Vision of Data-Intensive Analysis Centers and the Challenges to Get There

Kwo-Sen Kuo, Thomas L Clune, Amidu O Oloso, Michael L Rilee, Khao Doan, Rahul Ramachandran

Preface

This is a more conceptual presentation about the motivation behind our AIST-14 project:

"Data Environment for Rapid Exploration and Characterization of Hydrometeorological Organized Systems", DERECHOS.

Ultimate Goal

Optimize research quality and productivity by

- Alleviating researchers from data management chores,
- Enabling researchers to perform integrative analysis involving diverse datasets,
- Providing improved means and assurance to software/research quality and traceability,
- Facilitating seamless cross-discipline and cross-institution collaborations, and
- Easing the implementation of automated knowledge extraction techniques.
  - Machine/Deep Learning

Vision

Inspired by the "Ultimate Goal" we seek to construct a system that allows researchers to

- Conduct (moderately complex) Earth Science data analysis with real-time response,
  - E.g. aggregate statistics, conditional subsetting, etc.
- Analyze diverse datasets in a uniform manner independent of data (file) format,
- Have better assurance in software quality and analysis/research reproducibility,
- Collaborate seamlessly with colleagues,
- Carry out machine learning without spending disproportional effort in preparing data, and
- Systemize machine learning to achieve deep learning.

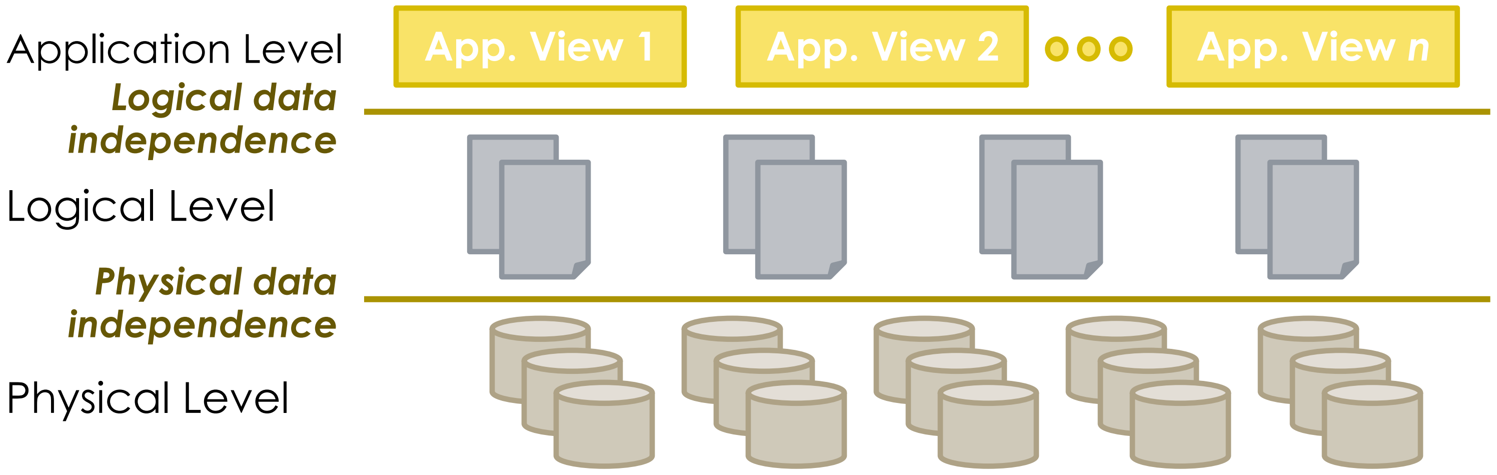
Big Data Challenges

- *Volume*
- *Variety*
- *Velocity*
- *Veracity*
- ***Value***

Scaling *Variety*

- Scalability –
  - The capacity to be changed in size or scale.
  - The ability of something, especially a computer system, to adapt to increased demands.
  - *Resources consumed is linearly (or better) proportional to the quantity and/or complexity involved.*
- Current efforts appear to focus mostly on scalability in *Volume*.
- We believe scalability in *Variety* is the key.
- **Data Independence** principle is crucial for scaling *Variety*.
  - Data independence is indispensable for data interoperability.

Data Independence in DBMSs



- Two levels of data independence in a database management system, DBMS.
  - *Physical data independence* – between physical representation and logical representation
  - *Logical data independence* – between logical representation and application views

Current File-based Practice

- Partial physical data independence is achieved with "standard file formats" via API, e.g. HDF, GRIB.
- There are at least three (3) logical representations: Grid, Swath, and Point.
- But, in reality, it is way more complicated than 3.
  - Non-uniform granularity.
  - Varied "profiles".

It is impossible to achieve "variety scalability" with the file-based approach!

Consequences of File-Based Practice

The file-based practice

- Necessitates download* for any extensive analysis,
- Induces duplications in compute and storage resources,
- Entails individualized data management and analysis algorithm development,
- Discourages software reuse and complicates quality assurance,
- Hampers software and research traceability, and
- Erects collaboration barriers.

Technical Challenges

- Choice of technology
  - Traditional HPC vs. shared-nothing architecture (SNA)
  - MapReduce/Hadoop, Spark, and SciDB
- Design issues
  - Data placement alignment
  - Unified data representation
  - Supportive transformations
    - Regridding/Remapping

SciDB

- SciDB, by Paradigm4, is an all-in-one data management and advanced analytics platform that features:
- Complex analytics inside a next-generation parallel **array** database (i.e. not row-based or column-based like RDBMS's based on table data model).
  - Supports extensive and flexible algebra operators that can be efficiently "wired" together for more complex operations.
- Based on the "shared nothing architecture" for data parallelism, data versioning and provenance to support science applications.
- Community version is open source.
- Extensible through user-defined types, functions and operator.
- A better performer than Hadoop (MapReduce), 2-10 times faster, in almost all benchmarks that we have performed so far.

Array is not enough!

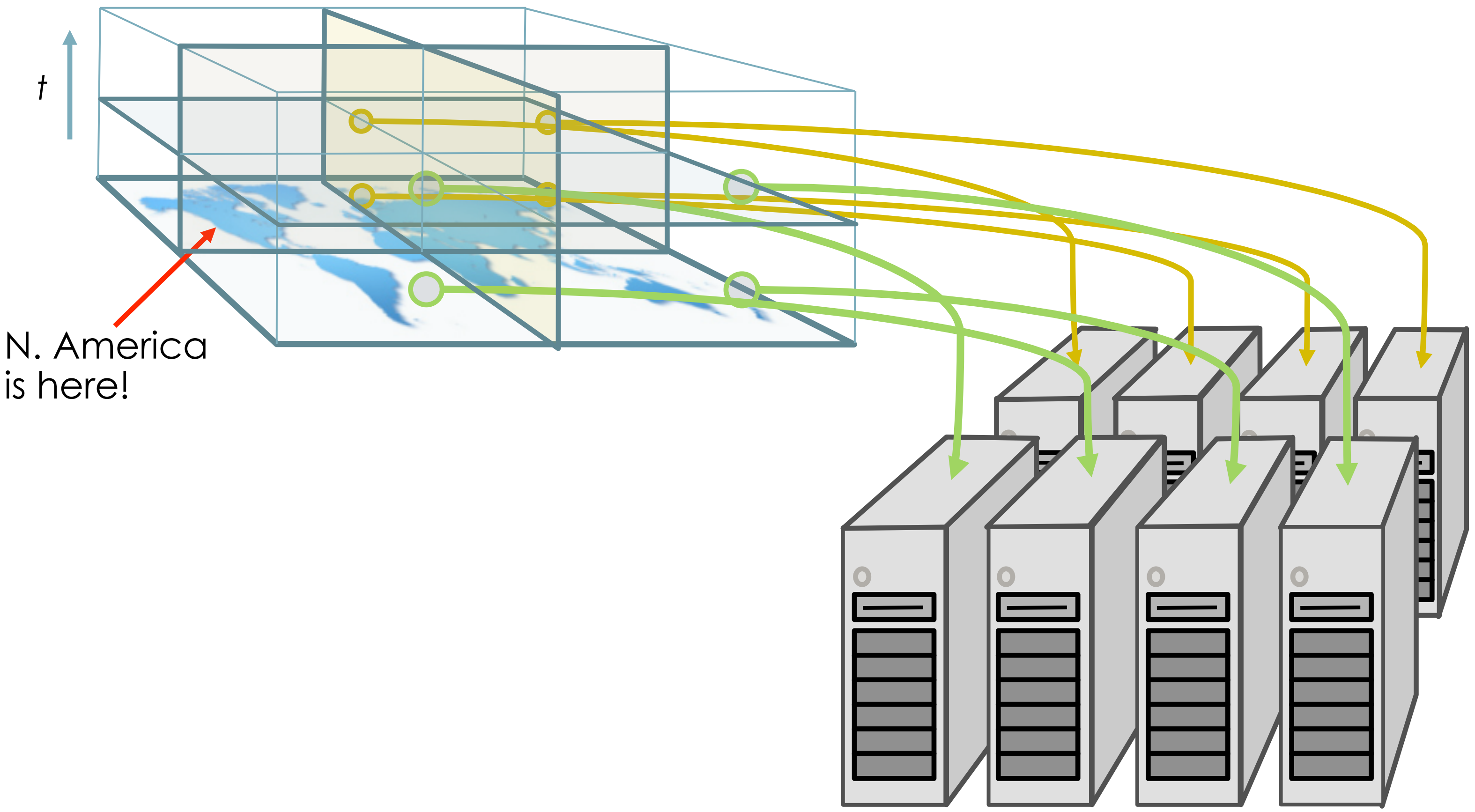
Although “array” is definitely a better model for scientific data,

it is not sufficient for Earth Science.

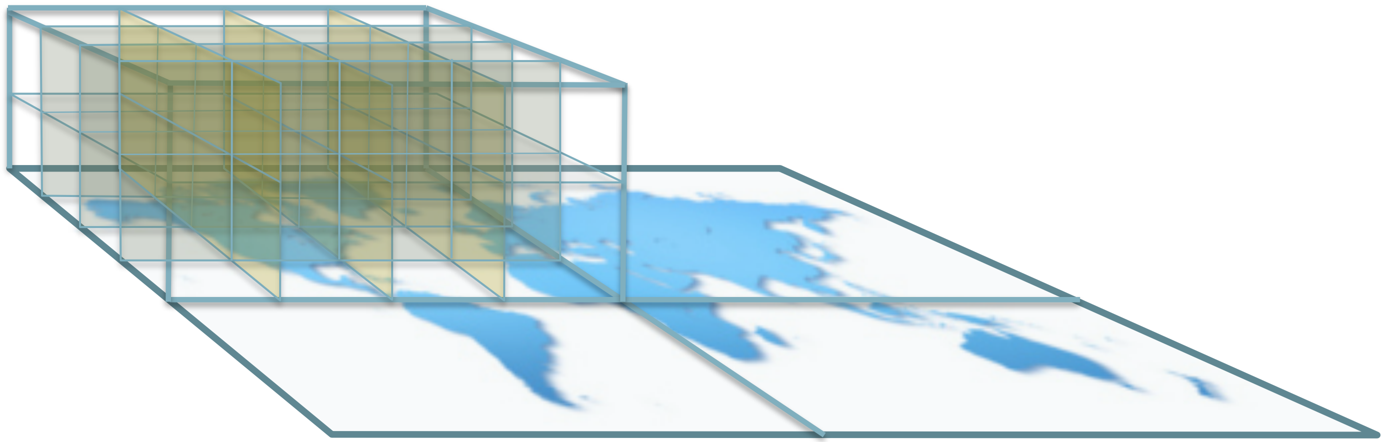
”Logical data independence” needs to be extended.

UNIFIED DATA REPRESENTATION

Simplistic Data Placement (Layout)



Better Load Balance – Smaller Chunks

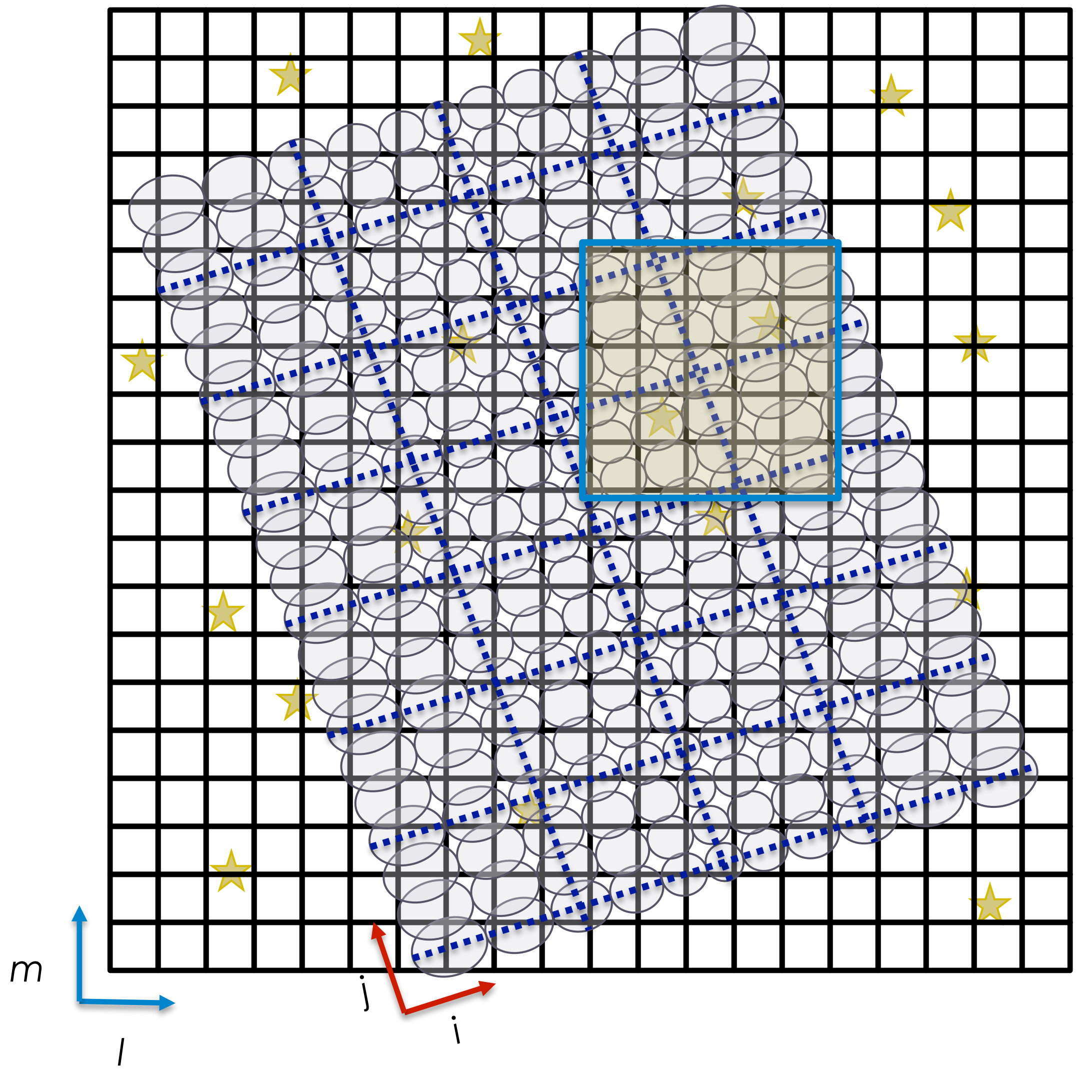


A Step Beyond Data Placement

Data Placement (Co-)Alignment

- In most cases, Earth science analysis requires *spatiotemporal coincidence*. For example:
  - Getting the environment conditions, e.g. pressure, temperature, humidity, etc., of where and when there is precipitation (conditional subsetting),
  - Comparing outputs from different models, and
  - Comparing the same geophysical quantity obtained from different modes of observations, instruments, and/or retrievals.
- If placements of required data are misaligned, data movements within the cluster (repartitioning on-the-fly) ensue.
  - Required partitions for dataset *A* reside on a different set of nodes for those from dataset *B*.
- If placements of different datasets can be aligned for the majority of analysis cases, we achieve better performance by avoiding expensive repartition operations.

Grid, Swath, and Point Data Models



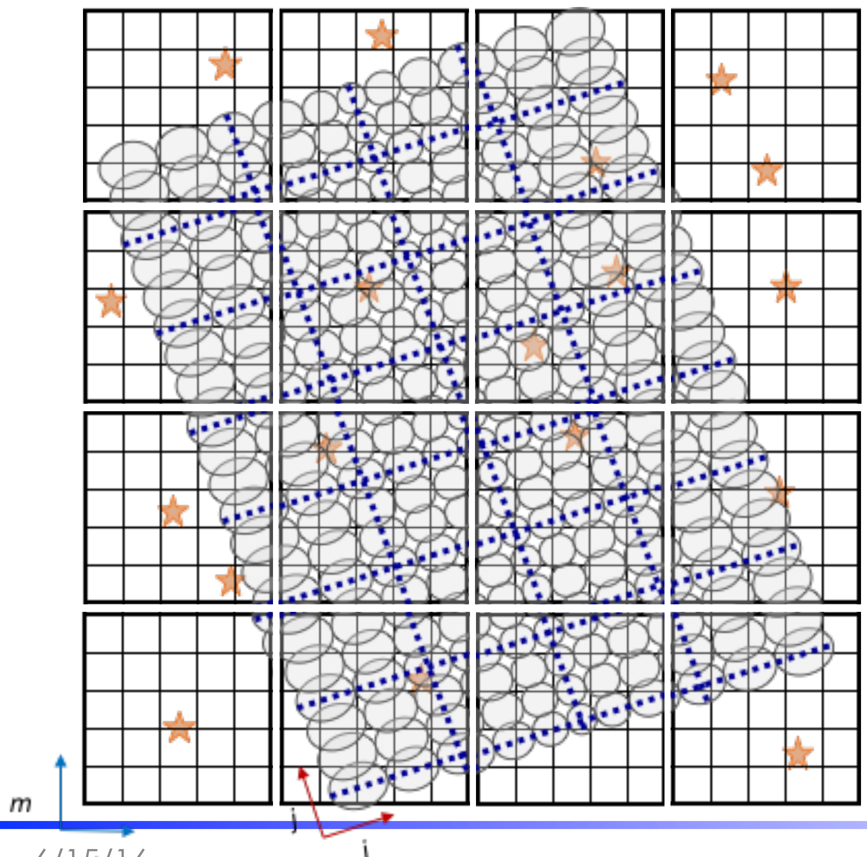
Swath
Point
Grid

★ Observation Station

◯ Sat. Instrument IFOV

Optimal Partition

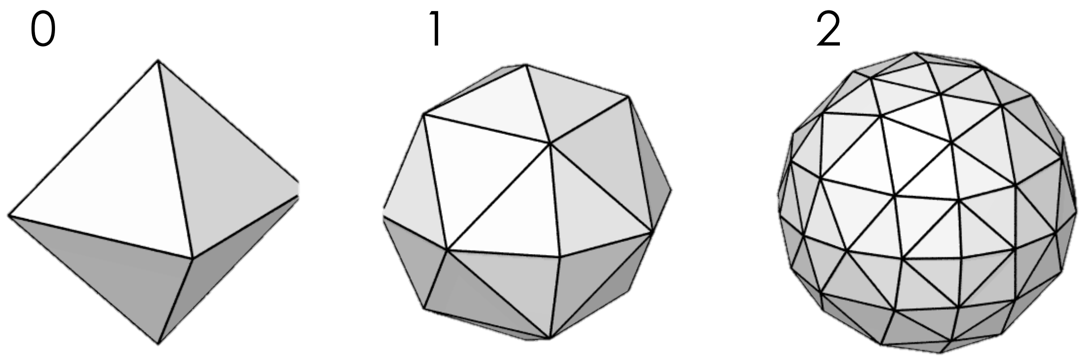
- Since most of our analyses require spatial or temporal coincidences, or both, data should be partitioned so that data for the same spatiotemporal subspace reside on the same node of the SNA, as demonstrated in the schematic diagram below (for space only).



As one can see, it will be tedious to partition datasets in the case-by-case manner illustrated on the left.

- ***How can we generalize it, so the partitioning may be simplified and automated to guarantee placement co-alignment?***

Hierarchical Triangular Mesh

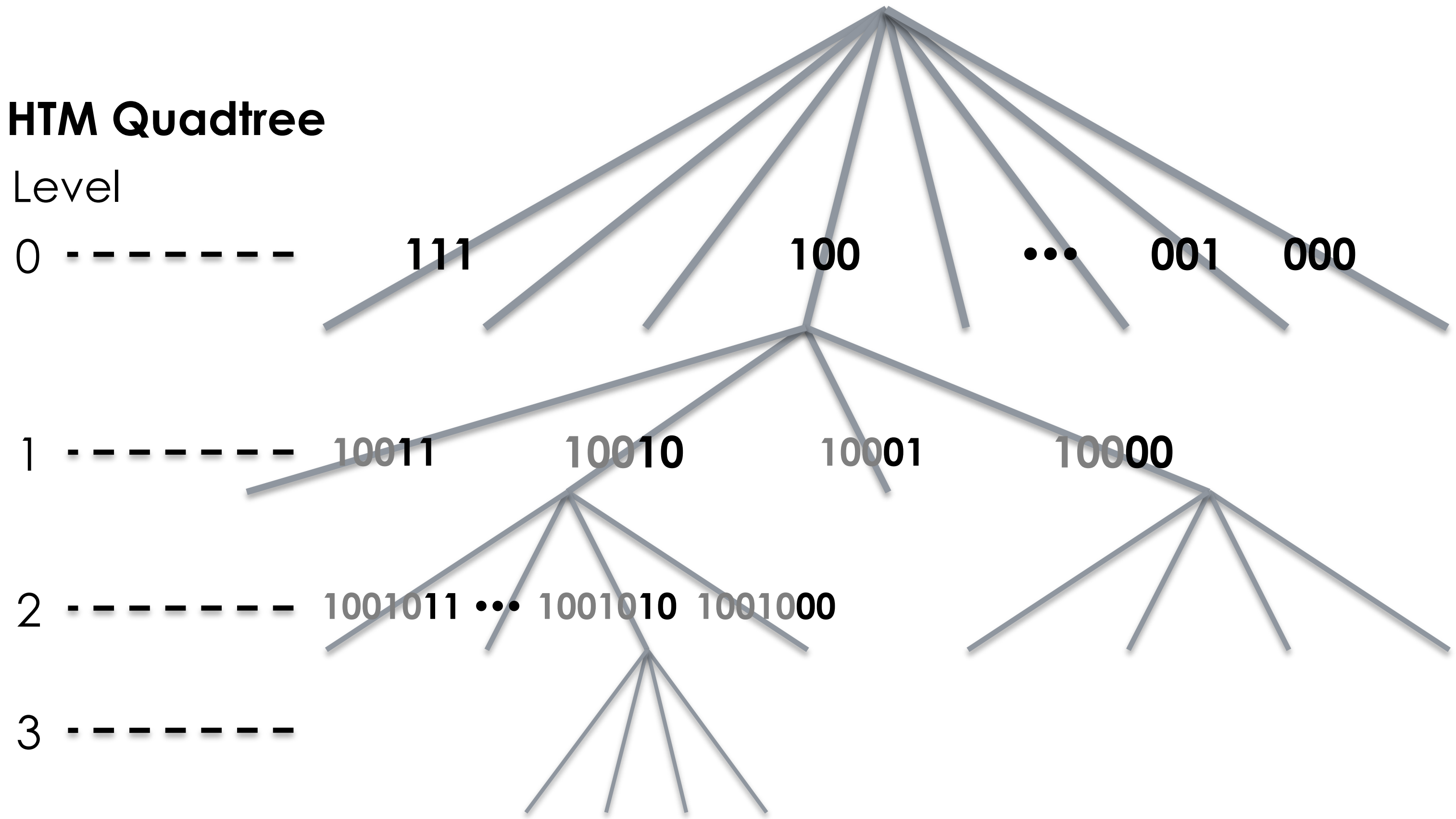


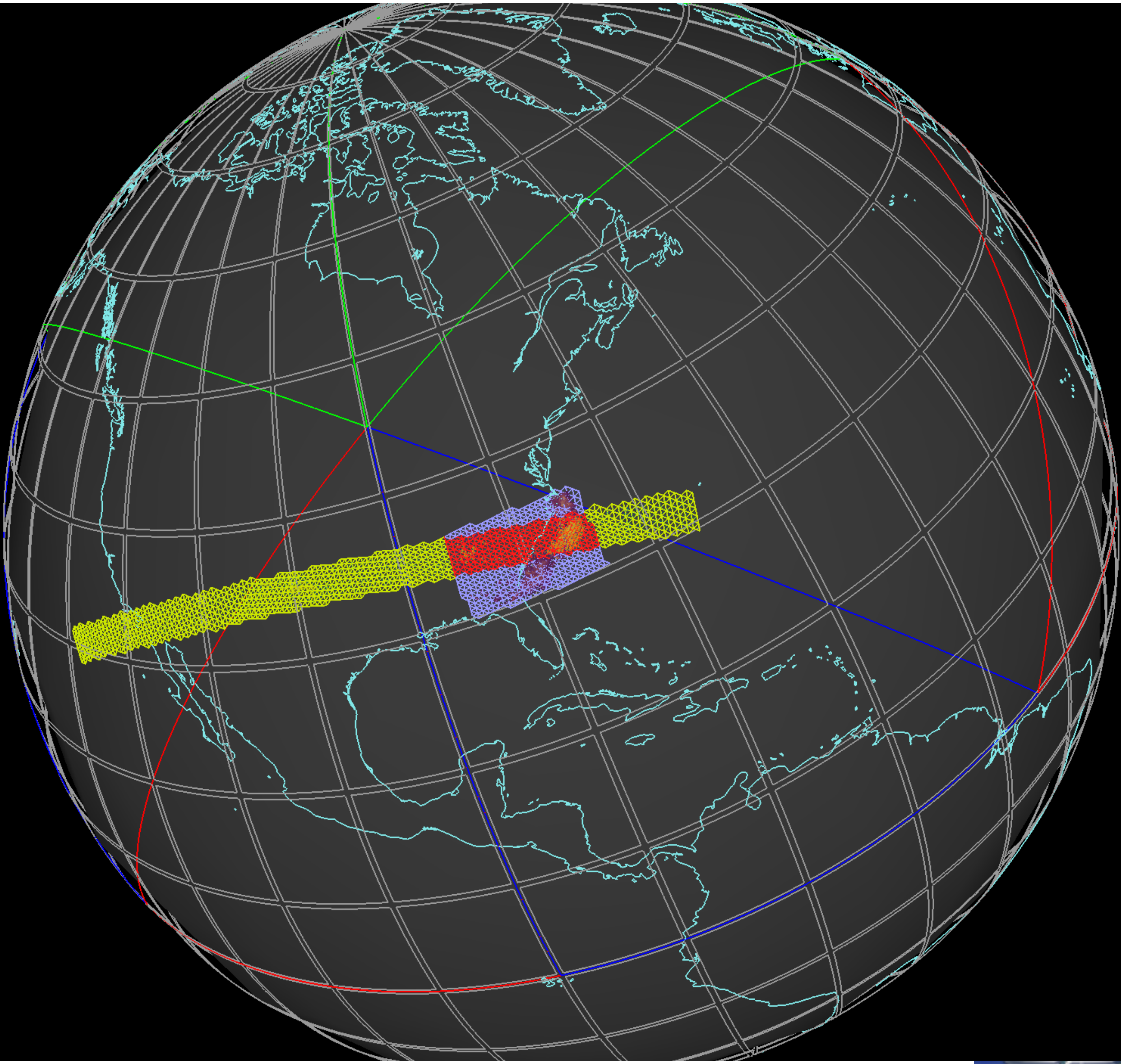
- Proposed solution: Hierarchical Triangular Mesh (HTM)
  - Start with an inscribing octahedron of a sphere.
  - Bisect each edge.
  - Bring the bisecting points to inscribe the sphere to form 4 smaller spherical triangles.
  - Repeat

Advantage of HTM

- Earth's 2D surface is indexed with 1D indices and form a **quad tree**.
  - An HTM index, HID, can be assigned to every geolocation.
- The resolution reaches **≲1 m**, at the 24th level (requiring 49 bits).
- If a level of the quad tree is chosen to be the chunk length for partitioning, all levels below it (same approximate geographical neighborhood) will reside on the same node.
- Thus, using the HTM index allows us to simultaneously
  - **Geo-reference different data representations in a uniform way**, and
  - **Ensure data placement (co-)alignment for diverse datasets.**

HTM Quadtree and Numbering





Anticipated Complications

- Algorithms for calculating quantities that require a neighborhood, e.g. derivatives or the Sobel operator, needs to be reformulated.
  - The principles for deriving these formulae are known.
- Earth scientists currently have little experience with HTM indexing.
  - Even when the aforementioned quantities can be efficiently calculated, scientists would want to check them out with their established practices.
  - Therefore, we need to be able to **convert back and forth easily and efficiently** between HTM indexing and original indexing of the source arrays!
    - A solution (conservative indexing) to this has been found and implemented!

Impact of Data Placement Misalignment

SCIDB VERSUS SPARK+HDFS

“Boundary Conditions”

The following comparisons were performed before HTM wasimplemented in SciDB.

Thus, it was necessary to transform and homogenize the arrays into the same “shape” – the *Data Preparation* step.

Impact of Data Misalignment

- Investigate impact of data placement misalignment and
- Compare SciDB versus Spark performance.
- MERRA (only CONUS) and NMQ data are "massaged" to have the same shape.
  - 1 MERRA array co-aligned with NMQ and 1 not.
- More complete examination than what was done for the "Data Container Project" and presented at AGU FM 2015.
  - Data preparation queries added to Spark+HDFS

SciDB vs. Spark: Queries

Perform the following queries in both SciDB and Spark:

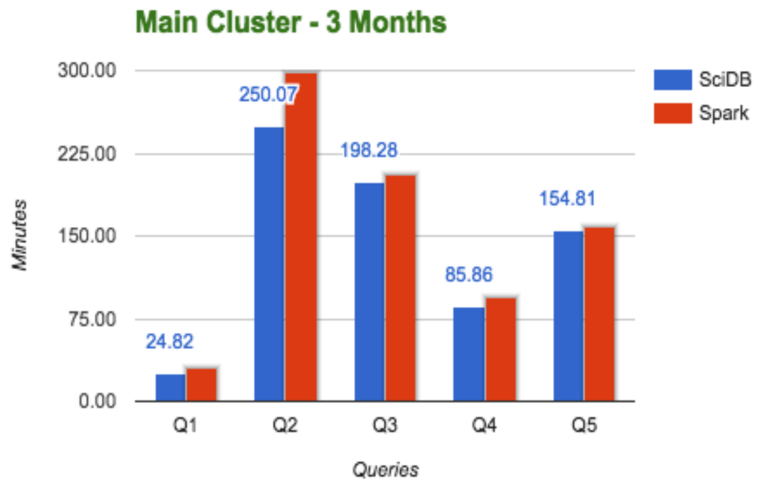
- ***Data Preparation***
  - **Q1**: Resample MERRA MAT1NX array to 0.1° resolution in both latitude and longitude.
  - **Q2**: Coarsen NMQ array also to 0.1° resolution.
- ***Data Analysis***
  - **Q3**: Repartition resampled MERRA array on the fly to compare with precipitation rates of coarsened NMQ array.
    - Combining both Query 4 and 5 in one query.
  - **Q4**: Align resampled MERRA and coarsened NMQ array.
  - **Q5**: Compare precipitation rates using co-aligned arrays.

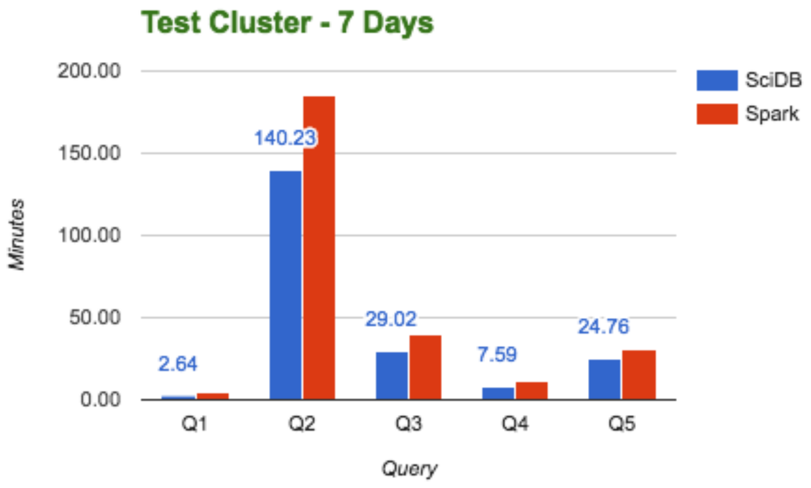
SciDB vs. Spark: Computing Environments

- 2 clusters:
  - Main Cluster: 28 nodes
  - Test Cluster: 3 nodes
- Node Specification:
  - Virtual Node, with both SciDB and Spark installed
  - Each on a separate physical container
    - CentOS 6.5.
    - 8 cores
    - 32GB RAM
- SciDB Configuration:
  - SciDB 15.7
  - 2 instances per virtual node
- Spark Configuration:
  - Cloudera 5.1, Spark 1.4.1, HDFS datastore
  - 8gb executors ~ 2 executors per node

SciDB vs. Spark: Running Times

- 28-node Main Cluster
- 3-node Test cluster





Spark+HDFS Overhead

- No user control over data placement in HDFS, since it handles placement automatically.
- MERRA MAT1NX and NMQ arrays must be combined into one array to guarantee placement alignment between them.
- The overhead for this operation in Spark +HDFS is >50% (~3 hours on the main cluster).
- This may be acceptable for 2 data arrays, it will become impossible as the number of datasets/arrays increases!

THANK YOU!

Business Model

High-Performance Computing and Data-Intensive Computing

- High-Performance Computing (HPC)
  - Computation-bound applications, e.g. model simulation.
  - Centralized high-performance (and expensive) file systems.
- Data-Intensive Computing
  - Input/Output (I/O)-bound applications, e.g. data production and data analysis.
    - Data production – e.g. producing L1, L2, L3, and/or L4 datasets for a mission.
    - Data analysis – e.g. analyzing data products and datasets.
  - Distributed commodity compute/storage (and low-cost) resources.

- Centralization on a distributed, robust, and fault-tolerant system.
- Sophistication in access control.