Querying and Visualizing Gridded Datasets for e-Science

Bill Howe
Department of Computer Science
Portland State University
Portland, Oregon
bill@cse.ogi.edu

David Maier
Department of Computer Science
Portland State University
Portland, Oregon
maier@cs.pdx.edu

We demonstrate a web service and client application for querying and visualizing datasets defined over a topological *grid*[1] structure. For example, a timeseries might be defined over a 1-dimensional grid, while fluid flow over an airplane wing might be defined over a 3-dimensional grid. Such gridded datasets are produced by Earth science simulations, 2D and 3D visualization software, and image processing systems.

Existing systems for manipulating gridded datasets capture only simple grids or do not support remote access. We exercise a novel data model developed in previous work [5] by using it to build a web-enabled visualization system. We show that complex visualizations, including animations, can be expressed using our data model and associated algebra. Further, we demonstrate a standards-based, service-oriented architecture for processing the algebra expressions and returning results over the web.

The context for our interest in gridded datasets is CORIE [2], an Environmental Observation and Forecasting System designed to support scientific and industrial interests in the Columbia River estuary. The CORIE system both measures and simulates the physical properties of the estuary, generating 5GB of data and thousands of *data products* for each simulation run, including visualizations, aggregated results and derived datasets. The data products are consumed for many purposes, including salmon habitability studies and environmental impact assessments.

In the current production CORIE system, "canned" visualizations are produced eagerly for every run. Users cannot customize their data products nor access the data directly, inhibiting data sharing. Using our web service, clients can not only retrieve data, but they can perform arbitrary transformations on the data prior to download (i.e., evaluate queries). Results of gridfield expressions are usually quite large; visualization techniques are required to convey their full meaning. Our prototype client application can choose an appropriate visualization automatically based on the structure of the gridfield.

Sharing scientific data with other researchers is certainly not a goal unique to the CORIE system. The term *e-Science* is used to connote global, distributed collaboration enabled by sharing of both data and compute resources [1].

The systems to support e-Science are still evolving. Advocates of Grid computing [4] suggest that large-scale scientific data management requires seamless inter-operation of heterogeneous compute resources and heterogeneous data sources. Currently, these *Data Grid* technologies are in their infancy, adopting uninterpreted files as the currency of exchange and analysis. As the database community has demonstrated, relying on client applications to interpret raw files leads to a problem of *data dependence*, where applications are brittle with respect to changes in data location, organization, or access methods.

To eliminate data dependence problems for grid-structured data, we have proposed a logical model of gridfields to shield applications from details of physical organization. Due to space constraints, we highlight a few notable features of our model and refer the reader to previous work for a full development. Our model offers the following benefits:

- Support for arbitrary, irregular grids, including *nested* grids.
- Algebraic optimization.
- Separation of topology and geometry to expose equivalences.
- Support for grids of arbitrary dimension.

[1] Our usage of the term "grid" is not directly related to the notion of Grid computing. Unfortunately, both meanings are used in the context of e-Science. We use the term "grid" for consistency with our previous work, but the word "mesh" also suffices.

Proceedings of the 21st International Conference on Data Engineering (ICDE 2005)
1084-4627/05 $20.00 © 2005 **IEEE**

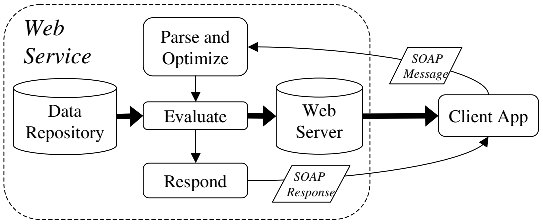

Figure 1. Architecture of the web service and client application.

- Neat integration with existing client tools.
- A full implementation with experimental results.

In this demonstration, we present a web service for evaluating gridfield expressions and a client program for automatic visualization of results.

1 Architecture and Demonstration

Our server is implemented as a web service, responding to XML requests and returning XML responses with a URL to an associated binary payload. Scientific datasets are usually too large to be translated into any ASCII representation, and schemes for transporting binary data directly within an XML document are fraught with problems [3]. Instead, we have chosen to use the Simple Object Access Protocol (SOAP) for passing messages, but use a separate protocol to transfer the potentially large query results.

Figure 1 illustrates our architecture. Thick arrows represent flow of (large) gridded datasets, thin arrows represent flow of (small) query plans or control information. A SOAP request is received from the client and parsed. Some rudimentary algebraic optimizations are applied, such as pushing restrictions down the tree. Then the plan is executed, using pre-compiled query operators. The output of the query is materialized and exposed to the Internet. The URL is sent back to the client, enveloped in a standard SOAP response.

Providing only indirect access to the query results opens up some interesting processing strategies. Clients might ask for the results to be computed asynchronously, accessing them at a later time. Further, clients could track URLs for computed results and serve this information to other sub-clients. Obviously, the lifetime of computed query results cannot be indefinite, but waiting a day or more before purging them is not infeasible at our current usage. Developing full-featured strategies for pre-computing and caching results is beyond the scope of this work.

Table 1. Visualizing gridfield structure.

GridField Structure	Visualization Dimension
"x", "y", and "z" attributes	Spatial Dimensions
1 non-spatial attribute	Color
2 or 3 non-spatial attributes	Vector Glyphs
4+ non-spatial attributes	Glyphs and Color
Nested Linear Grid	Animation
Nested 2-dimensional Grid	Screen Layout
2+ innner grids in one cell	Superimposition

The client application generates visualizations automatically based on the structure of the gridded dataset being viewed. Visualization systems offer only a few core dimensions onto which we can map data. Three spatial dimensions can be visualized, a fourth dimension can be captured via animation, visual objects can be colored by a scalar quantity, and glyphs can be used to visualize 2 or 3 dimensions of vector data. Table 1 shows how gridfield structures are mapped to visualization dimensions. Attribute names specify spatial dimensions, and extra attributes are interpreted as scalar or vector quantities depending on how many there are. A linear outer grid is interpreted as representing time; each inner grid is rendered as a frame of an animation. A 2-dimensional outer grid allows control of the arrangement of the inner grids on the screen (two halves, four quadrants, etc.).

References

[1] R. Allan. Building the e-Science Grid in the UK: Middleware, applications and tools deployed at level 2. In *Proceedings of the UK e-Science All Hands Meeting*, September 2003.

[2] A. Baptista, M. Wilkin, P. Pearson, P. Turner, C. McCandlish, and P. Barrett. Coastal and estuarine forecast systems: A multi-purpose infrastructure for the Columbia River. *Earth System Monitor, NOAA*, 9(3), 1999.

[3] A. Bosworth, D. Box, M. Gudgin, M. Nottingham, D. Orchard, and J. Schlimmer. Xml, soap, and binary data. Technical report, BEA Systems and Microsoft, February 2003. www.xml.com/pub/a/2003/02/26/binaryxml.html.

[4] A. Chervenak, I. Foster, C. Kesselman, C. Salisbury, and S. Tuecke. The Data Grid: Towards an architecture for the distributed management and analysis of large scientific datasets. *Journal of Network and Computer Applications*, 23:187–200, 2001.

[5] B. Howe and D. Maier. Algebraic manipulation of scientific datasets. In *VLDB*, 2004.

[6] W3C. *Simple Object Access Protocol (SOAP) 1.2*, 2003. http://www.w3c.org/TR/SOAP.