Fast Data Analysis with Integrated Statistical Metadata in Scientific Datasets

Jialin Liu
Department of Computer Science
Texas Tech University
Lubbock, Texas, USA
Email: jaln.liu@ttu.edu

Yong Chen
Department of Computer Science
Texas Tech University
Lubbock, Texas, USA
Email: yong.chen@ttu.edu

***Abstract*—Scientific datasets and libraries, such as HDF5, ADIOS, and NetCDF, have been used widely in many data-intensive applications. These libraries have their special file formats and I/O functions to provide efficient access to large datasets. Recent studies have started to utilize indexing, subsetting, and data reorganization to manage the increasingly large datasets. In this work, we present an approach to boost the data analysis performance, namely Fast Analysis with Statistical Metadata (FASM), via data subsetting and integrating a small amount of statistics into the original datasets. The added statistical information illustrates the data shape and provides knowledge of the data distribution; therefore the original I/O libraries can utilize these statistical metadata to perform fast queries and analyses. Various subsetting schemes can affect the access pattern and the I/O performance. We present a comparison study of different subsetting schemes by focusing on three dominant factors, the shape, the concurrency, and the locality. The added statistical metadata slightly increases the original data size, and we evaluate the cost and trade-off as well. This work is the first study that utilizes statistical metadata with various subsetting schemes to perform fast queries and analyses on large datasets. The proposed FASM approach is currently evaluated with the PnetCDF on Lustre file systems, but can also be implemented with other scientific libraries. The FASM can potentially lead to a new dataset design and can have an impact on big data analysis.**

***Keywords*—high performance computing; data-intensive computing; big data; storage systems; statistical techniques; FASM**

I. Introduction

The volume of data collected from instruments and generated from simulations for scientific discovery and innovation keeps increasing rapidly. For example, the Global Cloud Resolving Model (GCRM) project [4], part of DOE's Scientific Discovery through Advanced Computing (SciDAC) program, is built on a geodesic grid that consists of more than 100 million hexagonal columns with 128 levels per column. These 128 levels will cover a layer of 50 kilometers of atmosphere up from the surface of the earth. For each of these grid cells, scientists need to store, analyze, and predict parameters like the wind speed, temperature, pressure, etc. Most of these global atmospheric models process data in a 100-kilometer scale (the distance on the ground); however, scientists desire higher resolution and finer granularity, which can lead to significantly larger sizes of datasets. Another example is the data requirements of representative scientific applications run at Argonne Leadership Computing Facility (ALCF) through the DOE's INCITE program [17]. The data volume processed online by many of these applications has exceeded TBs or even tens of TBs; the off-line data is near PBs of scale. How to efficiently analyze and retrieve the datasets has become a key challenge in processing the rapidly growing volume of data.

Scientific datasets and libraries, including HDF5 [5], NetCDF [6, 7], PnetCDF [8], and ADIOS[1], have been well developed and widely used in recent years for scientific data management. With these libraries, datasets are stored in a specific format. Scientific applications can make use of these libraries' high performance I/O to accelerate the processing performance. Those libraries, however, are still not efficient enough for scientists, e.g., they lack non-procedural query and sufficient indexing. Thus researchers have begun employing existing useful techniques of traditional databases into scientific datasets. For example, FastBit is a powerful tool that implements indexing in datasets [21]. Users can use FastBit to build compressed bitmap indices over the datasets based on demand, and the performance can be improved dramatically.

In this study, we argue that *the raw datasets and current formats are not sufficient for achieving the best performance*. Significant performance improvements can be achieved with redesigning the datasets and enhancing relevant libraries. Our main idea is to perform data subsetting on raw datasets, and then add a small amount of statistical metadata into raw datasets to guide data accesses. Even though the idea of utilizing statistics in scientific datasets has been mentioned in [13, 15], none of their work presents a systematic study and detailed analyses on integrating statistics into scientific datasets.

In this paper, we present our investigation results and a system named Fast Analysis with Statistical Metadata (FASM). The contribution of this research is three-fold: First, we propose an idea of subsetting and adding statistical metadata into scientific datasets to facilitate data analysis and query processing. Second, we have designed and prototyped a FASM system to evaluate the idea. Third, we have carried out extensive experimental tests to verify the potential of FASM, and the results have confirmed that the FASM approach significantly improves the data analysis performance. It can have an impact on scientific dataset design, developing scientific data libraries and storage systems, and addressing big data challenges. An initial finding was presented in [14] and this paper presents a complete study of the FASM system.

978-1-4799-0898-1/13/$31.00

The rest of this paper is organized as follows. Section II discusses the background and the motivation of this work. Section III introduces the design of the FASM system. Section IV presents the evaluation results and analysis with both synthetic and real datasets. Section V discusses related work and compares with this study. We summarize this study and discuss future work in Section VI.

II. BACKGROUND AND MOTIVATIONS

Scientists are interested in understanding the phenomenon behind the data. The data in many data-intensive scientific applications are often written once and read many times, which is known as WORM access pattern [13]. For instance, astrophysics scientists try to understand cosmology behavior by analyzing data collected from large telescopes many times. Many other scientific applications share a similar discovery pattern, such as climate modeling, high-energy physics, medicine discovery, etc. These applications collect data from instruments and the datasets grow from many consecutive collections. These datasets will be read and analyzed many times later on in an attempt to understand the phenomenon. A typical data analysis conducted by atmosphere scientists for climate modeling study is shown in Figure 1. Scientists often need to perform such queries to select data points of interests for understanding the phenomenon behind the data.

```
Select data points
From datasets
Where pressure>80 And 12<temperature<25;
```

Fig. 1: Sample Query Request

Traditionally, without any prior knowledge of the datasets, one would need to look at all the data points and compare the values of pressure and temperature within a specified range to carry out this data analysis (e.g., 12<temperature<25), which is a costly process. Instead, we could potentially utilize the prior knowledge of the underlying datasets, such as the data structure, distribution, variable ranges, and other statistical information, to facilitate such data analysis. For instance, the pressure of the atmosphere is normally less than 70 (kPa) for 4,000 meters above the sea level [2]. If we could analyze the distribution and integrate a small amount of statistical information, such as the minimum and maximum value, into each subset, we could potentially improve the data analysis performance. For this specific example, if the maximum pressure within a subset is less than 80 (kPa), then there is no need to look into the data points in such subset, and therefore the data analysis performance can be significantly improved.

Figure 2 plots an initial performance comparison. This test was carried out using two groups of datasets with various dataset sizes. One group was integrated with statistical metadata (we term as metadata rich datasets), another was the original dataset. The input of the program was a range query request as described in Figure 1. We performed 30 tests and calculated the average. We can observe a clear performance improvement from the result. The data analysis response time was reduced significantly using the integrated statistical metadata by up to 4 folds.

Such statistical metadata, however, is not supported in existing data management libraries and programming interfaces. This study presents a systematic design of integrating such statistical metadata into scientific datasets and enhancing I/O libraries to facilitate data analysis. The idea presented in this research is similar to indexing schemes widely used to boost data access; however, not like an indexing scheme, statistical metadata will not increase the original dataset size as much as an indexing scheme does. A typical indexing scheme can double the size of original datasets [10], and can be a concern for big data applications with huge storage requirements. We argue that the proposed lightweight approach, with small amount of additional metadata, can improve the data processing performance without increasing the storage overhead dramatically, which is confirmed by the experimental results as well.

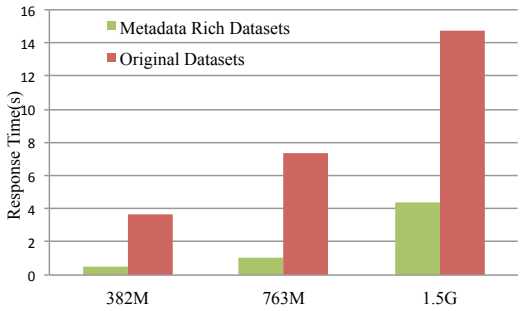

Fig. 2: Performance Improvement with Integrated Statistical Metadata

III. FAST ANALYSIS WITH STATISTICAL METADATA

In this section, we introduce the design of the proposed Fast Analysis with Statistical Metadata (FASM) system and discuss each component.

A. System Architecture

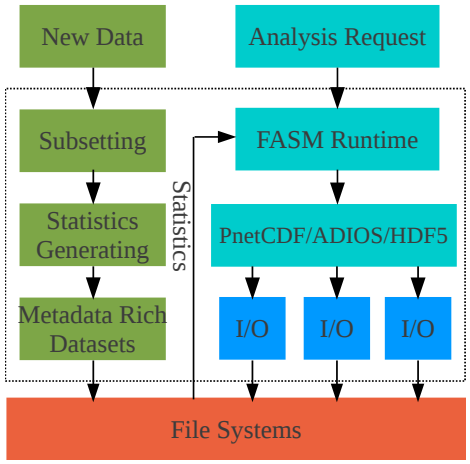

Fig. 3: System Architecture

As shown in Figure 3, the FASM system has four major components: Subsetting, Statistics Generating, Metadata Rich

Datasets, and FASM Runtime. When the datasets are stored in the file system at the first time, two components, subsetting and statistics-generating, will process the data first. The subsetting component will partition the datasets into subsets. The statistics-generating component will calculate the statistics of each subset and store them together with the raw data as the Metadata Rich Datasets. The enhanced datasets will be used in subsequent analyses, and the data access and analysis will be guided by the integrated statistics. These steps are supported by the enhanced I/O library and optimized access codes.

B. Subsetting

Given a query, e.g., Figure 1, there exists different optimizations to efficiently answer it and achieve less I/O cost in the data retrieval. The various subsetting schemes in FASM is designed to optimize the I/O. In real applications, the query usually targets a small subset. A carefully designed subsetting scheme can quickly locate the targeting subsets and reduce the I/O. For example, query happens on 2D planes at different altitude will perfer a pre-defined subsetting on 2D planes than 3D cubes.

Different ways of subsetting exist and can provide users various views of the datasets. The query and data analysis performance, in terms of response time and latency, could be varied. We define and study three subsetting schemes in this research. In current development, the FASM system maintains and supports different schemes at the runtime.

1) Dimension-Driven Subsetting: The first subsetting scheme, *dimension-driven subsetting* scheme, is shown in Figure 4 with a sample 3-D dataset. The scheme applies to other multiple dimensional datasets as well. The subfigure (a) illustrates a subsetting scheme with fine granularity. Each subset is extracted from only one dimension each time. The subfigure (b) shows a subsetting by partitioning the datasets into 2-D planes, either along the slowest dimension (in a real climate application, the slowest dimension is usually the time dimension) or along the fastest dimension. The subfigure (c) illustrates a 3-D subset scheme. The last subfigure (d) demonstrates a combination of the aforementioned subsetting schemes. A combination-subsetting scheme can potentially provide the user with rich statistical knowledge of the entire dataset. This is because that, for instance, in a global temperature dataset, the temperature value usually has different range and distribution in different seasons and different areas. Simple subsetting along one or two dimension can not reveal the data characteristics well. The input of a *dimension-driven subsetting* algorithm includes the original dataset and predefined subsetting scheme. The return includes the position and size of each subset.

For 1-D, 2-D, and 3-D subsetting schemes, the position and size of each subset are directly calculated and returned, e.g., subset_2D_i(time:1:2,level:2:3,lat:0:10,lon:0:10), in which, 'time:1:2' means subsetting from the second time step to the third time step (one time step totally). For the combination-subsetting scheme, the predefined subsets are extracted according to their scheme and size information. For instance, in subfigure (d) of Figure 4, there are three basic schemes: 1-D, 2-D, and 3-D. The subsetting scheme is recursively defined and performed till the entire dataset is partitioned. When all subsetting steps are done, the position and size of each subset are returned. The statistics-generating component in the FASM system utilizes this information to compute the needed statistical metadata.

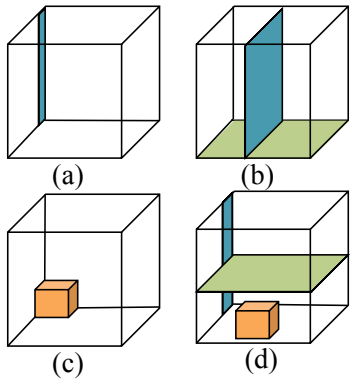

Fig. 4: Dimension-driven Subsetting

2) Locality-Driven Subsetting: It is often desired to prune the search space with integrated statistics, while accessing as much contiguous data as possible at the same time[11].

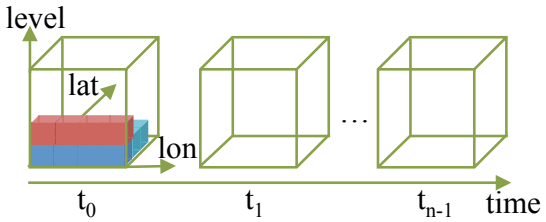

Fig. 5: Locality-driven Subsetting

We motivate the second subsetting scheme with the consideration of locality. We discuss how subsetting affects the locality and vice versa, using the same temperature model as an example. The dataset's four dimensions, ranked from the slowest to the fastest, are time, level, latitude, and longitude. For every two dimensions (2-D plane) of the 4-D datasets, we can calculate the statistics of all the points within the plane. Different subsetting schemes, however, can lead to various localities in accessing datasets. For instance, as shown in Figure 5, when the subsetting scheme is along the (lon, level) 2-D plane, then accessing a 2-D plane in these two dimensions will generate many non-contiguous I/Os. The elements on (lon, level:0) and (lon, level:1) are logically contiguous, however, the physical distance between them is $lon \times (lat - 1)$. The same situation happens in subsetting (lat, level) plane. When it comes to (lat, time) plane, the locality tends to be worse, as the physical distance between (lat, 0) and (lat, 1) is $(level \times lon - 1) \times lat$, which is the size of a single 3-D subset. Similar results can be found in (level, time) subsetting. Table I summarizes subsetting schemes and impact on locality (the distance between contiguous subsets).

The FASM considers the applications' read pattern to direct the subsetting scheme choice in consideration of data locality.

TABLE I: Subsetting Schemes and Locality

Type	Dimension	Distance
sub1	(lat, lon)	0
sub2	(lon, level)	$(lat-1)\times lon$
sub3	(lat, level)	$lat\times(lon-1)$
sub4	(lon, time)	$(level\times lat-1)\times lon$
sub5	(lat, time)	$(level\times lon-1)\times lat$
sub6	(level, time)	$level\times(lon\times lat-1)$

3) Concurrency-Driven Subsetting: Concurrency plays a critical role in exploring parallelism in the access and analysis of scientific datasets. It is contributed by data distribution among a set of nodes. In a parallel file system, the data are spread among multiple nodes. For example in Figure 6, using a traditional round-robin distribution strategy, the temperature datasets are distributed among four nodes, where each node contains four 3-D subsets. The FASM system calculates the concurrency before choosing a subsetting. We take the 4-D temperature example to show how the concurrency of possible subsetting is calculated.

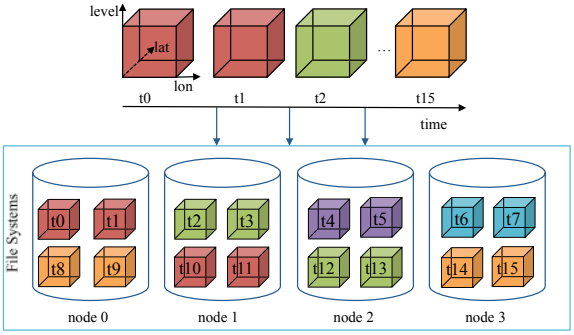

Fig. 6: Concurrency-driven Subsetting

Suppose the strip size is denoted as $strip\_size$, the number of storage nodes is n, and the size of each 3-D($level\times lon\times lat$) subset in this example is m. In PnetCDF, we need to specify the access length. For instance, when the access is conducted along the (lat, lon) plane, we can specify the number of level steps, denoted as x. Then on each node, there are $\frac{strip\_size}{m}$ subsets. The total number of levels in one node is $\frac{strip\_size}{m}\times level$. If we access x levels, the number of targeted storage nodes should be $\frac{x}{\frac{strip\_size}{m}\times level}$, and the maximum value is the number of the total nodes n. The concurrency of other subsetting schemes can be calculated too, and are shown in Table II. This table is used to guide the subsetting scheme in order to achieve the desired concurrency in parallel data accesses.

TABLE II: Subsetting Schemes and Concurrency

Scheme	Concurrency
sub1	$\min(\frac{(x\times m)}{(strip\_size\times level)}, n)$
sub2	$\min(\frac{(x\times m)}{(strip\_size\times lat)}, n)$
sub3	$\min(\frac{(x\times m)}{(strip\_size\times lon)}, n)$
sub4, sub5, sub6	$\min(\frac{(x\times m)}{(strip\_size)}, n)$

The FASM chooses a subsetting scheme based on the locality as well as the concurrency, the subsetting scheme with both high locality and high concurrency will be chosen first. But as we will see in the evaluation section, some subsetting schems usually reveal high locality and low concurrency at the same time. For the current FASM system, we utilize all the schemes in the runtime. In the future, we will have more intelligent way for subsetting scheme selecting.

C. Statistics Generating and Enhanced Datasets

There are different statistics we can utilize. A typical case is the minimum and maximum values extracted from the subset. With those statistics, we can skip searching certain amounts of subsets and improve the access performance. Other statistics can also be useful in guiding data analysis. For instance, the standard deviation and mean value can describe whether the datasets vary smoothly or fluctuate dramatically, which can be used to direct queries or to determine whether or not further subsetting when necessary.

```
netcdf temperature {
dimensions :
        time = UNLIMITED ; // (37658
currently)
        lat = 120 ;
        lon = 20 ;
        level = 30 ;
variables :
        int temperature(time, level,
lat, lon) ;
statistics:
min:
(time:0,level:0,lat:0,lon:0):10
(time:0,level:0,lat:0,lon:1):21
max:
(time:0,level:0,lat:0,lon:0):37
(time:0,level:0,lat:0,lon:1):47
data :
data =
 30, 21, 32, 33, 33, 32, 26, 37,
28, 29, 10, 11, 12, 12, 11, 15, 16,
17, 16, 19, 20, 21, 23, 24, 23, 26,
27, 27, 29, 30, 33, 32, 35, 34, 25,
46, 47, 38, 40, 40, …
```

Fig. 7: FASM Dataset Format

The raw dataset enhanced with subsetting schemes and generated statistics is called a metadata rich dataset in this study. A sample enhanced dataset generated from the original dataset is illustrated in Figure 7. We introduce a statistical metadata portion to describe the integrated statistics. In this example, min and max statistics are specified. The generated statistics are integrated with the raw dataset. At runtime, these statistics will guide data analysis and queries through the FASM Runtime component. The FASM system provides a user-defined method to extract the statistical metadata. With statistics selected and defined, the FASM system partitions datasets and generates statistics when they are first stored.

D. FASM Runtime

The FASM Runtime component leverages integrated statistical metadata to facilitate data analysis and queries. In PnetCDF library, the ncmpi_get_vara_type function can read an array of various types from a PnetCDF dataset (same with NetCDF dataset). The access pattern is defined by giving starting position and a vector of edge lengths. For example, in the function *int ncmpi_get_vara_float (int ncid, int varid, const size_t start [], const size_t count [], float *fp)*, access pattern is specified by starting position array 'start' and length array 'count'.

The FASM approach essentially optimizes these accesses at runtime by guiding the data range to be analyzed given subsetting and generated statistics. Without the support from the FASM, each access will start from the beginning position and run until the very end, meaning every data point is accessed. This process is costly and the results include many useless subsets. With FASM, after filtering the useless subsets, a vector storing the target position information is passed to the original read function. Thus, the read function gets a modified access pattern, in which each access has a modified starting position and reduced length for the target subset.

IV. EVALUATION

A. Experimental Platform

We have conducted experimental tests on a 640-node cluster. Each node within the cluster contains two Intel Xeon (Westmere) 2.8 GHz 6-core processors with 24 GB of memory. The nodes are connected with DDR Infiniband. Overall the cluster has a peak rating of 86 teraflops and a High Performance Linpack (HPL) score of 68 teraflops. We evaluated the FASM system with a group of datasets, which include both synthetic and real application datasets. For each dataset, we have varied subsetting schemes and statistics selected to verify the potential and study their impacts. The query response time and the overhead were analyzed.

B. Data Analysis and Queries Evaluated

We have generated a group of synthetic datasets. The file size was ranged from 300 KB to 100 GB. Each file was generated separately with different data values. The synthetic datasets were generated with the consideration of real application needs. For instance, the datasets usually spread across a multi-level grid. Therefore, in each box, we first generated a random value x, and the remaining data within the box were generated close to the x, (e.g., $x \pm 20$). With this way we can avoid a global normal distribution and match the synthetic datasets with the real datasets. We have also tested the FASM approach using a real dataset: the BCCR-BCM 2.0 model [3]. This is a climate model dataset downloaded from Bjerknes Center for Climate Research. The size of the dataset ranges from 100MB to 1.8GB, and the total size of these datasets is more than 12 GB. The data analysis and query statements were also generated randomly. We evaluated the FASM system using 20 different query statements. Different queries requested different volumes of data. The generated query statements had a normal distribution of those cases.

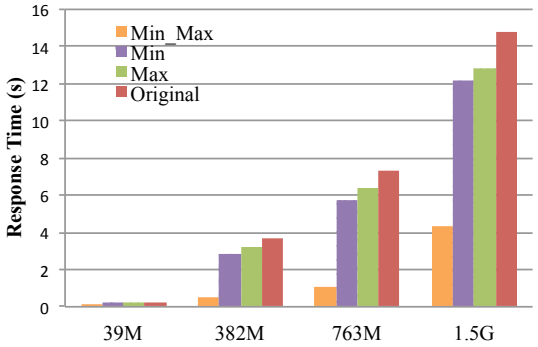

Fig. 8: Evaluations with Different Statistics

C. Performance Comparison of Statistics

The statistical metadata extracted in evaluations included the minimum and maximum values. In this subsection, we compare those different statistical metadata and their related performance. As reported in Figure 8, the FASM approach with integrated statistics did not improve the performance obviously when the dataset size was small. As the dataset size increased, the FASM approach demonstrated clear performance advantages. We have also shown different statistical metadata and their impacts. These two statistics (min and max) have a slight difference in query and analysis efficiency, but the difference is clearly dependent on the data distribution and data analysis requests.

D. Overhead Analysis

The integrated additional statistical metadata can cause storage overhead. We have measured the size of additional metadata for each subsetting scheme for a 4-D dataset and the result is shown in Figure 9. The conclusion is that the storage overhead is less than one percent for three subsetting schemes. The overhead is near nine percent for only one subsetting scheme. Overall, the FASM system can utilize this small amount of metadata to improve the performance without dramatically increasing the storage overhead, whereas the storage overhead is normally a concern for other techniques like indexing.

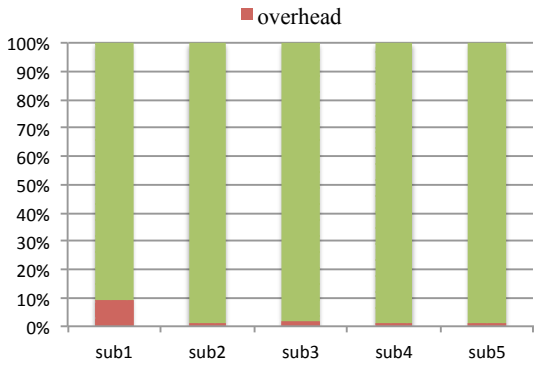

Fig. 9: Storage Overhead

The added metadata can also lead to a concern of computation cost. However, such cost only occurs once when data

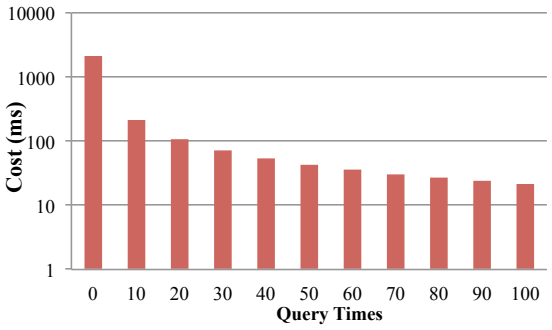

Fig. 10: Amortized Query Cost

generated and written to the file system for write-once and read-many types of applications. The incurred overhead by scanning data items and computing statistics will be offset by the read accesses of data analyses and queries. We present an amortized overhead analysis, which is reported in Figure 10. It can be observed that the average cost of query caused by initial analysis could be reduced to an acceptable value, when the number of query operations exceeds a certain value (e.g. 10 query operations).

E. Locality Analysis

We have carried out evaluations of the impact of locality on different subsetting schemes. This experiment was performed on a single node. We flushed the cache in each test. The dataset was a 382MBs 4-D dataset, which was generated randomly. We tested the FASM system with twenty query statements using five subsetting schemes discussed in Section III separately.

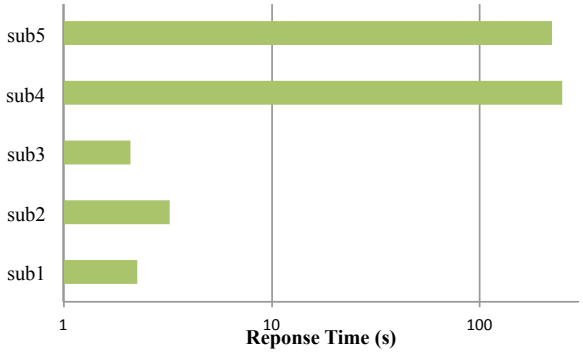

Fig. 11: Locality Analysis for Different Subsetting Schemes

As shown in Figure 11, the result matches with our theoretical analysis. Sub1, sub2 and sub3 schemes are better than sub4 and sub5 schemes. This observation confirms that the subsetting schemes of sub4 and sub5 decrease the locality, whereas sub1, sub2, and sub3 schemes are significantly better. The response time with sub4 and sub5 subsetting schemes was nearly 100 times greater than other subsetting schemes. This observation confirms that the locality is an important factor when accessing and subsetting scientific datasets. To be precise, the sub3 scheme is slightly better than sub1 and sub2 schemes. This is because that, even though the distance between two consistent data is the same as in the sub2 scheme and larger than that in the sub1 scheme, the data access pattern has an impact. The sub3 scheme accesses a bigger chunk each time and then moves certain distances to access another chunk; while in sub1 and sub2 schemes, the access of the same size of chunk as in the sub3 scheme results in non-contiguous accesses, which means multiple small non-contiguous chunks are read to construct a big chunk. Thus, the access latency was worse than that in the sub3 scheme. The FASM system supports different subsetting at the same time, with considering the locality, to achieve the desired performance for various access patterns.

F. Concurrency Analysis

To verify the impact of concurrency on subsetting schemes, datasets were stored on multiple nodes using a round-robin distribution on the Lustre file system. As discussed before, the access step and strip size are important factors that determine the concurrency. We set the striping units to 40 (that is, data blocks are striped over 40 OSTs). We varied the strip size as 1MBs, 2MBs, 5MBs, 10MBs and 50MBs separately to observe the impact. Based on the previous analysis on concurrency, we can derive the following general equation:

$$Concurrency = \frac{x \times m}{strip\_size \times k}, k \in \{lat, lon, level, 1\} \quad (1)$$

Since we have the exact dimensional size, we can calculate the relationship among concurrency, accessing steps, and strip size. For instance, from the equation 1, we know that the concurrency decreases as the strip size increases.

The result is shown in Figure 12. We only show three subsetting schemes in this test, which are enough to demonstrate the various performance in terms of concurrency. Other schemes were tested, showing same results, thus not reported here. The sub3 scheme achieved the best performance when the strip size is 5 MBs. The reason is that the locality and concurrency of accessing sub3 achieved an optimal balance when data are striped in 5 MBs. This observation matches with our theoretical analysis. The performance under such a setting is 1.67 times faster than the worst case.

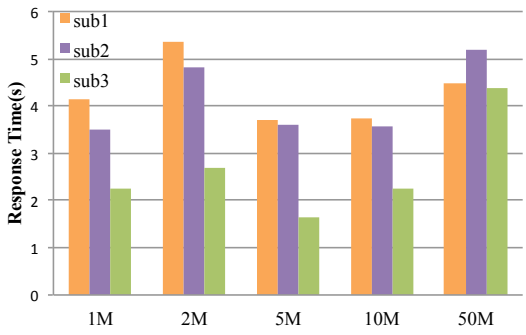

Fig. 12: Concurrency with Various Strip Sizes

In a real application, the access steps can be varied. For example, in sub1, the access is performed in the 2-D plane

($lat \times lon$) and the number of layers (levels) accessed at one time can be changed. We then have varied the number of steps and test the subsetting scheme 1. Figure 13 reports the results conducted on 40 storage nodes where the strip size was set to 5MBs. The dataset is 382MBs 4-D temperature data. As access steps were changed, the concurrency was changed too, with different response times. The best performance was 5.2 times faster than the slowest.

We can also find that the response time begun increasing when the access step was larger than 50. This is due to the I/O latency outweighing the concurrency gain. These tests confirmed that the concurrency plays a role in the overall system performance. The FASM system decides which subsetting scheme is better based on the data distribution and applications' read pattern.

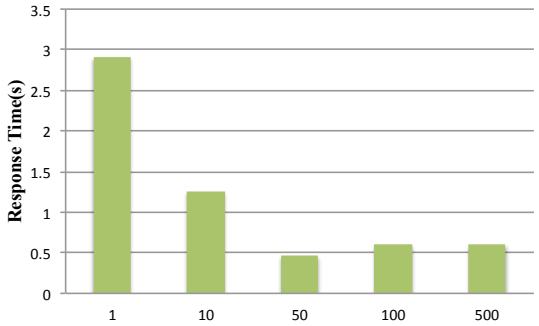

Fig. 13: Concurrency with Various Access Steps

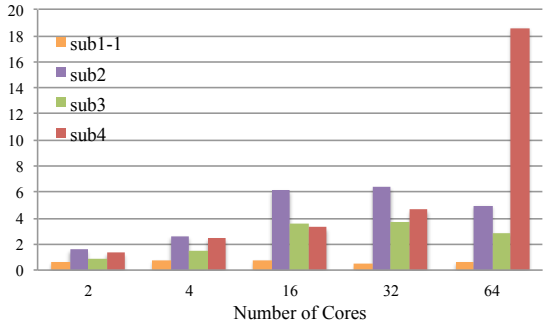

Fig. 14: Scalability Analysis of Different Schemes

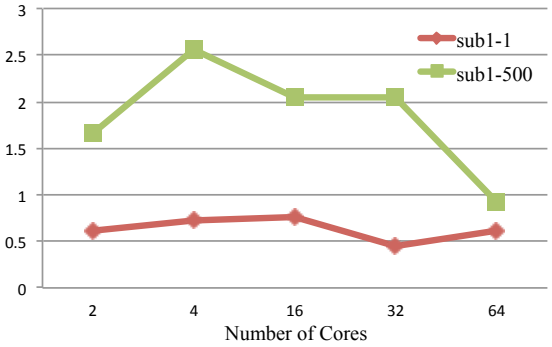

Fig. 15: Scalability Analysis of One Scheme with Different Access Steps

G. Scalability Analysis

We have carried out tests to verify the scalability of subsetting schemes in the FASM system as well. The results are reported in Figure 14. Y-aixs is the speedup. Previous evaluations show that certain subsetting schemes (e.g., sub 1) perform better than others in terms of locality; however, they tend to not scale well. We plot sub4 in Figure 14 to show that even one subsetting is worse in locality, but it could be better in scalability. Through such comparison and previous analysis, we can see there is no best scheme for all cases. The performance is partially determined by how the applications access the data. Therefore, we keep all schemes at the same time in current FASM system (each scheme has minor additional statistical metadata, thus does not increase the storage overhead considerably). The system utilizes all schemes to optimize the access. We have also tested the scalability with respect to the access steps, and Figure 15 shows the results (Y-axis is speedup). Sub1-1 refers to the subsetting scheme 1 with only one access step. Sub1-500 refers to the subsetting scheme 1 with accessing 500 levels each time. Sub1-500 tends to be more scalable due to the larger data chunks, while sub1-1 issues many non-contiguous small I/Os. This shows that the number of steps can have an effect on the performance, the FASM makes sure that the number of steps is not too small to cause scalability issue.

H. Real Application Test

We have performed tests with real application datasets from the BCCR model as well. Six NetCDF datasets were evaluated. Each dataset contains three dimensions: time, latitude and longitude. The latitude dimension has 64 or 128 different values. The longitude dimension has 128 or 256 different values. The time dimension is unlimited. Figure 16 shows the performance improvement. It can be observed that the FASM approach clearly reduced the response time in all cases. Taking the 1.8GB dataset as an example, the analysis cost was reduced from 5.5 to 3 seconds. The FASM approach was able to achieve up to 3.5 times speedup across all tests.

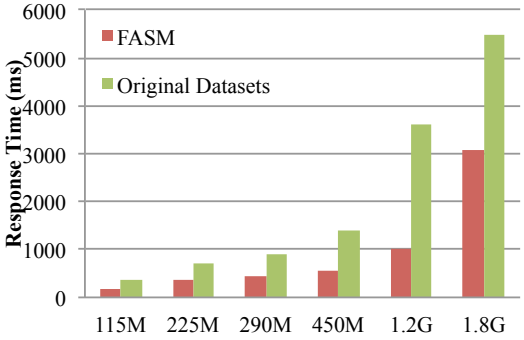

Fig. 16: BCCR Model Test Using FASM

V. Related Work

In [12, 16, 19], the authors introduced an overview of the trend of future data management. The authors concluded

a common set of requirements for a new science database system, which is named SciDB. In recent years, techniques in traditional databases have been integrated into scientific datasets [18]. The indexing scheme is a representative example. In [10, 21], the authors developed the FastBit system to provide a set of compressed bitmap indexes. Various research efforts have been carried out to optimize queries over the scientific dataset. In [20], the authors designed tools to support user-defined subsetting and aggregation over NetCDF datasets. In their system, the query results are aggregated on the server side before returning to the process, which reduces the data transfer cost. In this research, we propose and study a fast data analysis approach with integrated statistical metadata. The proposed approach has a similar idea to the indexing scheme, but is lightweight and incurs significantly less storage overhead when compared to the indexing schemes. In addition, the FASM approach complements existing database techniques that are already adopted for scientific datasets. An even better performance is expected when they are combined together.

The ADIOS also demonstrates that integrating the statistical metadata could potentially improve the performance [1, 9]. However, the current ADIOS only supports collecting local, simple, statistical and/or analytical data during the output operation for use in identifying desired datasets. The FASM system provides the mechanism of integrating the statistical metadata during the initial write. In addition, this metadata can be further utilized. This research has studied various subsetting schemes and the impact on performance as well. Parallel netCDF (PnetCDF) is a library that provides high-performance I/O while maintaining file-format compatibility with Unidata's NetCDF [8]. Our FASM system is prototyped based on PnetCDF currently. All these work have shown the importance of scientific datasets and data management libraries. In this study, we propose an approach to subsetting dataset and integrating with statistics. Our study complements these existing datasets and libraries and can be helpful to many data-intensive applications.

VI. CONCLUSION AND FUTURE WORK

Scientific datasets are used in many data-intensive computing fields, such as climate modeling, high-energy physics, astrophysics, computational chemistry, computational biology, medicine discovery, etc. Research efforts have started integrating modern database techniques and parallel I/O into the management of scientific datasets. In this work, we proposed a new idea of adding statistical metadata into the datasets. The statistical metadata illustrates the data distribution features, therefore, the parallel access code can utilize this metadata to perform fast queries and data analysis. Through experimental tests and evaluations, we have shown how the integrated statistical metadata improves the query and analysis performance. We have also analyzed the impact of various subsetting schemes, statistics generating, scalability, and overhead. The evaluation results have indicated that the FASM approach is promising and can be beneficial to many data-intensive scientific applications. In the near future, we will investigate further to support data modifications, resubsetting, and regeneration of statistics in the FASM at runtime. We will study the feasibility of automatic selection of statistics considering applications' access patterns as well.

REFERENCES

[1] ADIOS. http://www.olcf.ornl.gov/center-projects/adios/.

[2] Atmospheric pressure. http://en.wikipedia.org/wiki/Atmospheric_pressure.

[3] BCCR model. http://www.bcm.uib.no/.

[4] The Global Cloud Resolving Model (GCRM) project. http://kiwi.atmos.colostate.edu/gcrm/.

[5] HDF5. http://www.hdfgroup.org/HDF5/doc/index.html.

[6] NetCDF. http://www.unidata.ucar.edu/software/netcdf/.

[7] Netcdf climate and forecast (cf) metadata convention. http://cf-pcmdi.llnl.gov/.

[8] PnetCDF. www.mcs.anl.gov/parallel-netcdf.

[9] Hasan Abbasi, Jay F. Lofstead, Fang Zheng, Karsten Schwan, Matthew Wolf, and Scott Klasky. Extending I/O through high performance data services. In *CLUSTER*, pages 1–10. IEEE, 2009.

[10] Jerry Chou, Kesheng Wu, and Prabhat. Fastquery: A general indexing and querying system for scientific data. In Judith Bayard Cushing, James C. French, and Shawn Bowers, editors, *SSDBM*, volume 6809 of *Lecture Notes in Computer Science*, pages 573–574. Springer, 2011.

[11] Sakti P. Ghosh. Consecutive storage of relevant records with redundancy. *Commun. ACM*, 18(8):464–471, 1975.

[12] Jim Gray, David T. Liu, María A. Nieto-Santisteban, Alexander S. Szalay, David J. DeWitt, and Gerd Heber. Scientific data management in the coming decade. *SIGMOD Record*, 34(4):34–41, 2005.

[13] C. Ryan Johnson, Markus Glatter, Wesley Kendall, Jian Huang, and Forrest M. Hoffman. Querying for feature extraction and visualization in climate modeling. In *9th International Conference on Computational Science , Baton Rouge, LA, USA, May 25-27, 2009*, volume 5545, pages 416–425.

[14] Jialin Liu and Yong Chen. Improving data analysis performance for high-performance computing with integrating statistical metadata in scientific datasets. In *HPCDB, SC'12*, 2012.

[15] Jay F. Lofstead, Fang Zheng, Scott Klasky, and Karsten Schwan. Adaptable, metadata rich IO methods for portable high performance IO. In *IPDPS*, pages 1–10. IEEE, 2009.

[16] Jaechun No, Rajeev Thakur, and Alok Choudhary. Integrating parallel file I/O and database support for high-performance scientific data management. In *Proceedings of SC2000*, Dallas, TX, November 2000.

[17] Robert Ross, Robert Latham, Marc Unangst, and Brent Welch. Parallel I/O in practice, tutorial notes. In *SC'08*. ACM/IEEE, Austin, TX, November 2008.

[18] Rishi Rakesh Sinha, Soumyadeb Mitra, and Marianne Winslett. Bitmap indexes for large scientific data sets: a case study. In *IPDPS*, 2006.

[19] Michael Stonebraker, Jacek Becla, David J. DeWitt, Kian-Tat Lim, David Maier, Oliver Ratzesberger, and Stanley B. Zdonik. Requirements for science data bases and sciDB. In *CIDR*. www.cidrdb.org, 2009.

[20] Yu Su and Gagan Agrawal. Supporting user-defined subsetting and aggregation over parallel netCDF datasets. In *CCGRID*, pages 212–219. IEEE, 2012.

[21] Kesheng Wu. Fastbit: Interactively searching massive data, September 22 2009.