16 Intel Xeon X5550 (2.67
GHz) nodes, each with 8 cores and 24 GB of RAM. This SciDB array was loaded with data from the
2013 inaugural run of the LUX data matter detector.

LUX is a 370 kg liquid xenon time-projection chamber built to directly detect faint interactions
from galactic dark matter in the form of Weakly Interacting Massive Particles (WIMPs). A schematic
of the LUX detector is shown in Fig. 1. The LUX detector utilizes top and bottom arrays of 61 photomultiplier tubes (PMTs) each to detect particle interactions in the liquid xenon volume. An interaction produces two signals: scintillation photons and ionization electrons. The scintillation photons, called the S1 signal, are detected immediately by PMTs, while the ionization electrons are drifted upwards with an electric field into a gas phase, where they undergo electroluminescence and produce secondary photons called the S2 signal. The pair of S1 and S2 signals represents a single interaction in the xenon volume and is referred to as an “event”. Because the electrons move at a constant velocity, the time between the S1 and S2 signals indicate the depth (z-dimension) of the interaction vertex. Position reconstruction algorithms on the top-array hit pattern of the S2 signal allows for x-y localization of the interaction vertex. The majority of the background in the LUX experiment is composed of interactions with the xenon electrons (“electronic recoils”), while WIMPs are expected to interact with the xenon nucleus (“nuclear recoils”). Only nuclear recoil particle interactions in the center of the detector, called the fiducial volume (outlined with a red box in Fig. 3), are considered candidate WIMP events. For a complete technical description of the LUX detector, refer to [2].

Dark matter has never been detected directly, and dark matter signals are expected to be very weak (a few keV of energy deposited) and infrequent (a few interactions per year-tonne of exposure), resulting in a data set that is strongly background dominated. In the 2013 initial data run, which set the most stringent limits on WIMP dark matter interaction rates [3], LUX collected 83 million events containing more than 600 million pulses and wrote 32 TB of raw data. Of this data, only 160 events survived basic analysis cuts for potential WIMP candidates. All of these events were consistent with the background-only hypothesis. The data rate for the new dark matter run starting in 2014 is expected to exceed 250 TB/year.

The output of the LUX data processing framework reduces the raw PMT waveforms to a set of attributes, such as timing, number of photons, pulse shape, and several other characteristics. This set of reduced-feature data comprising about 10 TB of disk space was uploaded into the SciDB instance. Two arrays were created: EventArray and PulseArray. The EventArray contains information about every event in the detector and contains a unique timestamp identifier for cross-reference with the PulseArray. The PulseArray contains information about the pulses that comprise each event. Both arrays were structured to place variables commonly used for selection as array dimensions. For example, for PulseArray, these included the time of the event (in ns), pulse type (S1, S2, etc.), and pulse number (there are multiple pulses in an event). In addition to this, there were more than 50 other pulse variable stored as array attributes, such as pulse area and other pulse shape characterization quantities. Using these arrays, two representative LUX analyses were reproduced in SciDB: detector stability and WIMP signal search.
3.1. Detector Stability

Being able to characterize the stability of the detector is essential to LUX analyses. The size (number of detected photons) of an S2 pulse created by a single electron in the detector is an important calibration constant, as it allows for an observed S2 signal to be converted back to a number of electrons generated in a particle interaction (which in turn is proportional to the energy of the interaction). This single electron size is expected to be stable over time, with only observable statistical fluctuations, as long as detector conditions remain unchanged. Large deviations in electron pulse size would indicate changes in the electron extraction field or electroluminescence yield, and data taken during such excursions from baseline values would warrant additional scrutiny and potential exclusion from the WIMP search. Single electrons are tagged in the data stream during the LUX data processing at runtime, but due to the overlapping single and double electron size distributions, some double electron signals pollute this tag. This makes a simple average of the tagged electron pulse size insufficient to measure electron size stability. Instead, the electron pulse sizes for a given time period are plotted in a histogram and fit with a Gaussian distribution to extract the mean single electron size. The mean values of the fit versus time can be used to evaluate detector stability.

This is done in SciDB by utilizing two built-in SciDB functions: filter and regrid. First we use filter to select only those pulses that are identified as single electrons. The AFL command looks like this:

```
filter (PulseArray, pulse_type==single_electron)
```

This reduced the number of pulses to 200 million. This array still has the same dimensions of the mother array: time in 10 ns bins, pulse number in integer values, and pulse type (which is all single electron). In this example we were interested in the size of an electron pulse in time increments of 1 day. We want to produce a histogram of pulse sizes, which means we want a count of each pulse size occurrence for each time bin. To get this, we reshape this smaller array so that pulse size is a dimension. Then we used the regrid SciDB function to change the binning of the dimensions, collapsing the multidimensional array into a two dimensional array of about 2000 time bins and 60 pulse size bins. Because of the way the regrid function is structured, doing an aggregation call simultaneously has very little cost. We use this to generate an occurrence count for each pulse size bin.

Then we use the Scipy curve_fit function to fit each pulse size histogram with a Gaussian. The entire operation, starting from the first filter from the master array, takes 7 minutes. The result is shown in
Fig. 2. Please note that this analysis was performed with an early version of the LUX data processing output, and is not representative of the final values used in the LUX WIMP search analysis described in [3].

3.2. Signal Search

A WIMP interaction with the LUX detector would be characterized by a single nuclear recoil scatter at low energies (a few keV, depending on the WIMP mass). Because the inner 118 kg fiducial volume of xenon has the lowest background interaction rates, only events in that region are considered WIMP candidates. The search for dark matter interactions in LUX with the 2013 data loaded into the SciDB cluster involved searching through the 83 million events and finding all single scatters in the fiducial volume at low energies (S1 between 2 - 50 photons detected, S2 between 200 - 3300 photons detected). Additional basic quality cuts were applied after obtaining the result from the SciDB database. The reader should note that the analysis done in this example use case did not involve all of the more advanced data cuts used in the final LUX analysis, did not utilize detector stability cuts, and was performed with an early version of the processed data output. For the official WIMP search analysis result performed by the LUX collaboration, refer to [3]. This use case is only meant to illustrate the power of performing an analysis at this scale with a SciDB cluster.

First, all events in the EventArray tagged to contain only one S1 and one S2 pulse (a single scatter) were selected with a filter command in SciDB. Similarly, only S1 pulses between 2 - 50 photons detected and S2 pulses between 200 - 3300 photons detected were selected from the PulseArray with the filter function. The results from the EventArray and the PulseArray were joined with cross_join to only obtain the S1 and S2 pulses corresponding to the selected single-scatter events. Finally, the array of selected S1 and S2 pulses were joined by their event identifier so that each row corresponded to an event. The end result is a table where each row is an event that passes the single-scatter and low-
energy cuts, and the columns are the S1 and S2 properties for that event (which include the x, y, z
location for the interaction vertex). A plot of the squared radial distance from the detector center
versus depth in the detector for all of these events (where the depth is expressed in the raw units of
time delay between S1 and S2) is shown in Fig. 3.

Normally, this analysis requires a user to loop over 1 million ~10 MB files in the 10 TB dataset and
save the candidate events to another file. This process is slow (~day at best), cumbersome and thusly
the analysis steps are difficult to share. With the SciDB testbed, this search process took 1.5 minutes
from start to finish, which greatly enhances the analysis capabilities of the collaboration.

![Figure 3](image)

**Figure 3.** All single-scatter, low-energy particle interactions in the LUX detector
selected from about 83 million interaction records acquired in 3 months of livetime.
The position for each interaction vertex is shown in squared radius from the detector
center (R² vs. detector height (shown in units of time in µs) so that each bin
represents the same detector mass unit. Due to the lower background activity in the
center of the detector, only the interactions in the inner 118 kg fiducial volume (red
box) are considered for further WIMP dark matter analysis. All interactions were
consistent with the background-only hypothesis. Note that this example use case
was performed with an early version of the LUX data and only uses basic selection
criteria, so it is not representative of the official LUX WIMP search analysis.

4. Strengths and Weaknesses of SciDB

Loading and unloading data into SciDB can be difficult. Each instance has its own data store, so
offloading this can take some time. Because of this, SciDB is also not efficient for streaming analyses
where lots of data is read in and out.

SciDB offers a wide array of optimized functions for sampling, matrix operations (using
ScaLAPACK), and statistical calculations. There is also an interface for including user defined
functions. It offers optimized filtering and aggregating as well as more SQL-like table operations.
SciDB excels at offering a parallel architecture transparently to the user. Once the SciDB interface is
setup, the user does not need to know any information about the configuration to run an analysis.
Additionally, SciDB has a robust python and R interface, which lowers the threshold for learning
considerably.

5. Future Plans
In addition to successfully partnering with LUX, NERSC has also worked with bioinformaticians, climate scientists, and astronomers to develop SciDB use cases. These efforts have generated enough interest that NERSC is deploying a cluster dedicated to SciDB analyses. This cluster will have 12 IvyBridge, Intel Xeon nodes each with 20 cores and 64 GB of RAM. Future work includes expanding partnership with other scientific fields and workflows as well as investigating scaling on HPC systems.

References
[1] Y. Yao, et al., “SciDB for High-Performance Array-Structured Science Data at NERSC”, Computing in Science Engineering 17, 44 (2015).

[2] D. S. Akerib et al. [LUX Collaboration], “The Large Underground Xenon (LUX) Experiment,” Nucl. Instrum. Meth. A 704, 111 (2013) arXiv:1211.3788.

[3] D. S. Akerib et al. [LUX Collaboration], “First results from the LUX dark matter experiment at the Sanford Underground Research Facility,” Phys. Rev. Lett. 112, 091303 (2014) arXiv:1310.8214.