Addressing the Big-Earth-Data Variety Challenge with the Hierarchical Triangular Mesh

Michael L. Rilee[1,2], Kwo-Sen Kuo[1,3,4]
Thomas Clune[1], Amidu Oloso[1,5]
[1]NASA GSFC, Greenbelt, MD, USA,
[2]Rilee Systems Technologies, Derwood, MD, USA
[3]Bayesics, LLC, Bowie, MD, USA
[4]University of Maryland, College Park, MD, USA
[5]SSAI, Greenbelt MD, USA
mike@rilee.net
{amidu.o.oloso, kwo-sen.kuo, thomas.l.clune}@nasa.gov

Paul G. Brown
Paradigm4 Inc.
Waltham, MA, USA
pbrown@paradigm4.com

Hongfeng Yu
University of Nebraska, Lincoln, NE, USA
hfyu@unl.edu

***Abstract*—We have implemented an updated Hierarchical Triangular Mesh (HTM) as the basis for a unified data model and an indexing scheme for geoscience data to address the variety challenge of Big Earth Data. In the absence of variety, the volume challenge of Big Data is relatively easily addressable with parallel processing. The more important challenge in achieving optimal value with a Big Data solution for Earth Science (ES) data analysis, however, is being able to achieve good scalability with variety. With HTM unifying at least the three popular data models, i.e. Grid, Swath, and Point, used by current ES data products, data preparation time for integrative analysis of diverse datasets can be drastically reduced and better variety scaling can be achieved. HTM is also an indexing scheme, and when applied to all ES datasets, data placement alignment (or co-location) on the shared nothing architecture, which most Big Data systems are based on, is guaranteed and better performance is ensured. With HTM most geospatial set operations become integer interval operations with further performance advantages.**

Keywords-SciDB; HTM; array database; shared nothing architecture; variety; DAAC; remote sensing; data analysis; data fusion; load balancing; indexing; geographic metadata; GIS

I. INTRODUCTION

For decades, Earth Science (ES) data practice has followed a two-step approach: 1) package data into files for archival and distribution and 2) catalog metadata of the datasets and files into databases managed by relational database management systems (RDBMSs), making holdings discoverable and searchable. The establishment of the popular distributed active archive centers (DAACs) as data warehouses and the standardization of data file formats through the HDF/netCDF [1] Application Programming Interface (API) by NASA Earth Observing System Data Information System (EOSDIS) since the 1990s exemplify this approach, a de facto standard.

Despite its success the two-step approach has reached its limit with today's demands. Users cannot manipulate the data directly but must work through their containers (i.e., files) requiring specialized expertise and resources beyond mere scientific analysis, inevitably leading to data download (via low-bandwidth Internet links) and integration by users.

Lacking "in place" access, users use FTP links produced by metadata searches to download data files before starting analysis. To prepare, users (or their institutions) must first procure compute and storage resources, including associated management and maintenance. Researchers must also engage in data management activities, such as data organization and backup, and familiarize themselves with the data's structure and semantics. Largely irrelevant to research, these tasks unnecessarily encumber researchers, hampering productivity. The most disheartening aspect is the collective waste, since almost every data analysis research endeavor needs to duplicate this process and the resources it requires.

Downloaded data become "local," subject to end users' preferences, e.g. researchers' data management policies & programming language choices. The profusion of these preferences erects expensive barriers to collaboration. Moreover, most geoscience researchers are not professional software engineers, rarely following software engineering practices, e.g. unit testing and source code version control. Thus software quality varies and reproducibility becomes illusive.

A. Volume vs. Variety

Since supercomputing centers have achieved remarkable success in maximizing value for simulations, it is natural that the first attempts to address the above data analysis challenges leverage existing infrastructure. These solutions move data to dedicated, expensive, supercomputing file systems, where compute resources are close to the data.

Although good scaling in *volume* can be achieved with this approach, *variety* is a different story. The variety of ES data arises mostly from diversity of observations. "In situ" and "remotely sensed" are two main categories of ES data. By location, these subcategories subdivide into ground-based (e.g., weather radars), airborne, and space-based. By instrumentation operation mode, remote sensing subdivides into passive sensors (e.g., radiometer and imager) and active sen-

978-1-4673-9005-7/16/$31.00 ©2016 IEEE

sors (e.g., radar and lidar). Position, purpose, and physical or practical constraints often drive different sensing geometries with different spatial and temporal resolutions. Further processing yields a greater variety of data products.

B. *Variety's Toll on Iterative Analysis*

When preparing to integrate diverse datasets, using a supercomputing facility does not substantially reduce the effort required, because each file in each variety has to be processed and homogenized before integration. Unless the "fused" data of the integrative analysis is saved and shared, anyone who wishes to perform the same analysis will have to repeat the data preparation process. If a similar, but not exactly the same, integrative analysis is to be performed, e.g. using different spatiotemporal subsets or slightly different data products, data preparation generally has to be redone; prior results and intermediate products can scarcely be reused. Since scientific analysis is continuous reanalysis, data preparation costs exert an expensive toll at every iteration.

C. *In-place analysis via Partition Placement Co-alignment*

Big Data technologies enable in-place analysis, including the creation and sharing of data products, yet ES data places special demands on computing architectures, like *spatiotemporal coincidence*. For example, to analyze cloud formation, we need spatiotemporally coincident data, such as temperature, humidity, airflow, etc. Yet for distributed arrays of ES data with misaligned partitions, an integrative analysis has to perform computationally expensive data repartitionings on the fly [2], degrading performance. Better performance can be achieved by co-locating ES data partitions on a shared-nothing architecture (SNA), reducing unnecessary movements. Currently, end-users move and integrate data themselves, incurring the problems mentioned above. Partition Placement Co-alignment (PPC) is required for in-place analysis on distributed architectures, especially when transfer and integration is costly, further motivating SNA.

D. *A Unifying Approach*

We are developing an in-place analysis system for ES data analysis and sharing, building on SciDB, an analysis environment that scales to the massive, distributed, parallel systems required to integrate and analyze diverse ES datasets [3,4]. SciDB is based on the SNA and is exceptionally suited for pleasingly parallel computations (those with minimal inter-process communication). Inter-process communication here is synonymous to data movement within the SNA. The ideal PPC makes analyses as pleasingly parallel as possible. SciDB, as mentioned above, also supports tightly-coupled scientific calculations that are not pleasingly parallel.

Fundamental to ES integration and analysis, we are adding geometric functions to the SciDB array database and analysis platform [5]. We build on the work of Szalay et al. who developed the Hierarchical Triangular Mesh (HTM) to index the Sloan Digital Sky Survey (SDSS) [6,7]. HTM's efficient indexing and fast integer representation provides a common geographical reference for dataset integration.

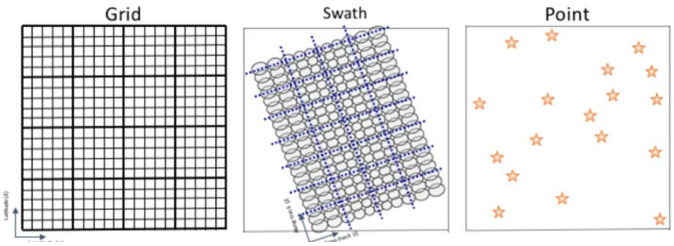

Figure 1. Three data models.

HTM's quad-tree-based scheme recursively subdivides triangles into four children by bisecting parents' edges, starting with a root spherical octahedron. This maps the sphere to integer intervals for search, intersection, data fusion, co-registration, among other tasks. It is especially leveraged here for PPC.

In this work we describe a new "Left Justified" format more appropriate to indexing and integrating distributed ES data. We discuss how our new hierarchical labeling aids the efficient distribution of data across computational nodes. Temporal indexing and other forms of variety are important but relatively simpler and beyond the scope of this paper.

II. Dealing with Diversity: Point, Swath, & Grid

Three data models can generally represent the spatial variety of geoscience data: Grid, Swath, and Point (Fig. 1). Grid is a mesh with fixed latitude and longitude spacing, with a simple linear relation between array indices and latitude-longitude geolocation coordinates. Swath retains a spaceborne instrument's observation geometry (e.g., cross-track × along-track) for its Instantaneous Field 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. For Point and Swath, geolocations of data elements are specified individually.

Dissimilarities among these models hamper integrative analysis. For example, simply determining the common area covered by two Swath arrays can be complex, especially if the satellites have different orbit characteristics. These data models, however, have one commonality: geolocations, which can serve as a basis for a unified data model. The heart of our unified model is thus an indexing scheme that assigns an "address" (index) to every surface element (up to a desired resolution) of Earth (i.e., geolocation), e.g. Fig. 2. Arrays indexed by this address allow quick retrieval of data associated with any geolocation, regardless of their original data models. HTM speeds the comparison and integration of data with different geometries (Fig. 3).

III. Hierarchical Spherical Triangular Mesh

The HTM is based on the recursive quadfurcation of a root spherical octahedron [5-13]. South (North) is labeled with a 0 (1), Fig. 2. The 4 triangles of each half are labeled 0-3 starting from the triangle nearest the x-axis proceeding counterclockwise around the sphere as viewed from outside and above the poles. At each following level of recursion, the 4

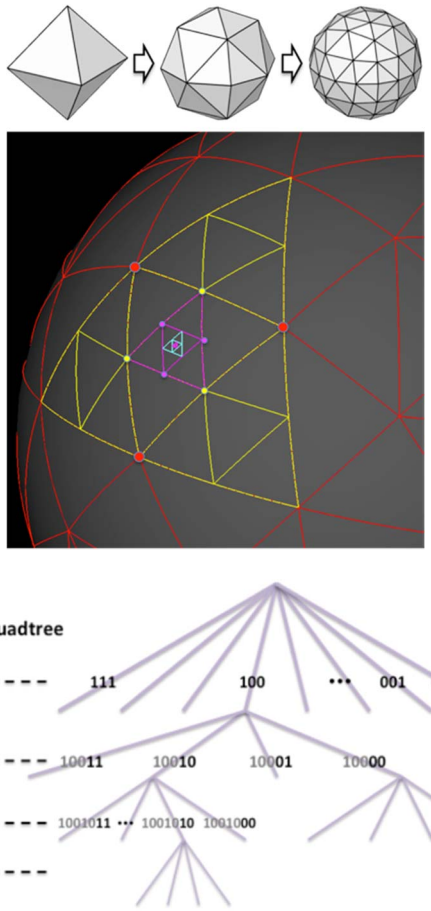

Figure 2. HTM: Projecting the octagon onto and recursively partitioning the sphere into a quad-tree, whose branches are labeled in binary.

child triangles constructed by adding a triangle connecting the midpoints of an existing triangle's edges are labeled 0-3 according to order in which the parent's vertices are stored. Each symbolic digit describes how to traverse the quad-tree structure from the root octahedron to the leaf triangle. For example, the 2nd child (a grandchild) of the 1st child of the first (0th) triangle counterclockwise from the x-axis in the northern hemisphere would be denoted N012, a level 2 triangle; examples are illustrated in Fig. 4.

A. Mapping HTM indices to integers

As pointed out by Gray, a triangle with an index contains all of the points inside it, completely covering that triangle [8]. One can map HTM symbolic names (and their triangular regions) to integers in a number of ways. With the symbolic form described above, triangles with the same "prefix" are children of the triangle denoted by that prefix. For example (Fig. 4), with their shared prefix in boldface, the triangles

N0123123 and
N0123333

are both contained in the triangle N0123.

Note triangles contained in a parent are readily labeled using the parent's representation as a prefix. For example

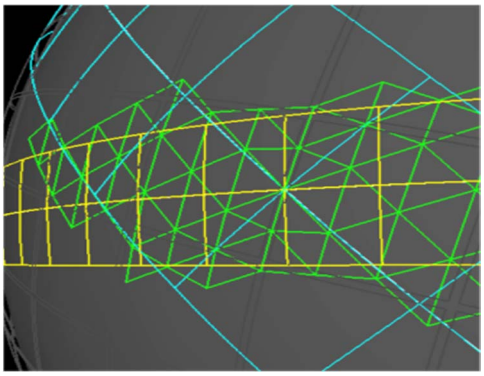

Figure 3. Finding spatial intersections using the HTM.

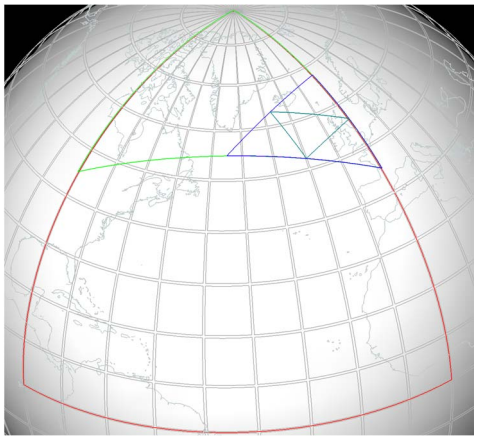

Fig. 4. Triangles N0-red, N01-green, N012-purple, N0123-cyan.

all level 6 children of the level 3 triangle N0123 are in the range `N0123000 - N0123333.` Regions on the sphere may be covered by sets of these HTM indexed triangles. Encoded as integers, sequences of HTM indices corresponding to regions can be replaced by integer intervals, such as in the previous example. Searching and calculations such as dataset intersections are made more efficient by substituting integer operations for trigonometry & 3D vector math. This reduction in effort and storage enables detailed geometric metadata, accelerating dataset integration, the crucial reason we add HTM to SciDB.

B. Right Justified HTM integer index

The SDSS/HTM implementation had a simple map from symbols to integers. For them, zero, 0x0 in hexadecimal, is the invalid HTM index. For valid indices, a top bit was set and succeeding bits were set according to the symbolic representation with S (N) being represented by 0 (1). With 0b indicating a binary representation, we have the translation of symbols to integers in a Right Justified Mapping (RJM).

```
 S0123  -> 0b1000011011    = 0x21b = 539
 N0123  -> 0b1100011011    = 0x31b = 795
 S01230 -> 0b100001101100 = 0x86c = 2156
```

Unfortunately, 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 implement-

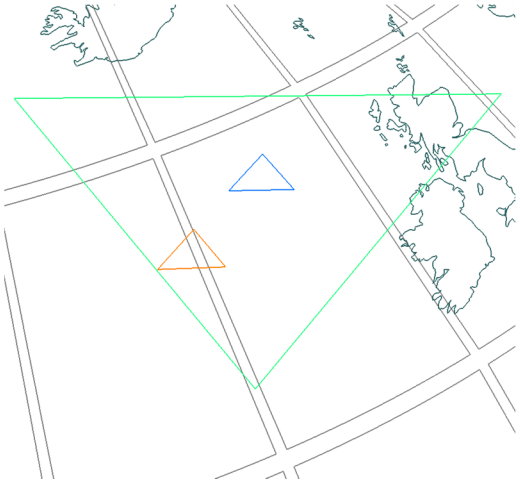

Figure 5. Triangles sharing a parent's prefix. Here N0123 (green) has children N0123123 (orange), and N0123333 (blue).

ing 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. Geometrically S0123 (corresponding to the digital value 539 RJM) contains S01230 (2156 RJM), but that when mapped to integers N0123 (795 RJM) it lies in between, even though S0123 and S01230 share the same prefix to the 3rd level (Figs. 5). 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.

For example, a data set with 5-km footprints might be matched by level 11 HTM triangles, while a 150-km footprint might get by with level 6. Combined groups of measurements, e.g. data swaths as opposed to individual IFOV measurements, are more important for co-location and re-gridding over geographic regions, so we use triangles from multiple levels of HTM to more succinctly approximate and index these regions, with compact integer intervals (Fig. 6).

C. Left Justified Integer Intervals

If indexing were our only goal, we could live with RJM, though it complicates some integer calculations. Yet for data locality and distributed processing RJM doesn't work.

A more convenient mapping that allows other tools to take advantage of HTM geometry using integer operations without falling back on HTM's floating-point geometry is to use a Left Justified Mapping (LJM) bit format. In LJM our example becomes (using 12-bits for clarity):

```
S0123 -> 0b100001101100 = 0x86c = 2156
N0123 -> 0b110001101100 = 0xc6c = 3180
S01230-> 0b100001101100 = 0x86c = 2156
```

With LJM the common HTM prefix is maps to integers in the same way at different HTM levels, so now geometric containment is respected by the mapping. Unfortunately, as demonstrated in the above example, we now have an aliasing problem, e.g. with S0123 and S01230. LJM as stated doesn't distinguish between levels, because it doesn't track how many bits from the left are significant. For the RJM, the top or depth bit encodes both the level of the HTM index and

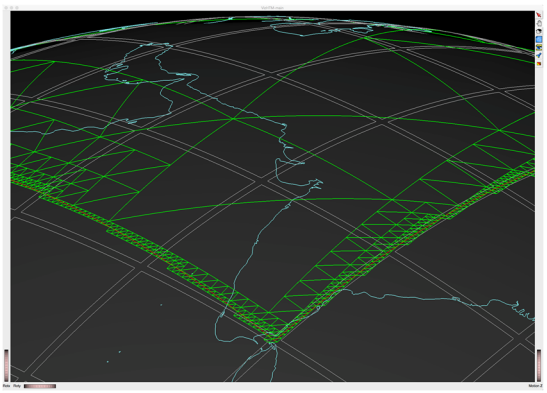

Figure 6. Approximating a region with triangles at multiple resolution levels.

where the significant bits start. LJM needs three more things.

First, we need to keep track of the level. We confine ourselves to signed 64-bit integers for technical reasons and so that other tools may take advantage of our implicit geometric encoding. To track the quadfurcation level we devote the rightmost (least significant) 6 bits, using the rightmost 5 for the actual level number and the remaining bit (the 6th) in reserve. We reserve the leftmost bit for internal use and set the remaining bits as in the left justified example above, dropping the top/depth bit as being unnecessary.

Table 1. Left Justified Mapping

Bit position	Use
most significant 63	Reserved // Top Bit
62	North-South Bit
60..61	Octahedral triangle index Resolution level 0 ~10,000 km
6..59	Quadtree triangle index Resolution levels 1-27 ~5,000 km to ~7 cm
5	Reserved // Terminator Bit
0..4 least significant	Resolution level // Terminator

Thus the triangles from our previous example become:

```
S0123   -> 0x06c0000000000003
S01230  -> 0x06c0000000000004
N0123   -> 0x46c0000000000003
```

This encodes the difference between S0123 and S01230 and integer comparisons can tell us that the latter is contained in the former but cannot distinguish the reverse. Therefore, the inclusion of levels respects the underlying HTM geometry. The difference between these representations of S0123 and S01230 is their levels. Since the maximum resolution level is 27 for our 64 bit encoding (see Table 1 above), for any two HTM integers that differ by 27 or less, the triangle associated with the lesser will contain the triangle of the greater level.

For the second step, note there are natural upper and lower bounds to the set of all labeled child triangles within a given triangle. For the lower bound one merely takes the HTM index of that given triangle as a prefix and then ap-

pends zeros down to the maximum allowed resolution of the representation (excluding the 6 bits reserved for the quadfurcation level), i.e. one needn't change the current representation. Consider the following consecutive level 3 triangles indexed and mapped as follows.

```
lower bound S0123 0x06c0000000000003
upper bound S0130 0x0700000000000003 faulty
```

Ignoring the 6 bits reserved for the quadfurcation level, the numbers between these two limits correspond to all of the triangles in S0123, i.e. traversals of the HTM quadtree from that triangle, representable in our 64-bit LJM. Using S0130 as an upper bound for triangles in S0123 is problematic, because of the aliasing problem. For performance, integer order operations, e.g. "<" or "<=", should replace geometric. However, the faulty upper boundary in LJM above requires care, because one has valid HTM integers less than upper bound S0130, but still not in S0123, e.g. S013 at level 2.

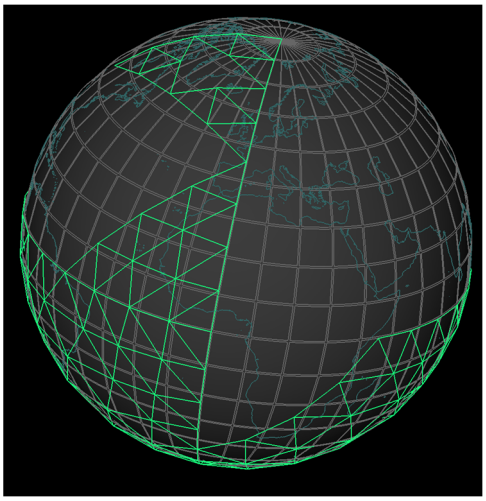

Figure 7. Level 3 interval S0123..N0123 with multiple triangles and a hole, the larger level 2 triangle N013 near the pole.

The third thing to do is thus to address the upper bound: if we used the encoding scheme discussed above, drawing a correspondence between integer interval operations and geometric set operations becomes troublesome. Some of these problems are lessened if we use the last, smallest indexed triangle as the upper bound, which are as follows.

```
upper bound S0123  -> 0x06c3fffffffffc3
upper bound S01230 -> 0x06c3fffffffffc4
```

Where we have essentially selected the 3rd triangle at each quadfurcation (adding ones at each bit position down to the rightmost 6 bits). However, this representation again confuses the logic associated with determining inclusion at the upper bound. One could mask off the level bits or make use of the unused 6th bit, but it is easier to introduce a terminator for the interval by simply setting all bits to the right of the significant HTM location bits to one. For a 6-bit field, this corresponds to the number 63, 0x3F. Level information is already contained in the lower bound for an interval and is thus redundant in the upper bound. Therefore, with intervals, the above examples become:

```
S0123  0x06c0000000000003-0x06c3fffffffffff
S01230 0x06c0000000000004-0x06c3fffffffffff
N0123  0x46c0000000000003-0x46c3fffffffffff
```

With this mapping the integer interval mappings of all of the triangles in S01230 fall within the interval `0x06c0000000000004-0x06c3fffffffffff`, and intersections can be performed with integer operations.

If we were only dealing with individual triangles, we needn't explicitly save the terminator, but the terminator is important when more than one triangle or level is concerned. An interval including multiple triangles at the same level is

```
S0123-N0123
      0x06c0000000000003-0x46c3fffffffffff
```

corresponding to a complex region on the sphere (Fig. 7). Note that all of these triangles in the interval are at the same quadfurcation level. When triangles at different levels are combined in one set of intervals, one must fix how one handles areas of overlap. For our searching and co-registration needs, subsuming higher resolution triangles into overlapping lower resolution intervals is appropriate, though it adds the complexity of editing the higher resolution intervals when intersections occur. E.g. in Figs. 4 and 5 the smaller triangles would be subsumed in their parents.

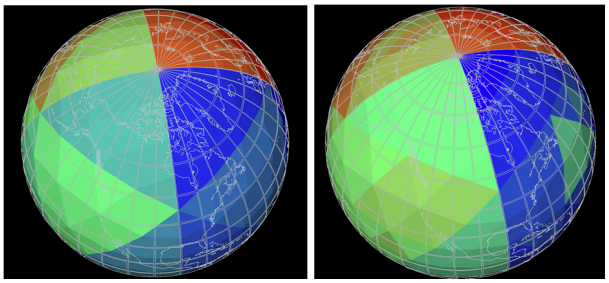

Figure 8. Automatic chunking/load-balancing based on HTM integer intervals. Left: homogeneous. Right: non-uniform.

The mapping from HTM to integer intervals described above tends to map nearby portions of the sphere to nearby portions of the number line. Since the indexing is based on a quadratic tree, it is possible for neighboring leaf nodes (triangles) to only have the root as a shared parent. In this case, neighboring triangles (in a geometric sense) correspond to integers that are rather far apart (in a numeric sense).

IV. Applying the HTM on SciDB

The left justified interval mapping plays a critical role in adding geospatial capabilities to SciDB, a shared-nothing array database system that works most efficiently when data communication from node to node is minimized [14]. The LJM provides a natural way to partition the sphere and distribute data across computational nodes (Fig. 8). HTM-based metadata allows datasets to be partitioned and distributed to nodes, each of which has its own index intervals. Communications are minimized because geometrically local operations are kept local to a node. Linking geometric and computational partitioning is essential for efficiently using *in place* diverse, large scale Earth Science data in SciDB.

We added our updated HTM to SciDB using its User Defined Type and Function (UDT and UDF) facilities and have constructed an HTM UDT verifying functions for constructing SciDB indices. The SDSS/HTM-based code has an efficient skip-list-based implementation of HTM integer sets for handling complex geometric regions [15]. We plan to apply HTM-based SciDB geometric functions on full-scale diverse Earth Science data to automatically identify spatio-temporally extended weather events, starting with blizzards. Tagging Earth Science data with HTM ranges, sets of our left justified integer intervals, will allow the geometric comparison, co-registration, and selection of diverse kinds of data via efficient metadata operations.

Researchers can rely on HTM indexing and SciDB to automate integration and fusion of diverse data geometries, including the Earth Science community standards such as the various geodesic and CubeSphere grids. This foundation will allow analyses to adapt to the intrinsic geometry of the data, whether gridded, unstructured, or swath, providing the means to transform the data to geometries that suit scientific goals. HTM does not suffer the extreme singularities and distortions of the popular lat-lon grids but also avoids the computational complexity of more uniform meshes [16-18]. By maintaining a close connection to efficient spherical geometric constructs (esp. great circles), complex regions are more efficiently and effectively characterized, requiring less storage and computation, when applied as an organizing metadata scheme. Fast and compact geometric metadata that *organizes ES data across distributed computing resources is critical* for eliminating the current wasteful and error-prone file-based end-user data preparation.

SciDB provides a compute and storage paradigm better suited to the large-scale distributed data intensive Earth Science than the traditional 2-step of download and integrate, even when supported by supercomputing centers. The new HTM with its left justified bit format provides an efficient uniform geographic platform for integrating diverse datasets, enables efficient implementation of the shared nothing architecture via data partition placement co-alignment, and provides a compact representation for geographic metadata. Thus SciDB and the new HTM point the way towards *an in place* data analysis environment that scales to the current Big Earth Science Data analysis variety challenge.

V. ACKNOWLEDGMENTS

Support for this work is primarily provided by the NASA Earth Science Technology Office's Advanced Information System Technology program and partially by the National Science Foundation's EarthCube program.

REFERENCES

[1] Network Common Data Form NetCDF: http://www.unidata.ucar.edu/software/netcdf/; The HDF Group, http://hdfgroup.org

[2] Clune, T., K.-S. Kuo, K. Doan, and A. Oloso. 2015. "SciDB versus Spark: A prel. comparison based on an Earth science use case," in *AGU Fall Meeting*, San Francisco, CA, 2015.

[3] SciDB by Paradigm4: www.paradigm4.com; J. Rogers et al. Overview of SciDB: Large scale array storage, proc. and analysis. In SIGMOD, 2010.

[4] Brown, P. 2015. SciDB and Geoinformatics Analysis. Geophysical Research Abstracts, Vol. 17, EGU2015-15102, 2015. EGU General Assembly 2015

[5] The Hierarchical Spherical Triangular Mesh (HSTM) website: https://github.com/michaelleerilee/hstm

[6] Szalay, A.S., Gray, J., Fekete, G., Kunszt, P.Z., Kukol, P., and Thakar, A. 2005. Indexing the Sphere with the Hierarchical Triangular Mesh. Micr. Res. Tech. Rpt., MSR-TR-2005-123.

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

[8] Gray, J., Szalay, A.S., Fekete, G., O'Mullane, W., Santisteban, M.A.N., Thakar, A., Heber, G., Rots, A.H. 2004. There Goes the Neighborhood: Rel. Algebra for Spatial Data Search. Microsoft Research Tech. Rpt., MSR-TR-2004-32.

[9] Fekete, G., Kuo, K.-S. 2015. Index. Earth with Trixels. Poster pres. at the 8th XLDB Conf., May 19-20 Stanford Univ., CA

[10] Kunszt, P.Z., Szalay, A.S., Thakar, A.R. 2001. The Hierarchical Triangular Mesh. In Mining the Sky: Proceedings of the MPA/ESO/MPE Workshop held at Garching, Berlin/Heidelberg, 2001, Ch. 83, p631.

[11] Barrett, P. 1994. Application of the Linear Quadtree to Astronomical Databases, Poster at ADASS 1994.

[12] Fekete, G. 1990. Rend. and managing spherical data with sphere quadtrees. Proc. of Vis. '90. IEEE Comp. Soc., Los Alamitos, CA. pp. 176-186.

[13] Crichton, D.J., Mattmann, C.A., Cinquini, L., Braverman, A., Waliser, D., Gunson, M., Hart, A.F., Goodale, C.E., Lean, P., and Kim, J. 2012. Sharing Satellite Obs. with the Climate-Modeling Community: Software and Architecture. IEEE Software, September/October 2012, pp. 73-81.

[14] Doan, K., A. Oloso, K.-S. Kuo, T. Clune, H. Yu, B. Nelson, J. Zhang. 2016. Eval. the Impact of Data Placement to Spark and SciDB with an Earth Sci. Use Case. 2016 IEEE Int'l. Conf. on Big Data, Dec. 5-8, 2016, Washington, D.C.

[15] Pugh, William. 1990. Skip lists: a prob. alt. to balanced trees in Comm. of the ACM, June 1990, 33(6) 668-676.

[16] Song, L., A.J. Kimerling, and K. Sahr. 2002. Developing an Equal Area Global Grid by Small Circle Subdivision. In M. Goodchild and A.J. Kimerling (Eds.), Discrete Global Grids. Santa Barbara, CA, USA: Nat'l Ctr for Geo. Inf. & Anal.

[17] Budavári, T., A. Szalay, and G. Fekete. 2010. Searchable Sky Coverage of Astronomical Observations: Footprints and Exposures. Publications of the Astronomical Society of the Pacific, **122**:1375-1388, 2010 November.

[18] Kondor, D., L. Dobos, I. Csabai, A. Bodor, G. Vattay. 2014. Efficient classification of billions of points into complex geographic regions using hierarchical triangular mesh. arXiv:1410.0709v1 (cs.DB) Proc. of the 26th Int'l. Conf. on Sci. and Stat. Database Management (2014)