Visual Analytics with Unparalleled Variety Scaling for Big Earth Data

Lina Yu[1], Michael L. Rilee[2,3], Yu Pan[1], Feiyu Zhu[1], Kwo-Sen Kuo[2,4,5], Hongfeng Yu[1]
[1]University of Nebraska-Lincoln, Lincoln, NE, USA
[2]NASA GSFC, Greenbelt, MD, USA
[3]Rilee Systems Technologies, Derwood, MD, USA
[4]Bayesics, LLC, Bowie, MD, USA
[5]University of Maryland, College Park, MD, USA

***Abstract*—We have devised and implemented a key technology, SpatioTemporal Adaptive-Resolution Encoding (STARE), in an array database management system, i.e. SciDB, to achieve unparalleled variety scaling for Big Earth Data, enabling rapid-response visual analytics. STARE not only serves as a unifying data representation homogenizing diverse varieties of Earth Science Datasets, but also supports spatiotemporal data placement alignment of these datasets to optimize a major class of Earth Science data analyses, i.e. those requiring spatiotemporal coincidence. Using STARE, we tailor a data partitioning and distribution strategy for the data access patterns of our scientific analysis, leading to optimal use of distributed resources. With STARE, rapid-response visual analytics are made possible through a high-level query interface, allowing geoscientists to perform data exploration visually, intuitively and interactively. We envision a system based on these innovations to relieve geoscientists of most laborious data management chores so that they may focus better on scientific issues and investigations. A significant boost in scientific productivity may thus be expected. We demonstrate these advantages with a prototypical system including comparisons to alternatives.**

***Index Terms*—STARE; SciDB; array database; variety; data analysis; load balancing; indexing; GIS**

I. Introduction

Earth Science data obtained from diverse sources have been routinely leveraged by scientists to study various phenomena. The principal data sources include observations and model simulation outputs. These data are characterized by spatiotemporal heterogeneity originating from different instrument design specifications and/or computational model requirements that are preserved in data generation processes. Such inherent heterogeneity poses several challenges in exploring and analyzing geoscience data. First, scientists often wish to identify features or patterns co-located among multiple data sources to derive and validate hypotheses. Heterogeneous data make it a tedious task to look for such features in dissimilar datasets. Second, Earth Science data are multivariate. It is challenging to tackle the high dimensionality of Earth Science data and explore relationships among multiple variables in a scalable fashion. Third, there is a shortage of lucidity in traditional automated approaches, such as feature detection or clustering, in that scientists often cannot adeptly interact with data during analysis and intuitively interpret results.

To address these issues, we present a new scalable approach that facilitates the scientific analysis of voluminous and diverse geoscience data in a distributed environment. The major contributions of our work are:

- *SpatioTemporal Adaptive-Resolution Encoding (STARE)*. We have implemented STARE as the basis of a unified data model and an indexing scheme for geo-spatiotemporal data to address the variety challenge of Big Data in Earth Science. With the generality of unifying at least the three popular geospatial data models, i.e. Grid, Swath, and Point, used in current Earth Science data products, data preparation time for interactive analysis of diverse datasets can be drastically reduced, achieving unparalleled variety scaling.
- *Spatiotemporal data placement alignment in a distributed environment*. We have applied STARE to extending the capabilities of the array database SciDB using a data placement strategy employing STARE-based data access patterns of our scientific analysis in a distributed environment. With these STARE-enabled tools, we can scalably co-locate data spatiotemporally by placing data chunks directly to the correct nodes, avoiding costly data transfer and repartitioning and ensuring scalable performance.
- *A visual analytics prototype*. We have constructed a high-level graphical query interface that allows users to easily express customized queries to search for features of interest across multiple heterogeneous datasets. For identified features, we have developed a visualization interface that enables interactive exploration and analytics in a linked-view fashion. Specific visualization techniques are employed in each view to facilitate easy and interactive exploration into various aspects of identified features. A user can interactively and iteratively build insights into the data through a variety of intuitive visual analytic operations.

II. Background and Motivation

Facing the deluge of ever-increasing data volume, there have been many efforts attempting to address the growing challenge of Earth Science data practice with Big Data technologies. Three data models generally represent the spatial variety of geoscience data: Grid, Swath, and Point. These data are typically volumetric, multivariate, and time-varying. Most of the Big-Data efforts emphasize the *volume* aspect of the challenge. We, however, have recognized *variety* as

978-1-5386-2715-0/17/$31.00 ©2017 IEEE

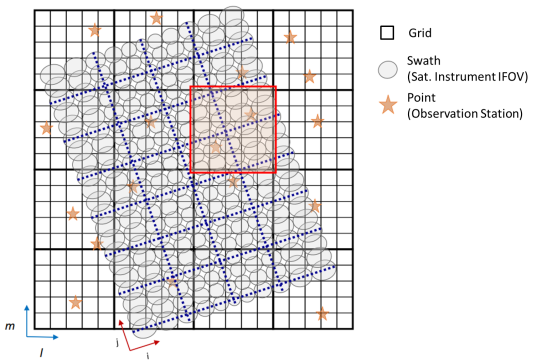


Fig. 1. Three Earth Science data models.

the key [1] beyond *volume* to attaining optimal scientific value. Moreover, after identifying certain regions of interest, scientists often conduct detailed data analysis and visualization using combinations of queries based on possibly complex functions of the primary variables [2]. Without special care, variety coupled with complex analysis naturally leads to poor data access patterns impacting data placement and system performance. This grand challenge can be illustrated by the disparity between data placement and data variety.

A. Data Placement Challenge

Data placement is especially important for technologies in which compute and storage are tightly coupled such as a database management system (DBMS). However, most existing frameworks for distributed big data analytics, such as MapReduce [3] and Spark [4], are only loosely coupled with the storage system (e.g., HDFS [5], Cassandra [6], etc.). Loosely coupled technologies provide better flexibility or elasticity but suboptimal performance and efficiency with equivalent resources. For example, Hadoop [7], the open-source version of MapReduce, is simple and worthy of high praise for one-pass computations, but it is inherently inefficient for multi-pass computations due to the lack of primitives for sharing intermediate states of the computation among passes, instead sending and retrieving intermediate states across a distributed file system. Thus, the communication and I/O overheads affect the overall performance.

B. Data Variety Challenge

Researchers have proposed various solutions based on the loosely coupled frameworks (e.g., Spark and Hadoop) and exploited multiple computer nodes (e.g., a cluster) in a distributed environment [8]. These solutions are viable for tackling the volume challenge of Big Earth Data, but have mostly downplayed even neglected the variety challenge.

Figure 1 illustrates the three data models: Grid, Swath, and Point. The Grid is shown in a black grid mesh with fixed latitude and longitude spacing. A simple linear relation exists between array indices and latitude-longitude geolocation coordinates. Swath retains a spaceborne instrument's observation geometry (e.g., cross-track $\times$ along-track) for its Instantaneous Fields of View (IFOV), and no simple relation exists between data array indices and geolocations. Point model is used mostly for in-situ observations at irregularly distributed locations. In contrast to a Grid's regular relationship between index and geolocation, geolocations of Point and Swath data elements are specified individually.

However, the different data models are specified using different reference frames and have different resolutions resulting in arrays of different shapes, even different geolocation representations. Thus, without a unifying representation, it is difficult, if not impossible, to spatiotemporally align their partitions when they are placed across cluster nodes. Misaligned arrays must be aligned on-the-fly, a computationally expensive process, before they can be processed together for integrative analysis, such as conditional subsetting and comparisons between multiple, dissimilar datasets. Since most integrative analysis requires spatial and/or temporal coincidence, data should be laid out so that data for the same spatiotemporal partition reside on the same node. Difference in data representations and array shapes thus gives rise to a formidable challenge to data co-alignment.

III. Our Approach

We aim to enable scalable data storage and analytics by holistically addressing Big Earth Data. We develop a new indexing scheme, named SpatioTemporal Adaptive-Resolution Encoding (STARE), to spatiotemporally co-align arrays of different shapes and data models. Using STARE, we tailor a data placement strategy to the data access patterns of scientific analysis, leading to optimal data partitioning and distribution in a distributed environment.

A. STARE

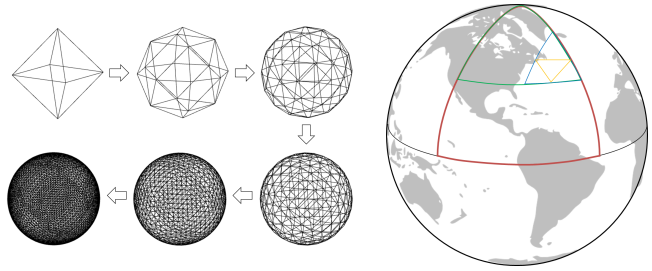
In our effort of developing viable approaches to achieve optimal scientific value from Big Earth Data, we realize that an appropriate indexing scheme for geolocation is crucial. The primary requirements for such an indexing scheme include:

R1: Support spatiotemporal data placement alignment.
R2: Include resolution information of the underlying data.

The rationale for the first requirement is straightforward. Most integrative analyses in Earth Science require spatiotemporal coincidence. Data placements aligned spatiotemporally ensure the minimization of node-to-node communication on a distributed parallel database and, as a result, optimization of performance. The second requirement ensures set operations over multiple, diverse datasets for integrative analyses can be carried out robustly and consistently without sacrificing performance.

STARE is an innovative outcome satisfying the requirements. It consists of two parts, a spatial component and a temporal component.

1) Spatial component: Hierarchical triangular mesh (HTM) [9] is a way to address the surface of a sphere using a hierarchy of spherical triangles. We build on the work of Kunszt *et al.* [9] who used HTM to index for the Sloan Digital Sky Survey (SDSS) [10]. The mesh is generated with the procedure below:



(a) (b)

Fig. 2. (a): HTM evolution within 5 iterations. (b): An example illustrating hierarchical triangular mesh: triangles N0 shown in red, N01 shown in green, N012 shown in blue, and N0123 shown in yellow.

1: Start with an inscribing octahedron (or other platonic solid) of a sphere. South (north) is labeled with a 0 (1), and the 4 triangles of each half are labeled 0-3 starting from the triangle nearest the x-axis proceeding counter-clockwise around the sphere as viewed from outside and above the poles.
2: Bisect each edge of the triangular facets.
3: Bring the bisecting points to inscribe the sphere to form 4 smaller spherical triangles.
4: Repeat from Step 2, until a desired resolution is reached.

We show the results of HTM evolution within 5 iterations in Figure 2(a). After the initial octahedron, each iteration is termed a quadfurcation, i.e., division/branching into 4 parts, requiring 2 more bits to index. An example is illustrated in Figure 2(b).

The spatial component of STARE is a customized variant of the HTM with two distinctions. First, while right-justified encoding is used for the original HTM indexing, we choose a left-justified encoding to facilitate spatial data placement alignment. Second, data resolution is added to the encoding using a few least-significant bits to facilitate set operations among diverse datasets.

The conventional HTM implementation had a simple map from symbols to integers in a Right Justified Mapping (RJM). However, RJM maps points in geometric proximity (e.g., in the offspring triangles of the same parent) to multiple, separated locations on the number line. Thus, implementing set operations (e.g., intersection) under RJM is complicated by this one-to-many mapping of the geometric points (in triangles) along the number line. For example, as shown in Figure 3, geometrically S0123 (corresponding to the digital value 539 in RJM) contains S01230 (2156 in RJM), but when mapped to integers N0123 (795 in RJM) it lies in between, even though S0123 and S01230 share the same prefix to the 3rd level. Thus, mapping HTM regions to contiguous RJM integer intervals holds only within the same HTM index levels, whereas our diverse datasets have a range of spatial resolutions.

Instead of using RJM, we use a Left Justified Mapping (LJM) bit format to encode HTM. Figure 4 shows the corresponding result of our example (using 12 bits for clarity).

$$\begin{aligned} S0123 &\Rightarrow 0b1000011011 = 0x21b = 539 \\ N0123 &\Rightarrow 0b1100011011 = 0x31b = 795 \\ S01230 &\Rightarrow 0b100001101100 = 0x86c = 2156 \end{aligned}$$

Fig. 3. RJM encoding example.

$$\begin{aligned} S0123 &\Rightarrow 0b100001101100 = 0x86c = 2156 \\ N0123 &\Rightarrow 0b110001101100 = 0x31b = 3180 \\ S01230 &\Rightarrow 0b100001101100 = 0x86c = 2156 \end{aligned}$$

Fig. 4. LJM encoding example.

However, in this case, we have an aliasing problem, e.g., with S0123 and S01230. LJM cannot distinguish between levels, as it does not track how many bits from the left are significant. To address this issue, we use a few least significant bits of a signed 64-bit integer to encode the approximate data resolution. Since we limit the number of quadfurcations to 27 (corresponding to ∼7-cm resolution), together with the starting 8 facets of the octahedron, only 57($= 27 \times 2 + 3$) bits are needed. Excluding the sign bit, we have 6 bits left to encode the quadfurcation level with the closet resolution to (but covering) that of the data.

Table I lists the bit-field arrangement of STARE's spatial index. Figure 5 shows the triangles from our previous example. This encodes the difference between S0123 and S01230 and integer comparisons and can tell us that the former contains the latter.

$$\begin{aligned} S0123 &\Rightarrow 0x06c0000000000003 \\ S01230 &\Rightarrow 0x06c0000000000004 \\ N0123 &\Rightarrow 0x46c0000000000003 \end{aligned}$$

Fig. 5. STARE spatial encoding example.

Essentially, STARE's spatial index concisely and uniquely maps a floating-point latitude-longitude 2-tuple to an integer with a given precision. For example, at the 23rd quadfurcation level the mapping has a ∼1-m uncertainty.

2) Temporal component: STARE's temporal component is also hierarchical but uses conventional date/time units to avoid unnecessary translations between temporal frameworks. Table II provides just one example encoding with a maximum time resolution of milliseconds, common for observations obtained from spacecraft. As an example, for an observation made on 2015 June 12, 8:10 AM with millisecond resolution, STARE yields

[+] 000-002015-06-3-3 08:0600.000 (07).

Here, the (07) corresponds to the highest level (finest) resolution available, milliseconds, and the [+] signifies positive

TABLE I
LEFT JUSTIFIED MAPPING OF STARE SPATIAL ENCODING.

Starting Bit	Ending Bit	No. Bit	Meaning
63	63	1	Reserved (Most Significant)
62	62	1	North-South Bit
60	61	2	Octahedral triangle index (Resolution level 0: ∼10,000 km)
6	59	54	Quadtree triangle index (Resolution levels 1-27: ∼5,000 km to ∼7 cm)
5	5	1	Reserved
0	4	5	Resolution level (Least Significant)

TABLE II
AN EXAMPLE OF STARE TEMPORAL ENCODING.

Starting Bit	Ending Bit	No. Bit	Meaning
0	2	3	Resolution/Unit
3	12	10	Millisecond
13	24	12	Second
25	29	5	Hour
30	32	3	Day of week
33	34	2	Week
35	38	4	Month
39	48	10	Year
49	58	10	Kilo-annum
59	62	4	Mega-annum
63	63	1	Before/After

years. With the Unix-based convention of numbering months starting at 0 and days at 1, the native STARE format can be read (partly) as the 3rd day of week 3 of month 6 (the 7th regularized 28-day month). Conversion to an array index is simple, and alternative encodings may be devised to meet application requirements.

3) Deployment: We use STARE to support our research into the automated analysis of phenomena such as blizzards as moving objects. We are extending the capabilities of the array database SciDB and developing tools (database ingest, preprocessing, and visualization) using STARE through a software library and APIs.

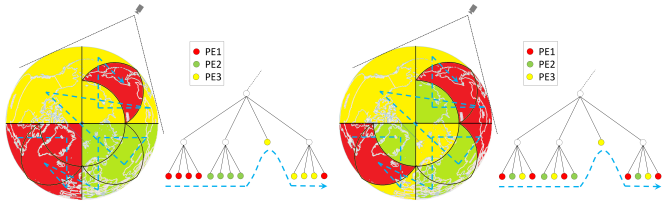
SciDB [11] is an open-source distributed data management system used primarily for application domains involving very large-scale array data. SciDB automatically distributes data and computational load across an arbitrary number of hardware instances while presenting an end user with the experience of a single, unified system. Unlike relational databases, which store data in flat tables, SciDB stores data in multidimensional arrays. Not only do multi-dimensional arrays generally lend themselves well to the representation of complex scientific data, they also permit the construction of an ever-growing library of efficient mathematical operators. In addition to its built-in libraries, SciDB provides user-defined types (UDTs), user-defined functions (UDFs), and user-defined operators (UDOs) to extend its core capabilities.

We have incorporated STARE to SciDB using its UDT and UDF facilities, including a STARE UDT verifying functions for constructing SciDB indices. Tagging Earth Science data with STARE ranges allows the geometric comparison, co-registration, and selection of diverse kinds of data via efficient metadata operations.

B. Data Placement Strategy

Building on our STARE indexing scheme, we study data access patterns in user analytics operations, and employ a STARE-based data placement strategy to effectively partition and distribute Earth Science data in a distributed environment.

1) Data access patterns: With the advances in computing and visualization techniques, nowadays Earth Science data is mostly explored and studied via visual analytics. Visual analytics operations can be roughly categorized into two types: *view-dependent* and *data-dependent*. For view-dependent operations, users access data by specifying their camera or view positions (e.g., visualizing data within a view port). For data-dependent operations, users access data by specifying their data of interest (e.g., querying data within a certain range or within a certain time interval). These two types of operations are often combined in practice. For example, users can apply visualization techniques to control the visual properties of queried data variables and find the features or regions of interest.



(a) (b)

Fig. 6. The spatial decomposition of a spherical surface using STARE and its corresponding hierarchical triangular mesh. A preorder traversal of triangle leaves is equivalent to the space-filling curve on the spherical surface. (a) shows that we evenly assign the triangle leaves among three processors from left to right in the hierarchical triangular mesh tree, and the distribution of their regions is contiguous along the space-filling curve. In this case, each processor's regions may not be always visible from different viewing directions. (b) shows that we assign the triangle leaves among three processors in round robin, and the neighboring regions are largely assigned to different processors. In this case, a portion of a processor's regions can be visible from any viewing direction.

2) Data partitioning and distribution: In a distributed environment, to achieve workload balancing among different processors, a common practice is to use regular gridding to divide a dataset into uniform blocks and evenly or randomly assign the blocks among the processors. When the block size is sufficiently small, it is easy to achieve workload balancing, but the overhead (increased bookkeeping due to the large number of blocks) counteracts overall performance.

We can achieve a more sparseness-adaptive assignment by leveraging the spatial locality encoded in a tree structure, such as a linear quadtree or a hierarchical triangular mesh quadtree [9], when data exhibit unbalanced density. STARE indexing scheme essentially encodes geolocations of multiple datasets, resulting in a unified quadtree for each time slice. A preorder traversal of the corresponding quadtree leaves is a space-filling curve which groups spatially nearby triangles together on the spherical surface. This characteristic can be used for optimizing data layout.

If we assign the processors contiguously along the space-filling curve for parallel processing, each processor will be responsible for contiguous regions on the surface as shown in Figure 6(a). However, users may only see a part of a region on the earth, and other regions are occluded from a certain camera position. Therefore, a contiguous assignment may result in unbalanced workload and suboptimal data locality. To address this issue, we distribute the triangles among the processors along the space-filling curve in the round robin fashion as shown in Figure 6(b). Data operations and calculations associated with a given triangle are also distributed in the same scheme. This guarantees that, given a sufficient number of

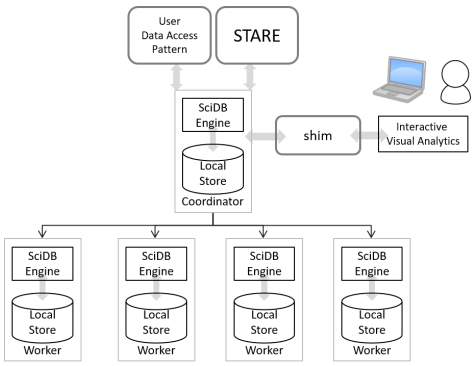


Fig. 7. The framework of our distributed storage and analytics system.

processors, the neighboring regions are assigned to different processors to achieve load balancing during users' interactive exploration.

IV. System Framework

We develop a distributed storage and analytics system based on our STARE indexing scheme and data placement strategy to tackle Big Earth Data.

The overall framework of our system is a typical three tier web application that is composed of a SciDB distributed database, a *shim* layer acting as the web server, and a front-end with STARE as a plugin of UDF to SciDB, as shown in Figure 7. Shim is a basic SciDB client that exposes SciDB functionality through a simple HTTP API. We implement the front-end to query data through the shim web server to SciDB and render data as an overlay on Google Maps within a browser. Every query from the front-end is encapsulated in the HTTP GET request as the suffix of the Uniform Resource Identifier (URI) and is encoded appropriately.

The back-end of our system, SciDB, is designed to run on a shared nothing architecture and allow scientists to analyze voluminous datasets as arrays with little concern for the details as to how massively parallel processing is achieved. It is comprised of a large set of worker nodes, a subset of which also act as coordinator nodes. A Postgres database runs on a coordinator node to manage all the metadata about the nodes, instances, arrays and so on. Each node contains its local SciDB engine and data store. Doan *et al.* [12] evaluated the impact of data placement on SciDB and Spark. SciDB can outperform Spark+HDFS and Spark+Cassandra by a factor of approximately 3-10 for equivalent integrative analysis, when data placement alignment is exploited.

Similar to a relational database, SciDB also has a query executor to parse, optimize, and execute queries written in the Array Query Language (AQL) and the Array Functional Language (AFL), which can be extended with UDTs, UDFs, and UDOs.

Using STARE to organize data before database ingests, we can place data chunks directly to the correct nodes, avoiding costly data transferring and repartitioning. With the enormous scale of Earth Science datasets, the savings from eliminating costly data repartitioning (or redimensioning) cannot be overemphasized. Because STARE uniformly translates geolocations and times to integer spatial and temporal indices with resolutions encoded therein, placements of data represented by different models and with different resolutions are aligned in storage. They naturally co-locate in space and time when distributed across computing and storage resources.

TABLE III
Datasets used in our experimental study.

name	type	time interval	latitude range	longitude range	resolution	size/slice
MERRA-2	grid	1 hour	90°S - 90°N	180°W - 180°E	576×361	0.83M
NMQ	grid	5 minutes	20°S - 55°N	130°W - 60°E	7001×3501	98M
TRMM	swath	15 slices/day	40°S - 40°N	180°W - 180°E	1601×7201	46M

We developed two UDFs to ingest data using STARE with intuitive APIs, specifically,

```
hstmFromLevelLatLon(int64 level, double
latitude, double longitude)
```

converting a geospatial location into a STARE spatial index, and

```
temporalIndexFromTradYrMoDyHrMiSeMsRl(int64
year, int64 month, int64 day, int64 hour,
int64 minute, int64 second, int64 minisecond,
int64 resolution)
```

converting a time into a STARE temporal index. Then, the process of ingesting data consists of three steps:

1: Use SciDB built-in *apply* operator to add the computed STARE spatial indices and temporal indices as new attributes into the array.
2: Use SciDB built-in *redimension* operator to change the spatial and temporal indices as the current array indices.
3: Store the redimensioned array into a new array.

Note that for data external to SciDB that is already indexed with STARE, it can be loaded and distributed directly without this redimensioning, a usually expensive step.

V. Result

In order to study the performance characteristics of our framework, we use a concrete set of typical queries in Earth Science domain. The queries operate on three multidimensional Grid and Swath datasets as described below. Since the handling of Point data is similar to that of Swath, it is not specifically demonstrated in this study.

A. Datasets

Two regular gridded datasets and one swath dataset for the period of Winter 2010 (i.e., from December 1st, 2009 to February 28th, 2010) are used to conduct our experiments. The first regular gridded dataset is extracted from an hourly dataset of the NASA Modern Era Retrospective-analysis for Research and Applications (MERRA-2) [13] data collection, while the second dataset is extracted from a reprocessed 5-minute National Mosaic and Multi-sensor QPE (NMQ, where QPE stands for quantitative precipitation estimate) [14]. MERRA-2 has global coverage, whereas NMQ is only available for the

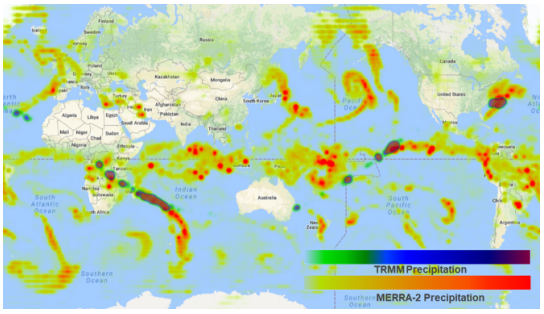


Fig. 8. Rendering of one time slice of MERRA-2 and a partial trajectory of TRMM.

contiguous United States (CONUS), specifically 20°N - 55°N in latitude and 130°W - 60°W in longitude. They are also of different resolutions. For MERRA-2, it is $\frac{5}{8}^\circ \times \frac{1}{2}^\circ$ (longitude $\times$ latitude) in space and hourly in time, whereas it is $0.01^\circ \times 0.01^\circ$ in space and every 5 minutes in time for NMQ.

The swath dataset, from NASA's Tropical Rainfall Measuring Mission (TRMM), derives vertical hydrometeor profiles using data from Precipitation Radar (PR) and TRMM Microwave Imager (TMI) that were orbiting the tropics of Earth (between 40°S and 40°N) $\sim$15 times per day generating a scan line every 600ms [15].

Three types of data are displayed within different ranges of location and they are not synchronized due to their different time intervals between contiguous time steps. Table III summarizes the characteristics of the three datasets.

B. Computing Environment

A cluster with 16 nodes is set up to carry out our experiments. All nodes have the same features: 32GB of main memory, an 8-core CPU and 9TB of local disk storage. They all run Centos 7 Linux operating system. The nodes are interconnected with 10 Gigabit Ethernet. We use the enterprise edition of SciDB release 16.9 that supports replication and advanced linear algebra operations, such as singular value decomposition (SVD).

C. Evaluation

1) Data placement: We first verify the data placement result of our approach. Figure 8 shows the rendering result that illustrates the distinct shapes and coverages of the MERRA-2 and TRMM datasets. Figure 9 visualizes the data placement of these two datasets on the 16 compute nodes using our STARE-based scheme and the conventional regular-grid partitioning and distribution scheme. From this qualitative comparison, we can clearly see that our solution can lead to co-alignment and co-location of the two datasets with completely different shapes and coverage. Next, we will show the quantitative evaluations to detail the advantages of our STARE indexing scheme and data placement strategy.

2) Join operation performance: There are two operators available for the join operation in SciDB: *cross-join* and *join*. The former is a generic join operator that makes no assumption about the schema of its array operands, while the latter requires

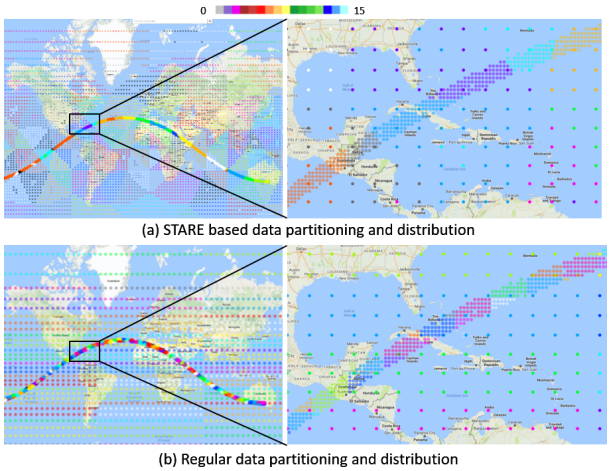


Fig. 9. Comparison between different data placements. (a): The top row shows the results using our STARE-based partitioning and distribution. Both the MERRA-2 and TRMM datasets are indexed using STARE (see Section III-A) and partitioned and distributed among 16 compute nodes using our round robin strategy (see Section III-B). Different nodes are denoted using 16 different colors in the top color map. The left image shows the partitioning and distribution of both datasets. The right image shows the zoom-in view in the highlight region (Gulf of Mexico and Caribbeans) of the left image. We can clearly see that the two datasets are well co-aligned and co-located among the nodes. (b): The bottom row shows the corresponding results using the regular-grid partitioning and distribution, which cannot co-align and co-locate the two datasets.

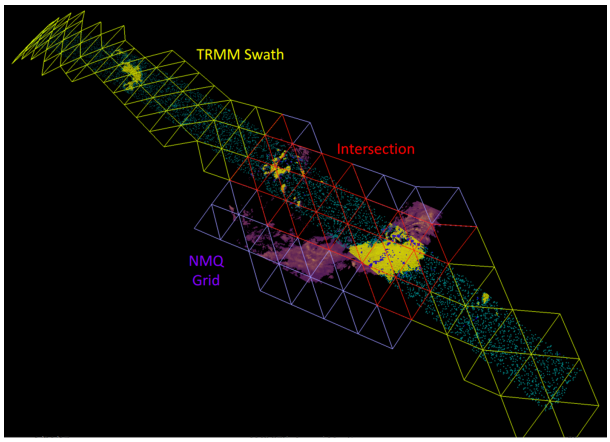
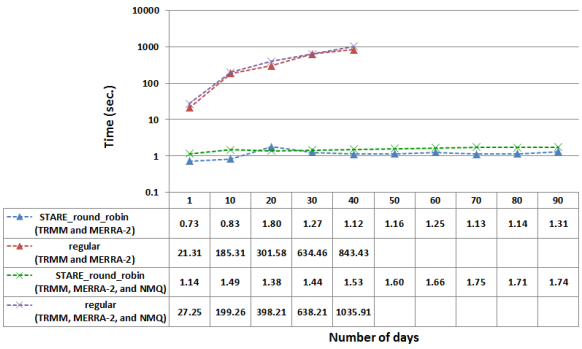


Fig. 10. After using STARE, we show the TRMM data in yellow meshes, NMQ data in purple meshes, and the intersection part in red triangle meshes.

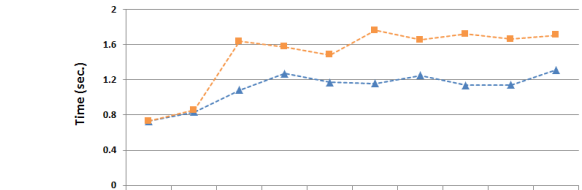
its array operands to be aligned (i.e., corresponding chunks co-located on the same nodes). Figure 10 illustrates the join operation between the TRMM swath and the NMQ grid. The yellow triangle mesh encodes the TRMM swath data, the purple triangle mesh encodes the NMQ grid, and the red mesh indicates the joined intersection.

We compare the timing results of the join operation between STARE-based scheme and the regular-grid partitioning and distribution scheme using the MERRA-2, TRMM, and NMQ datasets. With STARE, because all three datasets are co-aligned, we can directly conduct join queries for any combination of these datasets. Without STARE, the three datasets are misaligned, and cross-join is thus required for regular grid



	1	10	20	30	40	50	60	70	80	90
STARE_round_robin (TRMM and MERRA-2)	0.73	0.83	1.80	1.27	1.12	1.16	1.25	1.13	1.14	1.31
regular (TRMM and MERRA-2)	21.31	185.31	301.58	634.46	843.43					
STARE_round_robin (TRMM, MERRA-2, and NMQ)	1.14	1.49	1.38	1.44	1.53	1.60	1.66	1.75	1.71	1.74
regular (TRMM, MERRA-2, and NMQ)	27.25	199.26	398.21	638.21	1035.91					

Fig. 11. Comparison of timing results of joining operation between STARE and regular gridding using all the time slices within different numbers of days. The time axis is plotted in a logarithmic scale. The empty entries in the table are the data points that we did not get for regular gridding due to its long running time.



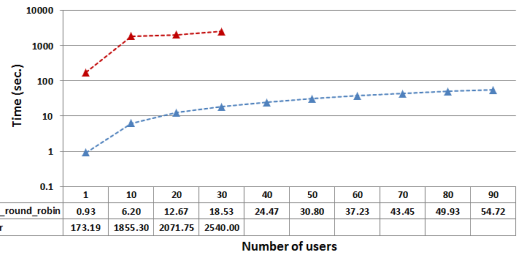
	1	10	20	30	40	50	60	70	80	90
STARE_round_robin	0.73	0.83	1.08	1.27	1.17	1.16	1.25	1.14	1.14	1.31
STARE_contiguous	0.73	0.85	1.64	1.57	1.49	1.76	1.65	1.72	1.66	1.71

Fig. 12. Comparison of timing results of joining operation between contiguous and round robin strategies for STARE using all the time slices of the TRMM and MERRA-2 datasets within different numbers of days.

partitioning.

Figure 11 shows the timing results of join operation using STARE and regular gridding. We first query the MERRA-2 grid cells where the precipitation rates according to TRMM are greater than 0.7 mm/hr for all the time steps within different numbers of days. The blue curve in Figure 11 shows that, with STARE, the time increases slowly and it only costs around 1.31 seconds for all the time slices within 90 days. The effect of data placement alignment afforded by STARE is plainly visible. As indicated by the red curve in Figure 11, without co-alignment before the query, the time has almost a linear increase. It costs around 843.43 seconds for a 40-day duration, beyond which the queries cannot complete in reasonable amount of time and are thus terminated. This is due to the fact that misaligned data chunks, as evidenced by the images in the bottom row of Figure 9, incur unnecessary communications.

The green and purple curves in Figure 11 show the comparison of the timing results for the join operation between STARE and regular partitioning methods using three datasets, i.e., MERRA-2, TRMM, and NMQ. Even with three datasets, the execution time of our STARE-based method only shows marginal increases relative to that with two datasets, further demonstrating the value of STARE in assuring data placement alignment. The timing results of the regular-grid method are significantly, but unsurprisingly, worse.



	1	10	20	30	40	50	60	70	80	90
STARE_round_robin	0.93	6.20	12.67	18.53	24.47	30.80	37.23	43.45	49.93	54.72
regular	173.19	1855.30	2071.75	2540.00						

Fig. 13. Comparison of timing results of multiple users' random join operations between STARE and regular gridding using the TRMM and MERRA-2 datasets for different numbers of users. The time axis is plotted in a logarithmic scale. The empty entries in the table are the data points that we did not get for regular gridding due to its long running time.

As discussed in Section III-B, we adopted the round robin strategy to traverse the STARE quadtree for partitioning, rather than the contiguous strategy. Figure 12 shows the comparison between these two strategies. The round robin strategy clearly outperforms the contiguous strategy. For example, more than 51% improvement has been achieved using the round robin strategy in a 50-day query. This performance gain is essential for supporting highly interactive analytics operations.

3) Multiple user queries: We also investigate the impact of multiple simultaneous users on STARE and regular-grid partitioning methods. We test different numbers of simultaneous users, and use random join operations of the TRMM and MERRA-2 datasets for a 10-day duration to simulate different user queries. As shown in Figure 13, again, only a marginal increase is observed for the STARE results from 1 to 100 simultaneous users. This further shows the effectiveness of our method with distinct operations across multiple users, achieved by the careful consideration of user data access patterns in our design (see Section III-B).

In contrast, the regular-grid partitioning cannot effectively support multiple users operations. As shown in Figure 13, system performance is quickly degraded with an increasing number of users. At 30 users, it took around 42 minutes for the system to respond, making interactivity unpractical and weakening user experience.

D. User Interface

To evaluate our system design for interactive analysis, we have constructed a high-level graphical query interface featuring multiple visual analytics capabilities, as shown in Figure 14. Our rudimentary user interface supports three essential functions: visualization, query, and statistical analysis.

First, all the datasets can be visualized in the main window. We provide a time slider at the bottom of the user interface to allow users choose any time slice and show the rendering results of the datasets. As shown in Figure 14, the TRMM precipitation dataset is shown in yellow to red, which looks like a ribbon across the map, while the NMQ precipitation dataset, only available for the contiguous United States (CONUS), is shown in green to purple. There are four buttons to the left of the time slider with the functions of

choosing different datasets, changing opacity, playing the time series visualization, and stopping the playing, respectively.

Second, users also can input their customized queries, such as Precip $\geq$ 1.0, x $\geq$ 800 and y $\leq$ 3600, in the search box at the top of the user interface to easily search for features of interest across multiple heterogeneous datasets.

Third, our user interface supports a few real-time statistical analytics (such as histogram and correlation). Figure 14 shows two examples. Users can interactively drag a rectangle to select their region of interest on the map. The corresponding histogram of each dataset will be displayed in the top-right analytics panel. Users can also put a marker on a specific location on the map to display time series of multiple datasets at this location in the analytics panel. Moreover, users can also download these time series values as a csv (comma-separated values) file by clicking the download button.

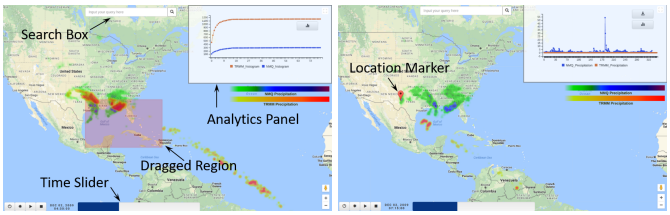


Fig. 14. The user interface of our system.

Our STARE indexing scheme and the data placement strategy ensure the scalability of various operations with possible combinations of variables, time slices, geolocations, datasets, and user numbers, to deliver interactive performance to each individual end user, rendering our user interface a practical and effective analytics tool for Big Earth Data.

VI. Conclusion

Big Earth Data imposes grand challenges in the development of a scalable end-to-end data analytics system. Although previous studies have proposed feasible solutions to address the large volume challenge by mostly leveraging loosely coupled techniques, scalable performance is still difficult to achieve when diverse large datasets are involved. We must holistically consider both the volume and variety challenges and address both the data co-location and co-alignment in a distributed environment. To this end, we present the SpatioTemporal Adaptive-Resolution Encoding, STARE, for uniform indexing of diverse, heterogeneous Earth datasets and, thus, for spatiotemporal co-alignment and co-location of data chunks. In addition, we partition and distribute the chunks along the traversal of the resulting STARE quadtree leaves in a round robin fashion, leading to balanced workload among the processors. We conduct an extensive experimental study by comparing our STARE-based approach with the conventional regular gridding based approach. We demonstrate the effectiveness of our approach using queries into different combinations of variables, time slices, geolocations, datasets, and user numbers. Our end-to-end distributed storage and analytics system is able to achieve scalable performance.

For future works, we hope to enhance our system performance using Graphics Processing Units (GPUs). More sophisticated visual analytics tasks, such as feature extraction and tracking and volume rendering, may then be carried out interactively by leveraging GPUs. We plan to investigate effect of complex data access patterns that are expected to exist across the storage hierarchy from persistent storage to main memory to GPU memory, and study the possibility [16] of extending STARE to indexing data among multiple GPUs.

Acknowledgment

This research has been sponsored in part by the National Science Foundation through grants ICER-1541043, ICER-1540542, and IIS-1423487 and with supplemental funding from the Advanced Information Systems Technology (AIST) program of NASA Earth Science Technology Office.

References

[1] M. L. Rilee, K.-S. Kuo, T. Clune, A. Oloso, P. G. Brown, and H. Yu, "Addressing the big-earth-data variety challenge with the hierarchical triangular mesh," in *Big Data (Big Data), 2016 IEEE International Conference on*. IEEE, 2016, pp. 1006–1011.

[2] L. Yu, H. Yu, H. Jiang, and J. Wang, "An application-aware data replacement policy for interactive large-scale scientific visualization," in *Parallel and Distributed Processing Symposium Workshops (IPDPSW), 2017 IEEE International*. IEEE, 2017, pp. 1216–1225.

[3] J. Dean and S. Ghemawat, "MapReduce: simplified data processing on large clusters," *Communications of the ACM*, vol. 51, no. 1, pp. 107–113, 2008.

[4] M. Zaharia, M. Chowdhury, M. J. Franklin, S. Shenker, and I. Stoica, "Spark: Cluster computing with working sets." *HotCloud*, vol. 10, no. 10-10, p. 95, 2010.

[5] D. Borthakur *et al.*, "HDFS architecture guide," *Hadoop Apache Project*, vol. 53, 2008.

[6] A. Lakshman and P. Malik, "Cassandra: a decentralized structured storage system," *ACM SIGOPS Operating Systems Review*, vol. 44, no. 2, pp. 35–40, 2010.

[7] T. White, *Hadoop: The definitive guide*. O'Reilly Media, Inc., 2012.

[8] S. Zhou, X. Li, T. Matsui, and W. Tao, "Visualization and diagnosis of earth science data through Hadoop and Spark," in *Big Data (Big Data), 2016 IEEE International Conference on*. IEEE, 2016, pp. 2974–2980.

[9] P. Z. Kunszt, A. S. Szalay, and A. R. Thakar, "The hierarchical triangular mesh," in *Mining the sky*. Springer, 2001, pp. 631–637.

[10] The Sloan Digital Sky Survey and HTM websites: http://skyserver.sdss.org/ and http://www.skyserver.org/htm/ .

[11] M. Stonebraker, P. Brown, D. Zhang, and J. Becla, "SciDB: A database management system for applications with complex analytics," *Computing in Science & Engineering*, vol. 15, no. 3, pp. 54–62, 2013.

[12] K. Doan, A. O. Oloso, K.-S. Kuo, T. L. Clune, H. Yu, B. Nelson, and J. Zhang, "Evaluating the impact of data placement to Spark and SciDB with an earth science use case," in *Big Data (Big Data), 2016 IEEE International Conference on*. IEEE, 2016, pp. 341–346.

[13] M. Bosilovich, R. Lucchesi, and M. Suarez. (2016) MERRA-2: file specification. GMAO Office Note No. 9 (Version 1.1), 73 pp, available from http://gmao.gsfc.nasa.gov/pubs/office_notes.

[14] J. Zhang, K. Howard, C. Langston, S. Vasiloff, B. Kaney, A. Arthur, S. Van Cooten, K. Kelleher, D. Kitzmiller, F. Ding *et al.*, "National Mosaic and Multi-Sensor QPE (NMQ) system: Description, results, and future plans," *Bulletin of the American Meteorological Society*, vol. 92, no. 10, pp. 1321–1338, 2011.

[15] C. Kummerow, W. Barnes, T. Kozu, J. Shiue, and J. Simpson, "The tropical rainfall measuring mission (TRMM) sensor package," *Journal of atmospheric and oceanic technology*, vol. 15, no. 3, pp. 809–817, 1998.

[16] J. Xie, H. Yu, and K.-L. Maz, "Visualizing large 3D geodesic grid data with massively distributed GPUs," in *Large Data Analysis and Visualization (LDAV), 2014 IEEE 4th Symposium on*. IEEE, 2014, pp. 3–10.