Data Management for Earth System Science

James Frew

Institute for Computational Earth System Science
University of California, Santa Barbara
frew@icess.ucsb.edu

Jeff Dozier

School of Environmental Science and Management
University of California, Santa Barbara
dozier@esm.ucsb.edu

ABSTRACT

Earth system science is a relatively recent scientific discipline that seeks a global-scale understanding of the components, interactions, and evolution of the entire Earth system. The data being collected in support of Earth system science are rapidly approaching petabytes per year. The intrinsic problems of archiving, searching, and distributing such a huge dataset are compounded by both the heterogeneity of the data, and the heterogeneous nature of Earth system science inquiry, which synthesizes models, observations, and knowledge bases from a several traditional scientific disciplines.

A successful data management environment for Earth system science must provide seamless access to arbitrary subsets and combinations of both local and remote data, and must be compatible with the rich data analysis environments already deployed. We describe a prototype of such an environment, built at UCSB using database technology pioneered by the Sequoia 2000 Project. We specifically address its application to a problem that requires combining point observations with gridded satellite imagery.

1. Introduction: Data Management Challenges

Earth system science is a relatively recent scientific discipline that seeks a global-scale understanding of the components, interactions, and evolution of the entire Earth system, particularly as they affect the survival of the human species. To answer questions such as “Is Earth’s climate warming?” and “What local and global impacts will likely result?” requires synthesizing data from a staggering array of sources, as well as modeling and analysis techniques from several traditional scientific disciplines, applied over timescales of decades to centuries.

The scope and diversity of Earth system science problems lead directly to the primary data management challenge: dealing with **huge, heterogeneous datasets**. Considering only data acquired from imaging sensors on Earth-orbiting satellites, we find they have already populated several multi-terabyte tape archives. A single pass over Earth’s surface at 1 km resolution will generate half a billion observations, and there are several currently operating satellite systems that acquire this level of coverage on a daily or weekly basis. The advent of the advanced sensors on the Earth Observing System (EOS) [*Asrar 1995*] satellites will increase this data rate by at least two orders of magnitude, up to about a terabyte of raw observational data per day within the next ten years.

While satellite-acquired data are mostly gridded (i.e., directly representable as multidimensional arrays), there are significant sources of point and vector data. These include base maps, atmospheric soundings (directly from balloons, and indirectly from satellites), and meteorological measurements (fixed weather stations and mobile field measurements). A recent and significant source of point data are satellite-borne radar altimeters, which we discuss in more detail below. Finally, a huge volume of synthetic (modeled) data is produced by global general circulation models (GCMs), typically as variable-resolution grids (fields).

However the data are structured, they are almost always packaged in a fashion whose granularity has more to do with ease of production, storage, or delivery, than with ease of use. Moreover, most Earth system science computing environments use file systems as their basic data management tool. But what scientists really want to do is **manage data, not files**. They are more concerned with retrieving data by attributes and/or contents, than by some complex and incomplete encoding of semantic metadata into a file name.

Most Earth system science computing environments have a substantial investment in processing and analysis tools, either commercial or locally-developed. The data access interfaces of these tools often cannot be easily changed. In addition, every local environment has different constraints on where and how compute- or I/O-intensive tasks may be performed. Scientists do not want to have to worry about any of this; thus a fundamental challenge is to **integrate data management with data analysis**, creating the illusion of a seamless data space that scientists may manipulate with their favorite analytical tools.

Finally, the Earth system science data management environment is unavoidably a distributed one. Data repositories are already widely dispersed geographically, a trend which is likely to accelerate [*NRC 1995*]. Similarly, some analyses require the kind of computing power that is either concentrated at supercomputing centers, or available only by marshaling hundreds or thousands of processors over wide-area networks. The data management implication of this situation is that any data a problem requires cannot be assumed to be local; thus, the data management environment must be able to **access remote data as if they were local data**.

2. “Database-Centrism:” the Sequoia Legacy

The Sequoia 2000 Project [*Dozier 1994*] was a 3-year $14M joint venture between the University of California and the Digital Equipment Corporation. The project explored the application of emerging database, network, storage, and visualization technologies to Earth system science problems, resulting in a “database-centric” metaphor for scientific computation that has carried over into our subsequent research.

Consider the end-to-end Earth system science data management problem as repeated iterations through the sequence:

ingest
store
locate
retrieve
analyze

A database management system (DBMS) can facilitate each step of this process:

- **ingest**: As new data are acquired, the DBMS logs their insertion with standard semantic metadata (e.g., data source, Earth surface location, etc.) Triggers in the DBMS may also initiate any default processing (i.e., any processing that should always occur as soon as the data are available).
- **store**: DBMSs manage their own secondary storage, presenting higher-level abstractions (e.g., relations) to their clients. Some Earth system science data (e.g., point measurements) are a good fit with standard DBMS types, while other data (e.g., images) require a type-extensible DBMS to be managed transparently. Either way, the storage abstraction is far more uniform and flexible than the physical granules (tapes, CD-ROMs, etc.) with which a scientist must contend outside a DBMS.
- **locate**: DBMS query languages (e.g., SQL) allow data to be located by specifying boolean or higher-level constraints on their values or associated attributes. Thus they support far more flexible location mechanisms than the file naming conventions most scientists would otherwise employ.
- **retrieve**: Data retrievals from a DBMS are potentially seamless, to the extent that the DBMS supports (a) the full range of Earth science data types, and (b) processing of (i.e., applying database functions to) the retrieved data before they are delivered to the requester. For example, while it is trivial for a “vanilla” relational database to retrieve exactly the scalar data requested from a table, seamless retrieval of image data requires extensions to the traditional DBMS set of data types and operators. When such extensions are available, a DBMS effectively eliminates the distinction between locating and retrieving data.
- **analyze**: The client/server architecture of current DBMSs allows them to be connected to an Earth system science analysis environment (e.g., [*Farrell 1994*]), although almost always through a middleware “glue” layer that

matches the two environments' differing semantics (e.g., functional versus nonprocedural). Moreover, extensible DBMSs allow analysis functionality to be implemented under the control of the DBMS, either as user-defined functions, or as external processing triggered by DBMS events. New objects created under DBMS control can be ingested into the DBMS, thus closing the data management loop.

To the extent that all data relevant to an investigation are either ingested into the DBMS, or created under its control, complete provenance or **lineage** is available for each object. This "laboratory notebook" functionality is itself a significant contribution of database-centric science data management.

3. Example: Ocean Modeling with TOPEX and AVHRR

We are building a prototype Earth system science data management system that applies the database-centric approach to a specific problem: For a study area in the Atlantic Ocean near Bermuda [*Michaels 1996*], we want to determine the local and advective components of the upper ocean heat balance. The overall processing flow is shown in Figure 1:

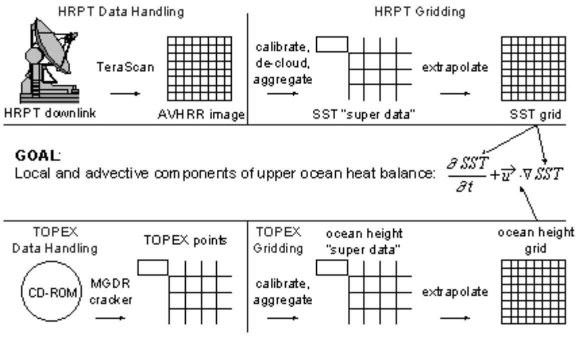

Figure 1: Ocean heat balance data flow

Two data sources are used: thermal imagery from the Advanced Very High Resolution Radiometer (AVHRR) on board the NOAA satellites [*Kidwell 1991*], and radar altimetry from the Ocean Topography Experiment (TOPEX) [*Benada 1993*]. The AVHRR data are acquired in real time at a "High-Resolution Picture Transmission" (HRPT) ground station in Bermuda, and are processed at UCSB into sea-surface temperatures (SST) grids. The TOPEX data are distributed on CD-ROMs by the Jet Propulsion Laboratory as "Merged Geophysical Data Records" (MGDR), which are processed at UCSB into sea level height anomaly grids (i.e., that portion of the measured sea level elevation not explained by systematic factors, such as tides or ocean bottom topography).

Although this analysis appears straightforward, there are several problems that make this a nontrivial exercise in data management. The most significant problem is the structural difference between the TOPEX and AVHRR data. AVHRR data are acquired as grids, with daily synoptic coverage of the entire planet from each of (currently) two satellites. We store the grids as files, but register them in the DBMS as external large objects, so their contents are accessible to DBMS server functions. TOPEX data, however, are acquired as point observations, made every second directly beneath the satellite, along rigidly repeating and widely separated ground tracks (projection of the satellite's orbit on the Earth's surface). We store each TOPEX observation as a tuple

(location, time, sea level anomaly)

in a single DBMS table. This table currently contains over 1.5M rows for our limited study area (a 10 by 10 degree latitude-longitude box centered on Bermuda) and time (1993). To put this in perspective, the TOPEX altimeters have acquired well over 100M observations since their launch in 1992.

The second problem is interpolating sparse input data to a common grid. For TOPEX data, the superimposition of the ascending (south to north) and descending (north to south) ground tracks yields a "fishnet" pattern of data coverage: dense along the groundtracks, but absent in the diamond-shaped regions in between. A similar problem occurs with the AVHRR data, because of the large fraction of pixels (grid cells) that are at any given time obscured by clouds. Although not as systematic as the gaps in the TOPEX coverage, the cloud-covered areas in an AVHRR scene must likewise be "filled in" to achieve a usable grid. In both cases, we use an interpolation technique called **objective analysis** [*Carter 1987*], which can produce a theoretically optimal interpolated field from sparse input data, if the statistical properties of the input data are sufficiently well characterized. To obtain robust statistics, we aggregate both the AVHRR and the TOPEX data into "super-data" (reduced temporal and spatial resolution). At this stage the AVHRR super-data are compact enough to be represented in DBMS tables, just like the TOPEX data.

Another problem is ensuring that sufficient semantic metadata are saved in the DBMS, both for searching, and for data quality tracking. We use a simplified version of the metadata standards and schema developed for the Sequoia Project [*Anderson 1994*].

We should note that, in addition to data and metadata, the database also contains all processing information: names of internal functions or external programs, saved parameter settings, etc. [*Brown 1995*]. This is best illustrated by an example, in which we will populate a pre-specified grid with interpolated TOPEX point data. First, we retrieve the name of the actual executable that currently implements the objective analysis interpolator generically referred to as ‘‘oa:’’

```
select ProcessPath from LocalProcs
where ProcessName='oa'
and ProcessVersion='1'
```

Next, we retrieve a saved set of objective analysis parameters that define a standard grid that we refer to as ‘‘Bermuda Small.’’ In this case, we use the ‘‘SQLcopy’’ function to retrieve the parameters directly into a file (whose name is returned), which is how the objective analysis interpolator expects to read them:

```
return SQLcopy('
    select Parameters
    from OA_Parameters
    where ParmsetName=''Bermuda Small'';
    ')
```

Now, we select the TOPEX data we wish to interpolate and save them in a second file. Again, the file name is essentially an opaque handle provided by the DBMS. The ‘‘select’’ statement retrieves TOPEX data located between 26 and 36 degrees north latitude, and 61 and 69 degrees west longitude, and obtained between 342 and 402 days since the launch of the satellite (the database now includes functions that cast more human-readable dates and times into days-since-launch):

```
return SQLcopy('
    select mean_days, sla,
        X(Point(location, 1)),
        Y(Point(location, 1))
    from TOPEX
    where Contains(
        ''(26,291,36,299)''::Box::Poly,
        location
        )
    and mean_days >= 342
    and mean_days <= 402;
    ')
```

At this point, we have enough information to run the interpolator outside the database. In subsequent versions of the system we plan to encapsulate the interpolator in a DBMS server function.

Once the interpolator finishes, its output grid must be registered in the database. This is a fairly complicated sequence of ‘‘inserts’’ into several metadata tables, which will not be reproduced here. However, we should note that the same table ‘‘OA_Parameters’’ from which the interpolation parameters were retrieved, also contains all primary key values necessary to orchestrate the population of the corresponding output grid metadata, so this process is as automated as possible.

4. Conclusions

Although we are still a long way from having a completely operational database-centric Earth system science environment, our efforts to date have been both promising and immediately useful. The DBMS is now the operational custodian of our TOPEX data, and the stored-parameter-set mechanism is being adapted to other analysis procedures. The most significant remaining problems are:

- **simpler scripting interface**: Current connections between the DBMS and external procedures (such as the objective-analysis interpolator) are coordinated by scripts running outside the DBMS. These scripts are difficult to write, because they mix query-oriented and procedural programming styles, and because they must often explicitly convert between data representations used in the DBMS and those expected by external procedures (e.g, converting angles between radians and degrees). To make the database-centric model more accessible to Earth system scientists, we need simpler ways to generate these scripts, either from templates, or by adding conversion functions and ‘‘canned’’ queries to the DBMS.
- **more functionality inside the DBMS**: It is now becoming reasonable to expect mainstream DBMSs to support user-defined functions and data types. With such extensions, a procedure such as objective-analysis interpolation could be implemented entirely as a database function, obviating the need to move the metadata associated with such a function (stored parameters, descriptions of input and output objects, etc.) between the database and an external procedure.

5. Acknowledgments

We thank Jean Anderson, Paul Brown, Jim Davidson, Erik Fields, David Siegel, Michael Stonebraker, and Richard Troy for their contributions to the design and implementation of the prototype data management system described herein.

This work was supported by the National Aeronautics and Space Administration.

6. References

Anderson 1994

J. T. Anderson and M. Stonebraker, "Sequoia 2000 Metadata Schema for Satellite Images," *SIGMOD Record*, vol. 23, no. 4 (December 1994).

http://www.acm.org/sigmod/sigmod_record/9412/stonebraker.ps

Asrar 1995

G. Asrar and R. Greenstone (eds.), *1995 MTPE EOS Reference Handbook*, EOS Project Science Office, NASA / Goddard Spaceflight Center, Greenbelt, MD (1995).

http://eospso.gsfc.nasa.gov/eos_reference/TOC.html

Benada 1993

R. Benada, *PO.DAAC Merged GDR (TOPEX/POSEIDON) Users Handbook*, Rep. JPL D-11007, Jet Propulsion Laboratory, Pasadena, CA (1993).

http://podaac-www.jpl.nasa.gov:2031/DATASET_DOCS/topex_mgdr.html

Brown 1995

P. Brown and M. Stonebraker, "BigSur: A System for the Management of Earth Science Data," in U. Dayal, P. D. Gray, and S. Nishio (eds.), *VLDB '95: Proceedings of the 21st International Conference on Very Large Data Bases, Zurich, Switzerland, Sept. 11-15, 1995*, pp. 720-728, Morgan Kaufmann Publishers, Inc., San Francisco, CA (1995).

Carter 1987

E. F. Carter and A. R. Robinson, "Analysis Methods for the Estimation of Oceanic Fields," *Journal of Atmospheric and Oceanic Technology*, vol. 4, no. 1, pp. 49-74 (1987).

http://www.taygeta.com/objan.html

Dozier 1994

J. Dozier, M. Stonebraker, and J. Frew, "Sequoia 2000: a next-generation information system for the study of global change," in *Proceedings of the Thirteenth IEEE Symposium on Mass Storage Systems, Annecy, France, June 12-16, 1994*, IEEE Computer Society, Los Alamitos, CA (1994).

Farrell 1994

W. Farrell, J. Gaffney, J. Given, R. D. Jenkins, and N. Hall, "A Hydrographic Database built on Montage and S-PLUS," *Sequoia 2000 Technical Report 94/47*, University of California, Berkeley (1994).

http://s2k-ftp.CS.Berkeley.EDU:8000/sequoia/tech-reports/s2k-94-47/

Kidwell 1991

K. Kidwell, *NOAA Polar Orbiter Data User's Guide*, U.S. Dept. of Commerce, Washington, DC (1991).

Michaels 1996

A. Michaels and A Knap, "Overview of the US JGOFS Bermuda Atlantic Time-Series Study and the Hydrostation S Program," *Deep-Sea Research*, vol. 43, pp. 157-198 (1996).

NRC 1995

National Research Council, *A Review of the U.S. Global Change Research Program and NASA's Mission to Planet Earth/Earth Observing System*, National Academy Press, Washington, DC (1995).

http://www.gcrio.org/USGCRP/LaJolla/cover.html