



Supporting Large Scale Medical and Scientific Datasets

U. Catalyurek, S. Hastings, K. Huang, V.S. Kumar, T. Kurc,
S. Langella, S. Narayanan, S. Oster, T. Pan, B. Rutt,
X. Zhang, J. Saltz

published in

Parallel Computing:

Current & Future Issues of High-End Computing,

Proceedings of the International Conference ParCo 2005,

G.R. Joubert, W.E. Nagel, F.J. Peters, O. Plata, P. Tirado, E. Zapata
(Editors),

John von Neumann Institute for Computing, Jülich,

NIC Series, Vol. 33, ISBN 3-00-017352-8, pp. 3-14, 2006.

© 2006 by John von Neumann Institute for Computing

Permission to make digital or hard copies of portions of this work for personal or classroom use is granted provided that the copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise requires prior specific permission by the publisher mentioned above.

<http://www.fz-juelich.de/nic-series/volume33>

Supporting Large Scale Medical and Scientific Datasets*

Umit Catalyurek^a, Shannon Hastings^a, Kun Huang^a, Vijay S. Kumar^a, Tahsin Kurc^a,
Stephen Langella^a, Sivaramakrishnan Narayanan^a, Scott Oster^a, Tony Pan^a, Benjamin Rutt^a,
Xi Zhang^a, Joel Saltz^a

^aDepartment of Biomedical Informatics, The Ohio State University, Columbus OH, 43210

1. Introduction

In many fields of medicine, engineering and science, the volume of data generated and processed is in the order of terabytes. Providing support for storing, accessing and analyzing these vast amount of datasets in a distributed environment is a challenge. This paper presents a compendium of run-time and compiler techniques and tools for supporting such applications. We will also provide a quick overview of two applications, oil reservoir management studies and digital imaging of pathology slides, that would benefit from such support.

Simulation-based oil reservoir management studies are an example of applications that generate and reference large volumes of simulation and experimental data. The objective is to develop complex numerical models of subsurface reservoirs and use these models to efficiently search for alternative oil production strategies in order to optimize profits and minimize adverse effects to the environment [35,30,41,48]. In this optimization process, there is a need to provide support for management and querying of large volumes of data generated by simulations or collected from field measurements to refine model parameters and determine the next set of simulations to be carried out. In addition, the datasets can be generated and stored at multiple locations, since the computational requirements of the simulations may require use of machines at supercomputing centers.

Digital imaging of pathology slides have gained an increasing interest over the last decade as hardware for digitizing tissue samples and microscopy slides has rapidly advanced [28]. A main challenge is storing and accessing the very large volumes of data required to represent a large collection of slides [16]. With high resolution scanners, a single focal plane of a digitized slide can be $100K \times 100K$ pixels (30 Gigabytes). Another challenging problem is the querying and processing of large volumes of data for analysis [46,47]. Analysis operations on digital slides range from simple 2D visualization and browsing of images to queries to calculate density distributions to extraction of features representing different layers of tissue or cell types to 3D reconstruction from multiple 2D scans. High performance machines are required to handle the complexity of operations and the large volume of data.

There has been considerable progress in Grid computing technologies in recent years. In addition to a wide array of middleware systems and tools, a services-based view of the Grid has emerged. In this view, data sources and applications are exposed to the environment using standard interfaces. Users interact with the resources through well-defined Grid services protocols. In this way, the complexities and heterogeneity of individual resources can be hidden from clients and greater interoperability among applications can be achieved.

*This research was supported in part by the National Science Foundation under Grants #ACI-9619020 (UC Subcontract #10152408), #EIA-0121177, #ACI-0203846, #ACI-0130437, #ANI-0330612, #ACI-9982087, #CCF-0342615, #CNS-0406386, #CNS-0426241, Lawrence Livermore National Laboratory under Grant #B517095 (UC Subcontract #10184497), NIH NIBIB BISTI #P20EB000591, Ohio Board of Regents BRTTC #BRTT02-0003.

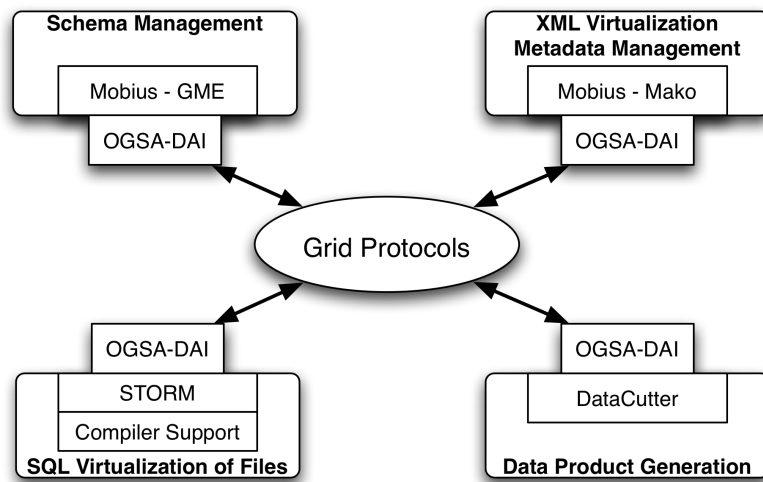


Figure 1. Middleware components and toolkits to support data-driven scientific applications in the Grid.

Several core functionalities need to be supported in an end-to-end system for enabling data-driven scientific applications in a Grid environment. These include management of data types and metadata, virtualization of data sources and data subsetting, data product generation (e.g., data aggregates from data subsets) and Grid services interfaces. In our group, we develop an integrated suite of middleware components and compiler techniques to provide these functionalities. These middleware components are shown in Figure 1. In this suite, DataCutter, which is a component-based middleware, enables combined use of task and data parallelism and is used to support data product generation (e.g., aggregates of data subsets) [13]; STORM [36,37] provides virtualization of file based datasets as object-relational tables and support for data subsetting; Mobius [27] supports management of data definitions and data types as XML schemas, XML virtualization of data, and metadata management. In an ongoing project [38], we are integrating these middleware systems with the Open Grid Services Architecture Data Access and Integration (OGSA-DAI) middleware toolkit [39] to allow access to the functionality provided by these components via OGSA-DAI Grid services protocols.

In this paper, we present a compendium of these run-time and compiler techniques and tools for supporting large scale, data-driven applications on the Grid, using two motivating applications from two different domains; oil reservoir management and digital microscopy imaging.

2. Applications

2.1. Oil Reservoir Management Studies

Effective oil reservoir management requires accurate characterization of the reservoir properties and efficient management strategies that involve optimized placement of production and injection wells. Simulation-based oil reservoir management is a viable approach to evaluate different optimization strategies and to understand changes in reservoir properties over long periods of time [30]. Various production strategies (i.e., the number and placement of injection and production wells) are simulated using a numerical model of the reservoir under study. In addition, changes in reservoir characteristics (e.g., rock properties) over time are tracked by seismic data simulations (or seismic measurements in the field). Data obtained from seismic and reservoir simulations are stored for analysis. The data analysis processes subsets of seismic simulation datasets and reservoir simulation

datasets in order to generate summary data such as production rates over a time period, bypassed oil regions in the reservoir, and rock properties in the reservoir. The results of the analysis can be used to refine the reservoir models, simulate new production strategies and collect additional seismic data.

A good understanding of fluid and rock properties in an oil reservoir is necessary for designing optimized production strategies. Since only a partial knowledge of critical parameters such as rock permeability in the reservoir is available, it is desirable to incorporate geologic uncertainty in complex reservoir models. An approach is to simulate alternative production strategies with varying number, type, timing and location of wells, applied to multiple realizations (simulation runs) of geostatistical models. This approach can lead to large volumes of output data [48].

Simulations are performed on a three-dimensional mesh over several time steps. Each realization corresponds to different geostatistical models and different number of wells and well placements. At each time step, the value of seventeen separate variables and cell locations in 3-dimensional space are output for each cell in the grid. Common analysis scenarios involve queries for economic evaluation as well as technical evaluation, such as determination of representative realizations and identification of areas of bypassed oil. Examples of client requests include “*Find all the potential bypassed oil cells between time T_1 and T_2 in realization A.*” and “*Retrieve the oil saturation values at all mesh points from realizations A and B between time steps T_1 and T_2 ; visualize the results.*”.

The physical characteristics of a reservoir change over time. These changes in reservoir material properties should be detected and incorporated into reservoir models. Seismic surveys of the reservoir can be used to track changes [30]. Seismic data is recorded as sound traces generated by multiple sound sources on the surface and sampled by receivers at the bottom and on the surface of the reservoir. The sound traces are used to infer subsurface material properties. The surveys can be either carried out in the field or simulated using the seismic models of the reservoir.

A seismic dataset is stored in files in a standard exchange format, referred to as SEG-Y, defined by the Society of Exploration Geophysics. A seismic data file consists of a 3600-byte header followed by a record for each sound trace. Each record contains a 240-byte header and the sound trace. The header information stores the metadata associated with the sound trace including sound source id, receiver id, receiver location, the number of samples stored for the trace. Traces collected for a single sound source are usually stored in a single file. When numerical models are used to generate seismic data, each data file can be up to 25 Gigabytes in size and there can be thousands of data sources simulated, resulting in datasets ranging from a few terabytes to hundreds of terabytes in size. We currently have about 8TB seismic simulation data on a large storage system at Ohio Supercomputer Center, 35TB on multiple TeraGrid sites, and continue to generate additional datasets.

Seismic data can be used in creating subsurface images and predicted subsurface material properties [59]. The reservoir model can be revised by imaging and inversion of output from seismic data simulations. Imaging analysis requires that subsets of seismic data be selected based on, for example, the type of sensor in a recording array and for each of a suite of sources.

2.2. Digital Microscopy Imaging

High resolution digitized microscopy enables analysis of digital mouse placenta slides to carry out quantitative examination of phenotypes and measurements of tissue structure within the placenta. One of the important components of this process is the segmentation of each image into regions of tissue layers. In our earlier work [46], we describe an approach to segment the labyrinth layer in the placenta from the rest of the image. This algorithm is based on the observation that the labyrinth layer (1) has a higher red blood cell (RBC) density compared to the spongiotrophoblast and glycogen layers and (2) is elongated in shape, forming a confined strip in the image.

The algorithm consist of mainly four data processing steps: 1) *RedFinder* detects red pixels that belong to red blood cell (RBC), 2) *Counter* identifies red pixels in the regions with high RBC density, 3) *Histogram Aggregation* collects all local histograms and aggregates them into the global histogram, 4) *PCA* computes the principal direction of the high RBC density regions, and determine the bounding box for the labyrinth layer. The steps of the algorithm algorithm has been has been implemented as pipelined data processing operations using DataCutter and Mobius frameworks [46].

Mouse brain offers a convenient model for studying neuroscience. The mouse branch of BIRN [15] aims to create an integrated anatomical atlas, which provides spatial correlation for datasets ranging from CT, MR, bright-field microscopy, confocal microscopy, as well as molecular data such as genomic and protenomic data. The atlas also serves as a visual query system.

One of the challenges in creating and integrating such an atlas is in the amount of data one must work with. For example, a single image generated by a 2-photon confocal microscope can be up to 500GB. The microscope generates data as tiled images that are then stitched as a three-dimensional stack of image mosaics. While the individual image tiles are small (512 by 480 pixels), a whole image for a single z-plane is far larger. Current capability of the scanner and the amount of data it generates has already exceeded the capability of the existing software tools for stitching the images together.

The vast size necessitates distributed tools that allow parallel image stitching, volumetric visualization and analysis operations to occur in a Grid environment. In a recent work [47], we have developed mechanism for efficient preprocessing of very large digitized microscopy images on PC clusters to support efficient evaluation of a class of aggregation queries. The target class of queries involve Sum or Count operations on image pixels and pixel values. These queries are useful in computing the densities of various image attributes (e.g., red or blue pixels) in a given region of interest. The results of the queries can be used to detect and classify features in large images and segment the images. We develop a task- and data-parallel implementation of the operations using a component-based runtime system. This implementation enables pipelined processing of data through task parallelism and effective use of distributed memory and computing capacity through data parallelism. A key operation in preprocessing is the 2-dimensional(2D) prefix sum operation. We have proposed a *local 2D prefix sum* approach which eliminates the serialization and interprocessor communication overheads of a parallel full 2D prefix sum. However, this approach introduces extra operations during query evaluation. We have investigated under what conditions local 2D prefix sum is preferable to full 2D prefix sum. All the implementation builds on top of DataCutter framework.

3. Run-time System Support

Figure 1 depicts the overall system architecture and how the middleware components are interacting with each other. If we take the oil reservoir management application, in this architecture, Mobius can be used to support management of metadata associated with simulation runs, seismic field measurements, and analysis results. The structure of data types can be managed by the Mobius Global Model Exchange (GME). The support for data product generation (e.g., reconstruction of 3D volumes from seismic data, visualization of reservoir results) is provided by DataCutter. The STORM middleware can be used to support SQL-style select queries against large simulation datasets stored in distributed collection of files on a cluster of storage nodes. These components can be exposed to the Grid environment via OGSA-DAI service interfaces and can be accessed by clients using Grid Service protocols. In this section, we present brief summary of the middleware components involved in this architecture.

```

SELECT < Data Elements >
  FROM Dataset1, Dataset2, ..., Datasetn
  WHERE < Expression > AND < Filter(< Data Element >) >
  GROUP-BY-PROCESSOR ComputeAttribute(< Data Element >)

```

Figure 2. Database queries supported by STORM.

3.1. OGSA-DAI

The Grid has emerged as an integrated infrastructure for distributed computation [21,23]. The Open Grid Services Architecture (OGSA) [22] defines mechanisms for creating, managing, and exchanging information among entities called Grid services. The objective of the OGSA-DAI [39] initiative is to build upon the OGSA infrastructure to deliver high level data management functionality for the Grid. It defines services and interfaces that can be used by clients to specify operations on data resources and data. OGSA-DAI services can be configured and customized to expose a specific database management system.

3.2. DataCutter

DataCutter [12,14,11,13] is a component-based middleware framework [40,43,29,1,5,2,17] designed to support coarse-grain data-flow execution on heterogeneous environments. In DataCutter, application processing structure is implemented as a set of components, referred to as *filters*, that exchange data through *logical streams*. A *stream* denotes a uni-directional data flow from one filter (i.e., the producer) to another (i.e., the consumer). A filter is required to read data from its input streams and write data to its output streams only. The DataCutter runtime system supports both data- and task-parallelism. Processing, network and data copying overheads are minimized by the ability to place filters on different platforms. The filtering service of DataCutter performs all steps necessary to instantiate filters on the desired hosts, to connect all logical endpoints, and to call the filter's interface functions for processing work. Data exchange between two filters on the same host is carried out by memory copy operations, while a message passing communication layer (e.g. TCP sockets or MPI) is used for communication between filters on different hosts.

3.3. STORM

STORM [36,37] is a services-oriented framework designed to support processing of large datasets in a distributed environment. It provides basic database support for 1) *selection of the data of interest* from scientific datasets stored in files and 2) *transfer of selected data from storage nodes to compute nodes for processing*. The current implementation is based on a component infrastructure, called DataCutter [13], which supports distributed execution of networks of application-specific data processing components. Using the DataCutter runtime support, STORM can perform parallel I/O on distributed data and execute data selection and data filtering operations on heterogeneous collections of storage and compute clusters.

In order to support data subsetting on file-based datasets, STORM implements three abstractions: *virtual tables*, *select queries*, and *distributed arrays*. The first two abstractions are based on object-relational database models [51]. *SELECT* operation of the form shown in Figure 2 are supported on virtual tables. Data elements selected by the *SELECT* operation are grouped based on a computed attribute. In the figure, the < *Expression* > statement can contain value-based selections and range queries. *Filter* allows implementation of user-defined operations that are difficult to express with simple comparison operations.

The client program that carries out data processing can be a parallel program. In that case, the distribution among client nodes of the data elements returned as the result of the query can be rep-

resented as a *distributed array*. This abstraction is incorporated into the STORM framework by the *GROUP-BY-PROCESSOR* operation in the query formulation. *ComputeAttribute* is another user-defined function that generates the attribute value on which the selected data elements are grouped together based on the application specific partitioning of data elements.

3.4. Mobius

Mobius is a middleware framework [32,27,31] designed for efficient metadata and data management in dynamic, distributed environments. It provides a set of generic services and protocols to support distributed creation, versioning, management of database schemas, on-demand creation of databases, federation of existing databases, and querying of data in a distributed environment. Its services employ XML schemas to represent metadata definitions and XML documents to represent and exchange metadata instances. The role of *Global Model Exchange* (GME) service of Mobius is to ensure distributed model evolution and integrity while providing the ability for storage, retrieval, versioning, and discovery of models of all shape, complexity, and interconnectedness in a distributed environment. *Mobius Mako* is a service that exposes data resources as XML data services through a set of well-defined interfaces based on the Mako protocol. Our current Mako implementation provides support to expose XML databases that support the XMLDB API and contains an implementation of MakoDB. MakoDB is an XML database built on top of MySQL [34]. The Mako protocol defines methods for submitting, updating, removing, and retrieving XML documents. Upon submission, Mako assigns each entity a unique identifier. Documents, or subsets of XML documents, can be retrieved by specifying their unique identifier. XML documents can be removed by specifying their unique identifier or by specifying an element identifying XPath [9] expression. XML documents that reside in a Mako can be updated using XUpdate².

4. Compiler Support

Scientific datasets are typically stored as binary or character flat-files. Such *low-level* layouts enable compact storage and efficient processing. Because the use of relational or other database technologies can result in significant storage overheads and slower processing, they have typically not been very popular in most scientific communities.

The use of low-level and specialized data formats, however, makes the specification of processing much harder. Recognizing this, several ongoing projects, such as BinX and Binary Format Description (BFD) [7], are proposing machine-interpretable descriptions of binary data layouts. Data Format Definition Language (DFDL) working group under the Global Grid Forum (GGF) is trying to standardize such efforts. While such proposals can allow precise description of the datasets in a remote repository, they do not alleviate the need for detailed understanding of the formats, or the dependence of an application on a particular low-level data layout.

Using data virtualization and data services, low-level, compact, and/or specialized data formats can be hidden from the applications analyzing grid-based datasets. However, supporting data virtualization can require significant effort. For each dataset layout and abstract view that is desired, a set of data services need to be implemented. An additional challenge arises from the fact that the design and implementation of efficient data virtualization and data services oftentimes require interaction of two complementary players. The first player is the scientist who possesses a good understanding of the application, datasets, and their format, but is less knowledgeable about database and data services implementation. The second player is the database developer who is proficient in the tools and

²<http://www.xmldb.org>

techniques for efficient database and data services implementation, but has little knowledge of the specific application.

In [55], we proposed a meta-data and compiler-oriented approach to facilitate a common meeting ground for the two players and to enable automatic creation of efficient data services to support data virtualization. Specifically, we showed how a relational table like data abstraction can be supported for complex multi-dimensional scientific datasets that are resident on a cluster. By using a well-defined meta-data description language, the scientist and database developer together can describe the format of the datasets generated and used by the application. Using a compiler that can parse the meta-data description and generate code to navigate the datasets, the database developer (or the scientist) can conveniently generate data services that will serve the datasets.

In [56] we have extended our work for executing SQL-3 queries over scientific data stored as flat files. The class of queries we consider involve retrieval using Select and Where clauses, and processing with user-defined aggregate functions and group-bys.

In a more recent work [57], compiler and runtime approach has been also applied for efficient execution of multi-dimensional range queries when partial replicas of a dataset exist. A compile-time query planning strategy has been presented to select the best combination of replicas in order to minimize query execution time in a distributed environment.

5. Related Work

Grid-technologies have been employed in several large-scale, multi-institutional projects in a wide range of science and engineering domains [6,18,19,26]. GriPhyN [26] targets storage, cataloging and retrieval of large, measured datasets from large scale physical experiments. The goal is to deliver data products generated from these datasets to physicists across a wide-area network. The objective of Earth Systems Grid (ESG) [18] is to provide Grid technologies for storage, publishing, and movement of large scale data from climate modeling applications. The EUROGRID project [19] intends to develop tools for easy and seamless access to High Performance Computing (HPC) resources. The BioGrid component of the project implements the support for a uniform interface that will allow biologists and chemists to submit work to HPC facilities without having to worry about the details of running their work on different architectures. Biomedical Informatics Research Network (BIRN) (<http://www.nbirn.net>) [42] initiative focuses on support for collaborative access to and analysis of datasets generated by neuroimaging studies. The BIRN project uses the Storage Resource Broker (SRB) [44], which provides a distributed file system infrastructure, as a distributed data management middleware layer. MammoGrid [4] is a multi-institutional project funded by the European Union (EU). The objective of this project is to apply Grid middleware and tools to build a distributed database of mammograms and to investigate how it can be used to facilitate collaboration between researchers and clinicians across the EU. eDiamond [50] targets deployment of Grid infrastructure to manage, share, and analyze annotated mammograms captured and stored at multiple sites. One of the goals of MammoGrid and eDiamond is to develop and promote standardization in medical image databases for mammography and other cancer diseases. MEDIGRID [33,53] is another multi-institutional project investigating the application of Grid technologies for manipulating large medical image databases.

These large scale, multi-institutional projects share the same goal of deploying an infrastructure, building on Grid technologies, to facilitate sharing of data across institutions. In order to harness the collective power of distributed systems in a Grid environment, an array of tools and frameworks have been developed to support distributed storage, data replication, data processing, monitoring, security, and high-speed data transfers in Data and Computation Grids [25,24,44,58,13,3,52,54,10,27]. As

Grid computing has become more ubiquitous, an Open Grid Services Architecture (OGSA) [20,22] has been proposed. There are some recent efforts to develop Grid and Web services implementations of database technologies. Raman et. al. [45] discusses a number of *virtualization* services to make data management and access transparent to Grid applications. These services provide support for access to distributed datasets, dynamic discovery of data sources, and collaboration. Bell et. al. [8] develop uniform web services interfaces, data and security models for relational databases. Smith et. al. [49] address the distributed execution of queries in a Grid environment. They describe an object-oriented database prototype running on MPICH-G and Globus.

References

- [1] The ABACUS project. <http://www.cs.cmu.edu/~amiri/abacus.html>.
- [2] Martin Aeschlimann, Peter Dinda, Julio Lopez, Bruce Lowekamp, Loukas Kallivokas, and David O'Hallaron. Preliminary report on the design of a framework for distributed visualization. In *Proceedings of the International Conference on Parallel and Distributed Processing Techniques and Applications (PDPTA'99)*, pages 1833–1839, Las Vegas, NV, June 1999.
- [3] W. Allcock, 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.
- [4] R. Amendolia, F. Estrella, T. Hauer, D. Manset, D. McCabe, R. McClatchey, M. Odeh, T. Reading, D. Rogulin, D. Schottlander, and T. Solomonides. Grid databases for shared image analysis in the mammogrid project. In *The Eighth International Database Engineering & Applications Symposium (Ideas'04)*, July 2004.
- [5] Khalil Amiri, David Petrou, Gregory R. Ganger, and Garth A. Gibson. Dynamic function placement for data-intensive cluster computing. In *the USENIX Annual Technical Conference*, San Diego, CA, June 2000.
- [6] Asia Pacific BioGrid. <http://www.apgrid.org>.
- [7] R. Baxter, R. Carroll, D. J. Ecklund, B. Gibbins, D. Virdee, and Q. Wen. BinX - A Tool for Retrieving, Searching, and Transforming Structured Binary Files. Available at <http://www.edikit.org/binx/pub.htm>, 2003.
- [8] William H. Bell, Diana Bosio, Wolfgang Hoschek, Peter Kunszt, Gavin McCance, and Mika Silander. Project spitfite - towards grid web service databases. <http://www.cs.man.ac.uk/grid-db/documents.html>.
- [9] Anders Berglund, Scott Boag (XSL WG), Don Chamberlin, Mary F. Fernandez, Michael Kay, Jonathan Robie, and Jerme Simon. *XML Path Language (XPath)*. World Wide Web Consortium (W3C), 1st edition, August 2003.
- [10] Francine Berman, Andrew Chien, Keith Cooper, Jack Dongarra, Ian Foster, Dennis Gannon, Lennart Johnsson, Ken Kennedy, Carl Kesselman, John Mellor-Crummey, Dan Reed, Linda Torczon, and Rich Wolski. The GrADS Project: Software support for high-level Grid application development. *The International Journal of High Performance Computing Applications*, 15(4):327–344, November 2001.
- [11] M. Beynon, T. Kurc, A. Sussman, and J. Saltz. Design of a framework for data-intensive wide-area applications. In *Proceedings of the 9th Heterogeneous Computing Workshop (HCW2000)*, pages 116–130. IEEE Computer Society Press, May 2000.
- [12] Michael D. Beynon, Renato Ferreira, Tahsin Kurc, Alan Sussman, and Joel Saltz. DataCutter: Middleware for filtering very large scientific datasets on archival storage systems. In *Proceedings of the Eighth Goddard Conference on Mass Storage Systems and Technologies/17th IEEE Symposium on Mass Storage Systems*, pages 119–133. National Aeronautics and Space Administration, March 2000. NASA/CP 2000-209888.
- [13] Michael D. Beynon, Tahsin Kurc, Umit Catalyurek, Chialin Chang, Alan Sussman, and Joel Saltz. Distributed processing of very large datasets with DataCutter. *Parallel Computing*, 27(11):1457–1478,

October 2001.

- [14] Michael D. Beynon, Tahsin Kurc, Alan Sussman, and Joel Saltz. Optimizing execution of component-based applications using group instances. In *Proceedings of CCGrid2001: IEEE International Symposium on Cluster Computing and the Grid*, pages 56–63. IEEE Computer Society Press, May 2001.
- [15] Biomedical Informatics Research Network (BIRN). <http://www.nbirn.net>.
- [16] Umit Catalyurek, Michael D. Beynon, Chialin Chang, Tahsin Kurc, Alan Sussman, and Joel Saltz. The virtual microscope. *IEEE Transactions on Information Technology in BioMedicine*, 7(4):230–248, Dec 2003.
- [17] Common Component Architecture Forum. <http://www.cca-forum.org>.
- [18] Earth Systems Grid (ESG). <http://www.earthsystemgrid.org>.
- [19] EUROGRID. <http://www.eurogrid.org/>.
- [20] I. Foster, C. Kesselman, J. Nick, and S. Tuecke. Grid services for distributed system integration. *IEEE Computer*, 36(6):37–46, June 2002. Open Grid Services Architecture (OGSA).
- [21] Ian Foster and Carl Kesselman. *The Grid: Blueprint for a New Computing Infrastructure*. Morgan Kaufmann Publishers, San Francisco, CA, USA, second edition, 2003.
- [22] Ian Foster, Carl Kesselman, Jeffrey M. Nick, and Steven Tuecke. The physiology of the Grid: An open grid services architecture for distributed systems integration. <http://www.globus.org/research/papers/ogsa.pdf>, 2002.
- [23] Ian Foster, Carl Kesselman, and Steven Tuecke. The anatomy of the Grid: Enabling scalable virtual organization. *The International Journal of High Performance Computing Applications*, 15(3):200–222, Fall 2001.
- [24] James Frey, Todd Tannenbaum, Ian Foster, Miron Livny, and Steven Tuecke. Condor-G: A computation management agent for multi-institutional grids. In *Proceedings of the Tenth IEEE Symposium on High Performance Distributed Computing (HPDC10)*. IEEE Press, Aug 2001.
- [25] The Globus Project. <http://www.globus.org>.
- [26] Grid Physics Network (GriPhyN). <http://www.griphyn.org>.
- [27] Shannon Hastings, Stephen Langella, Scott Oster, and Joel Saltz. Distributed data management and integration: The mobius project. In *GGF Semantic Grid Workshop 2004*, pages 20–38. GGF, June 2004.
- [28] Interscope technologies. <http://www.interscopetech.com>, 2001.
- [29] Carsten Isert and Karsten Schwan. ACDS: Adapting computational data streams for high performance. In *14th International Parallel & Distributed Processing Symposium (IPDPS 2000)*, pages 641–646, Cancun, Mexico, May 2000.
- [30] T. Kurc, U. Catalyurek, X. Zhang, J. Saltz, R. Martino, M. Wheeler, M. Peszyńska, A. Sussman, C. Hansen, M. Sen, R. Seifoullaev, P. Stoffa, C. Torres-Verdin, and M. Parashar. A simulation and data analysis system for large scale, data-driven oil reservoir simulation studies. *Concurrency and Computation: Practice and Experience.*, 17(11):1441–1467, September 2005.
- [31] Stephen Langella, Shannon Hastings, Scott Oster, Tahsin Kurc, Umit Catalyurek, and Joel Saltz. A distributed data management middleware for data-driven application systems. In *Proceedings of 2004 IEEE International Conference on Cluster Computing*, September 2004.
- [32] The Mobius Project. <http://www.projectmobius.org>.
- [33] J. Montagnat, V. Breton, and I. E. Magnin. Using grid technologies to face medical image analysis challenges. In *BioGrid'03, The 3rd International Symposium on Cluster Computing and the Grid (CCGrid 2003)*, pages 588–593, May 2003.
- [34] MySQL Database. <http://www.mysql.com/>.
- [35] Sivaramakrishnan Narayanan, Umit Catalyurek, Tahsin Kurc, Vijay S Kumar, and Joel Saltz. A runtime framework for partial replication and its application for on-demand data exploration. In *High Performance Computing Symposium (HPC 2005), SCS Spring Simulation Multiconference*, Mar 2005.
- [36] Sivaramakrishnan Narayanan, Umit Catalyurek, Tahsin Kurc, Xi Zhang, and Joel Saltz. Applying database support for large scale data driven science in distributed environments. In *Proceedings of*

- the Fourth International Workshop on Grid Computing (Grid 2003)*, pages 141–148, Phoenix, Arizona, Nov 2003.
- [37] Sivaramakrishnan Narayanan, Tahsin Kurc, Umit Catalyurek, and Joel Saltz. Database support for data-driven scientific applications in the grid. *Parallel Processing Letters*, 13(2):245–271, 2003.
 - [38] Sivaramakrishnan Narayanan, Tahsin M. Kurc, Umit V. Catalyurek, and Joel H. Saltz. Servicing seismic and oil reservoir simulation data through grid data services. In *Proceedings of VLDB Workshop Data Management in Grid 2005 (VLDB DMG'05)*, pages 98–109, Sep 2005.
 - [39] Open Grid Services Architecture Data Access and Integration (OGSA-DAI). <http://www.ogsadai.org.uk>.
 - [40] Ron Oldfield and David Kotz. Armada: A parallel file system for computational grids. In *Proceedings of CCGrid2001: IEEE International Symposium on Cluster Computing and the Grid*, Brisbane, Australia, May 2001. IEEE Computer Society Press.
 - [41] Manish Parashar, Vincent Matossian, Wolfgang Bangerth, Hector Klie, Benjamin Rutt, Tahsin M. Kurç, Ümit V. Catalyurek, Joel H. Saltz, and Mary F. Wheeler. Towards dynamic data-driven optimization of oil well placement. In Vaidy S. Sunderam, G. Dick van Albada, Peter M. A. Sloot, and Jack Dongarra, editors, *International Conference on Computational Science (2)*, volume 3515 of *Lecture Notes in Computer Science*, pages 656–663. Springer, 2005.
 - [42] S.T. Peltier and M.H. Ellisman. *The Biomedical Informatics Research Network, in The Grid, Blueprint for a New Computing Infrastructure*. 2nd Edition: Elsevier, 2003.
 - [43] Beth Plale and Karsten Schwan. dQUOB: Managing large data flows using dynamic embedded queries. In *IEEE International High Performance Distributed Computing (HPDC)*, August 2000.
 - [44] Arcot Rajasekar, Michael Wan, and Reagan Moore. MySRB & SRB - components of a data grid. In *The 11th International Symposium on High Performance Distributed Computing (HPDC-11)*, July 2002.
 - [45] Vijayshanker Raman, Inderpal Narang, Chris Crone, Laura Haas, Susan Malaika, Tina Mukai, Dan Wolfson, and Chaitan Baru. Data access and management services on grid. <http://www.cs.man.ac.uk/grid-db/documents.html>.
 - [46] M. Ribeiro, T. Kurc, T. Pan, K. Huang, U. Catalyurek, X. Zhang, S. Langella, S. Hastings, S. Oster, R. Ferreira, and J. Saltz. *Grid Computing: The New Frontier Of High Performance Computing*, 14, chapter Tools for Efficient Subsetting and Pipelined Processing of Large Scale, Distributed Biomedical Image Data. Elsevier, 2005.
 - [47] Benjamin Rutt, Vijay S. Kumar, Tony Pan, Tahsin Kurc, Umit Catalyurek, Yujun Wang, and Joel Saltz. Distributed out-of-core preprocessing of very large microscopy images for efficient querying. In *The 2005 IEEE International Conference on Cluster Computing*, Boston, MA, Sep 2005.
 - [48] J. Saltz, U. Catalyurek, T. Kurc, M. Gray, S. Hastings, S. Langella, S. Narayanan, R. Martino, S. Bryant, M. Peszynska, M. Wheeler, A. Sussman, M. Beynon, C. Hansen, D. Stredney, , and D. Sessanna. Driving scientific applications by data in distributed environments. In *Dynamic Data Driven Application Systems Workshop, held jointly with ICCS 2003*, Melbourne, Australia, June 2003.
 - [49] Jim Smith, Anastasios Gounaris, Paul Watson, Norman W. Paton, Alvaro A.A. Fernandes, and Rizos Sakellariou. Distributed query processing on the grid. <http://www.cs.man.ac.uk/grid-db/documents.html>.
 - [50] A. Solomonides, R. McClatchey, M. Odeh, M. Brady, M. Mulet-Parada, D. Schottlander, and S.R Amendolia. Mammogrid and ediamond: Grids applications in mammogram analysis. In *Proceedings of the IADIS International Conference: e-Society 2003*, pages 1032–1033, 2003.
 - [51] Michael Stonebraker and Paul Brown. *Object-Relational DBMSs, Tracking the Next Great Wave*. Morgan Kaufman Publishers, Inc., 1998.
 - [52] D. Thain, J. Basney, S. Son, and Miron Livny. Kangaroo approach to data movement on the grid. In *Proceedings of the Tenth IEEE Symposium on High Performance Distributed Computing (HPDC10)*, 2001.
 - [53] T. Tweed and S. Miguet. Medical image database on the grid: Strategies for data distribution. In *HealthGrid'03*, pages 152–162, January 2003.

- [54] S. Vazhkudai, S. Tuecke, and I. Foster. Replica selection in the globus data grid. In *International Workshop on Data Models and Databases on Clusters and the Grid (DataGrid 2001)*. IEEE Computer Society Press, 2001.
- [55] Li Weng, Gagan Agrawal, Umit Catalyurek, Tahsin Kurc, Sivaramakrishnan Narayanan, and Joel Saltz. An approach for automatic data virtualization. In *Proceedings of the Thirteen International Symposium on High Performance Distributed Computing (HPDC-13)*, Honolulu, HI, Jun 2004. IEEE Press.
- [56] Li Weng, Gagan Agrawal, Umit Catalyurek, and Joel Saltz. Supporting sql-3 aggregations on grid-based data repositories. In *Proceedings of the 17th International Workshop on Languages and Compilers for Parallel Computing*, 2004.
- [57] Li Weng, Umit Catalyurek, Tahsin Kurc, Gagan Agrawal, and Joel Saltz. Servicing range queries on multidimensional datasets with partial replicas. In *Proceedings of the 5th IEEE/ACM International Symposium on Cluster Computing and the Grid (CCGrid 2005)*, May 2005.
- [58] R. Wolski, N. Spring, and J. Hayes. The network weather service: A distributed resource performance forecasting service for metacomputing. *Journal of Future Generation Computing Systems*, 15(5-6):757–768, 1999.
- [59] Xi Zhang, Benjamin Rutt, Umit Catalyurek, Tahsin Kurc, and Joel Saltz. Supporting scalable and distributed data subsetting and aggregation in large-scale seismic data analysis. *The Journal of High Performance Computing Applications*, 2006. to appear.

