

DOI:10.1145/1743546.1743568

Needed are generic, rather than one-off, DBMS solutions automating storage and analysis of data from scientific collaborations.

**BY ANASTASIA AILAMAKI, VERENA KANTERE,
AND DEBABRATA DASH**

Managing Scientific Data

DATA-ORIENTED SCIENTIFIC PROCESSES depend on fast, accurate analysis of experimental data generated through empirical observation and simulation. However, scientists are increasingly overwhelmed by the volume of data produced by their own experiments. With improving instrument precision and the complexity of the simulated models, data overload promises to only get worse. The inefficiency of existing database management systems (DBMSs) for addressing the requirements of scientists has led to many application-specific systems. Unlike their general-purpose counterparts, these systems require more resources, hindering reuse of knowledge. Still, the data-management community aspires to general-purpose scientific data management. Here, we explore the most important requirements of such systems and the techniques being used to address them.

Observation and simulation of phenomena are keys for proving scientific theories and discovering facts of

nature the human brain could otherwise never imagine. Scientists must be able to manage data derived from observations and simulations. Constant improvement of observational instruments and simulation tools give modern science effective options for abundant information capture, reflecting the rich diversity of complex life forms and cosmic phenomena. Moreover, the need for in-depth analysis of huge amounts of data relentlessly drives demand for additional computational support.

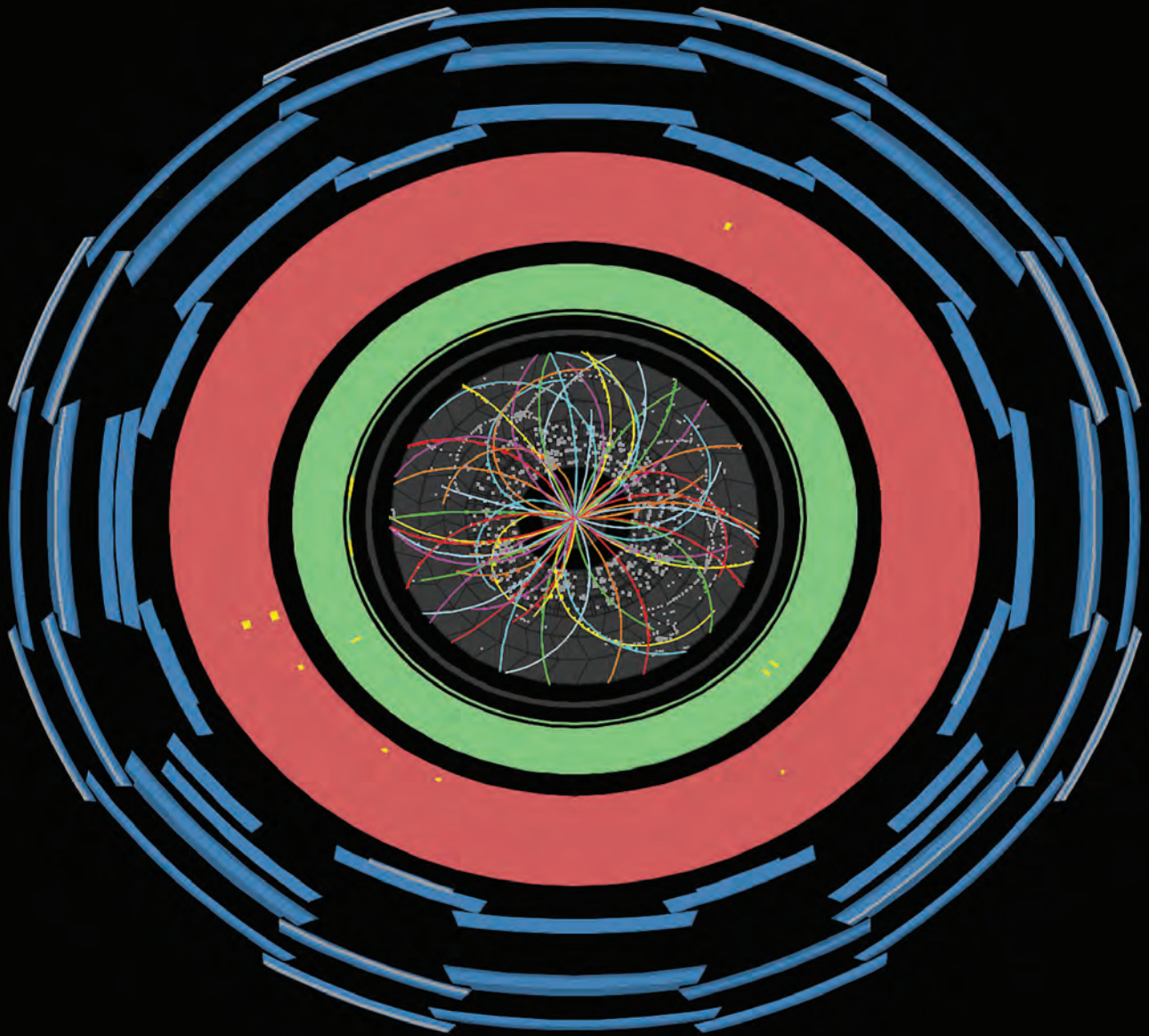
Microsoft researcher and ACM Turing Award laureate Jim Gray once said, “A fourth data-intensive science is emerging. The goal is to have a world in which all of the science literature is online, all the science data is online, and they interoperate with each other.”⁹ Unfortunately, today’s commercial data-management tools are incapable of supporting the unprecedented scale, rate, and complexity of scientific data collection and processing.

Despite its variety, scientific data does share some common features:

- ▶ Scale usually dwarfing the scale of transactional data sets;
- ▶ Generated through complex and interdependent workflows;
- ▶ Typically multidimensional;
- ▶ Embedded physical models;
- ▶ Important metadata about experiments and their provenance;
- ▶ Floating-point heavy; and
- ▶ Low update rates, with most updates append-only.

» key insights

- **Managing the enormous amount of scientific data being collected is the key to scientific progress.**
- **Though technology allows for the extreme collection rates of scientific data, processing is still performed with stale techniques developed for small data sets; efficient processing is necessary to be able to exploit the value of huge scientific data collections.**
- **Proposed solutions also promise to achieve efficient management for almost any other kind of data.**



Result of seven-trillion-electronvolt collisions (March 30, 2010) in the ATLAS particle detector on the Large Hadron Collider at CERN, hunting for dark matter, new forces, new dimensions, the Higgs boson, and ultimately a grand theory to explain all physical phenomena.

Persistent common requirements for scientific data management include:

- ▶ Automation of data and metadata processing;
- ▶ Parallel data processing;
- ▶ Online processing;
- ▶ Integration of multifarious data and metadata; and
- ▶ Efficient manipulation of data/metadata residing in files.

Lack of complete solutions using commercial DBMSs has led scientists in all fields to develop or adopt application-specific solutions, though some have been added on top

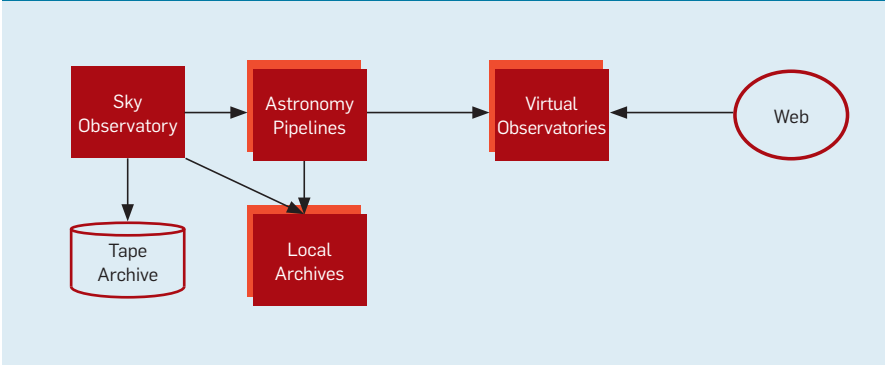
of commercial DBMSs; for example, the Sloan Digital Sky Survey (SDSS-1 and SDSS-2; <http://www.sdss.org/>) uses SQL Server as its backend. Moreover, the resulting software is typically tightly bound to the application and difficult to adapt to changes in the scientific landscape. Szalay and Blakeley⁹ wrote, “Scientists and scientific institutions need a template and best practices that lead to balanced hardware architectures and corresponding software to deal with these volumes of data.”

Despite the challenges, the data-management research community

continues to envision a general-purpose scientific data-management system adapting current innovations: parallelism in data querying, sophisticated tools for data definition and analysis (such as clustering and SDSS-1), optimization of data organization, data caching, and replication techniques. Promising results involve automated data organization, provenance, annotation, online processing of streaming data, embedded complex data types, support for declarative data, process definition, and incorporation of files into DBMSs.

Scientific databases cover a wide

Figure 1. Workflow of SDSS data.



scope, with notable demand for high performance and data quality. Scientific data ranges from medical and biological to community science and from large-scale institutional to local laboratories. Here, we focus on the big amounts of data collected or produced by instruments archived in databases and managed by DBMSs. The database community has expertise that can be applied to solve the problems in existing scientific databases.

Observation and Simulation

Scientific data originates through observation and/or simulation.¹⁶ Observational data is collected through detectors; input is digitized, and output is raw observational data. Simulation data is produced through simulators that take as input the values of simulation parameters. Both types of data are

often necessary for scientific research on the same topic; for instance, observational data is compared with simulation data produced under the same experimental setup. Consider three examples, one each for observational, simulation, and combined:

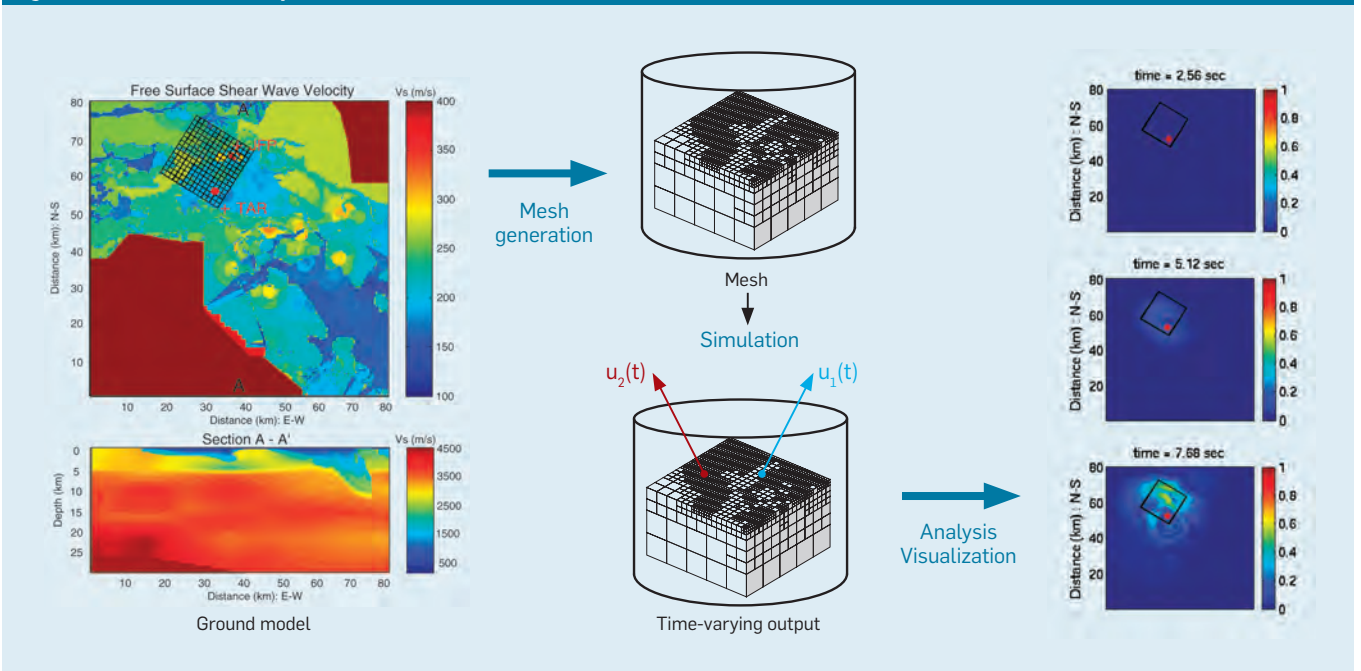
Observational scientific data. The SDSS located at Ohio State University and Johns Hopkins University, is a long-running astronomy project. Since 2000, it has generated a detailed 3D map of about 25% of the sky (as seen from Earth) containing millions of galaxies and quasars. One reason for its success is its use of the SQL Server DBMS. The SDSS uses the telescope at Apache Point Observatory, NM, to scan the sky at regular intervals to collect raw data. Online processing is done on the data to detect the stars and galaxies in the region. This online

processing also helps detect and fine-tune the telescope's alignment. The image data is then recorded onto tape for archival purposes.

The tapes are physically mailed to a processing center at Fermilab in Batavia, IL, to be processed through automated software pipelines to identify celestial objects. Many astronomical-processing-software pipelines process the data in parallel. The output of this processing, along with the image data, is stored in the local archives at Fermilab. The metadata generated from the processing pipelines is converted to relational format and stored in a MS-SQL Server database. Astronomers closely associated with the SDSS access the data from the local archives (see Figure 1).

The data is also published (publicly) once every two years through virtual observatories (<http://www.sdss.org/dr7>, the final release of the SDSS-II project) by running SQL on the database at the observatories or downloading the entire database over the Internet while running the queries locally. The SDSS project began providing public data sets in 2002 with three such observatories located at the Space Telescope Science Institute in the U.S., the National Astronomical Observatory of Japan, and the Max Planck Institute for Astrophysics in Germany.

Figure 2. Workflow of earthquake-simulation data.



The schema of SDSS data includes more than 70 tables, though most user queries focus on only a few of them, referring, as needed, to spectra and images. The queries aim to spot objects with specific characteristics, similarities, and correlations. Patterns of query expression are also limited, featuring conjunctions of range and user-defined functions in both the predicate and the join clause.

Simulation scientific data. Earth science employs simulation models to help predict the motion of the ground during earthquakes. Ground motion is modeled with an octree-based hexahedral mesh¹⁹ produced by a mesh generator, using soil density as input (see Figure 2). A “solver” tool simulates the propagation of seismic waves through the Earth by approximating the solution to the wave equation at each mesh node. During each time step, the solver computes an estimate of each node velocity in the spatial directions, writing the results to the disk. The result is a 4D spatio-temporal earthquake data set describing the ground’s velocity response. Various types of analysis can be performed on the data set, employing both time-varying and space-varying queries. For example, a user might describe a feature in the ground-mesh, and the DBMS finds the approximate location of the feature in the simula-

tion data set through multidimensional indexes.

Combined simulation and observational data. The ATLAS experiment (<http://atlas.ch/>), a particle-physics experiment in the Large Hadron Collider (<http://lhc.web.cern.ch/lhc/>) beneath the Swiss-French border near Geneva, is an example of scientific data processing that combines both simulated and observed data. ATLAS intends to search for new discoveries in the head-on collision of two highly energized proton beams. The entire workflow of the experiment involves petabytes of data and thousands of users from organizations the world over (see Figure 3).

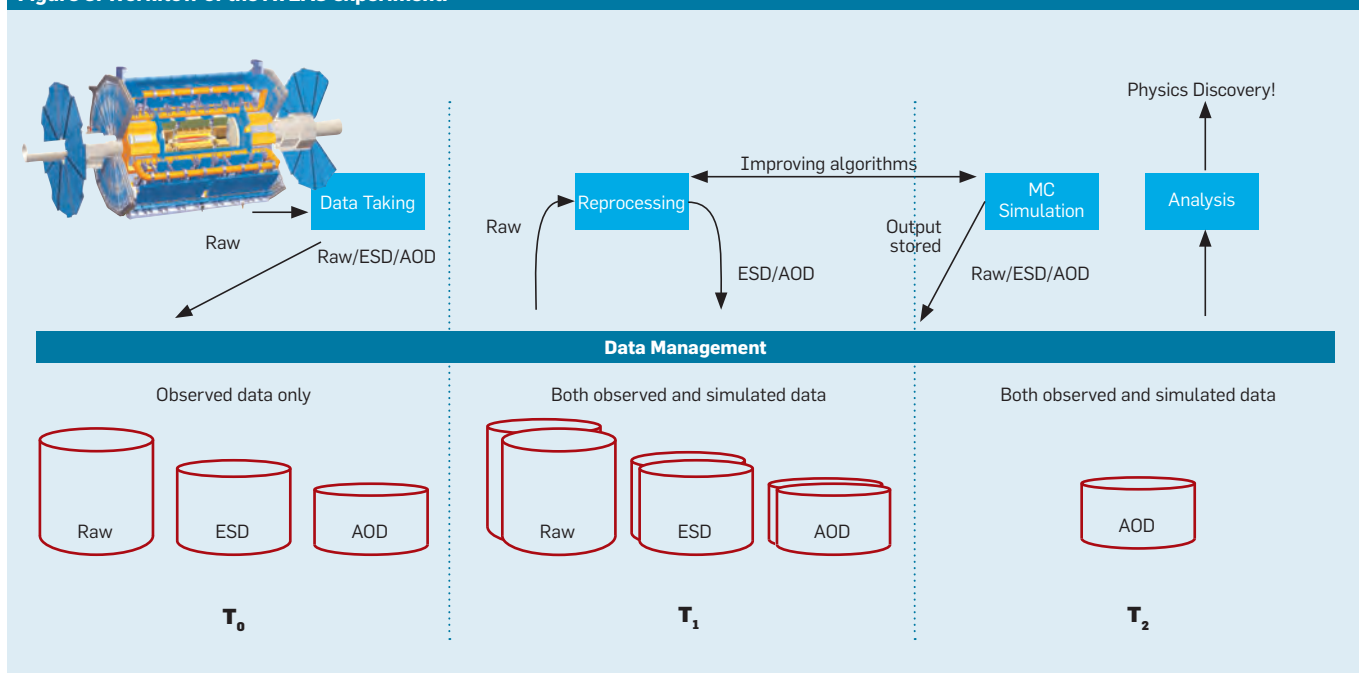
We first describe some of major ATLAS data types: The raw data is the direct observational data of the particle collisions. The detector’s output rate is about 200Hz, and raw data, or electrical signals, is generated at about 320MB/sec, then reconstructed using various algorithms to produce event summary data (ESD). ESD has an object-oriented representation of the reconstructed events (collisions), with content intended to make access to raw data unnecessary for most physics applications. ESD is further processed to create analysis object data (AOD), a reduced event representation suitable for user analysis. Data volume decreases gradually from raw to ESD

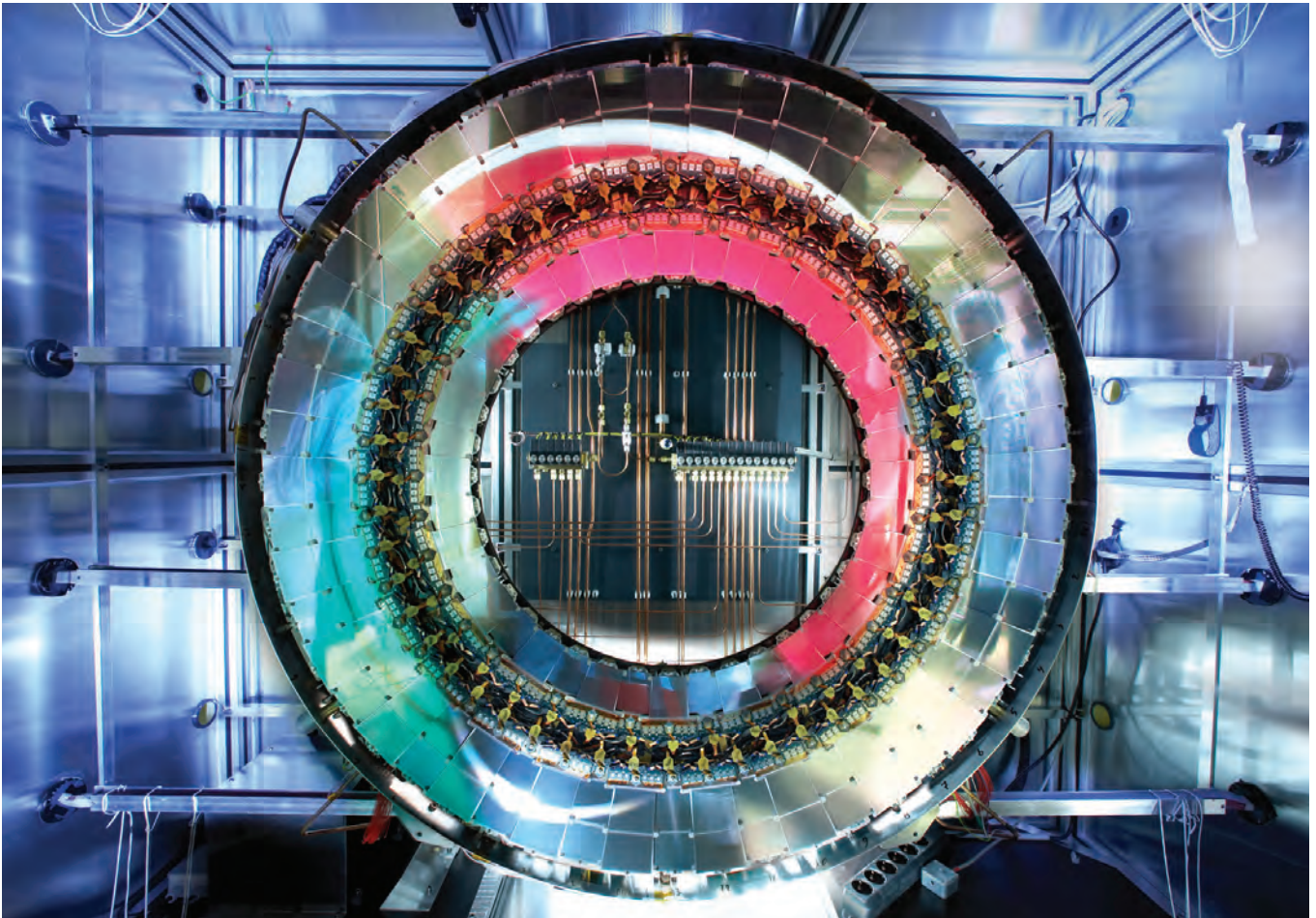
to AOD. Another important data type is tag data, or event-level metadata, stored in relational databases, designed to support efficient identification and selection of events of interest to a given analysis.

Due to the complexity of the experiment and the project’s worldwide scope, participating sites are divided into multiple layers. The Tier-0 layer is a single site—CERN itself—where the detector is located and the raw data is collected. The first reconstruction of the observed electrical signals into physics events is also done at CERN, producing ESD, AOD, and tag data. Tier-1 sites are typically large national computing centers that receive replicated data from the Tier-0 site. Tier-1 sites are also responsible for reprocessing older data, as well as for storing the final results from Monte Carlo simulations at Tier-2 sites. Tier-2 sites are mostly institutes and universities providing computing resources for Monte Carlo simulations and end-user analysis. All sites have pledged computing resources, though the vast majority is not dedicated to ATLAS or to high-energy physics experiments.

The Tier-0 site is both computation- and storage-intensive, since it stores the raw data and performs the initial event reconstruction. It also serves data to the Tier-1 sites, with aggregate sustained transfer rates for raw, ESD,

Figure 3. Workflow of the ATLAS experiment.





Disk with silicon sensors as an endcap of the ATLAS silicon strip detector in its testbox at the Nationaal Instituut voor Subatomaire Fysica, Amsterdam, The Netherlands.

and AOD in excess of 1GB/sec over the WAN or dedicated fiber. Tier-1 sites are also computation- and storage-intensive, since they store new data samples while permanently running reconstruction of older data samples with newer algorithms. Tier-2 sites are primarily CPU-intensive, since they generally run complex Monte Carlo simulations and user analyses while only transiently storing data, with archival copies of interesting data kept at Tier-1s sites.

The ATLAS experimental workflow involves a combination of observed and simulated data, as outlined in Figure 3. The “Data Taking” component consumes the raw data and produces ESD, AOD, and tags that are replicated to a subset of the Tier-1s sites and, in the case of AOD, to all Tier-1s and Tier-2s, where each Tier-2 site receives AOD only from its parent Tier-1 site. The “Reprocessing” component at each Tier-1 site reads older data and produces new versions of ESD and

AOD data, which is sent to a subset of other Tier-1s sites and, in the case of AOD, to all sites. The primary difference between the first reconstruction at the Tier-0 site and later reconstructions at the Tier-1 sites is due to better understanding of the detector’s behavior. Simulated data is used for this reconstruction.

Simulated data, using Monte Carlo techniques, is required to understand the behavior of the detector and help validate physics algorithms. The ATLAS machine is physically very large and complex, at 45 meters \times 25 meters and weighing more than 7,000 tons, including more than 100 million electronic channels and 3,000 kilometers of cable. A precise understanding of the machine’s behavior is required to fine-tune the algorithms that process its data and reconstruct simulation data from the observed electrical signals. This is the role of the Monte Carlo simulations, using large statistics and enough data to compensate

and understand the machine’s bias. These simulations are run at Tier-2 sites by the “MC Simulation” component in Figure 3, with results sent to a Tier-1 site.

Finally, the “User Analysis” component aims to answer specific physics questions, using a model built with specific analysis algorithms. This model is then validated and improved against Monte Carlo data to compensate for the machine’s bias, including background noise, and validated against real, observed data, eventually testing the user’s hypothesis.

Beyond observational and simulation data and its hybrids, researchers discuss special cases of “information-intensive” data.¹⁶ Sociology, biology, psychology, and other sciences apply research on heterogeneous data derived from both observation and simulation under various conditions. For example, in biology, research is conducted on data collected by biologists under experimental conditions

intended for a variety of purposes. Scientists involved in combustion chemistry, nanoscience, and the environment must perform research on data related to a variety of phenomena concerning objects of interest ranging from particles to devices and from living organisms to inorganic substances. Information-intensive data is characterized by heterogeneity, in both representation and the way scientists use it in the same experiment. Management of such data emphasizes logical organization and description, as well as integration.

Independent of the data type characterizing particular scientific data, its management is essentially divided into coarse phases: workflow management, management of metadata, data integration, data archiving, and data processing:

Workflow management. In a simple scenario, a scientific experiment is performed according to a workflow that dictates the sequence of tasks to be executed from the beginning until the end of the experiment. The tasks coarsely define the manner and means for implementing the other phases of data management: data acquisition, metadata management, data archiving, and data processing. Task deployment may be serial or parallel and sequential or looping. Broad and long-lasting experiments encompass sets and hierarchies of partial experimental studies performed according to complex workflows. In general, many experiments could share a workflow, and a workflow could use results from many experiments.

Scientific workflow management systems have been a topic for much research over the past two decades, involving modeling and enactment¹⁵ and data preservation.¹² Numerous related scientific products are used in the sciences.

Metadata management. Raw data is organized in a logically meaningful way and enriched with respective metadata to support the diagnosis of unexpected (independent) reinvestigation or reinterpretation of results, ideally automatically. Metadata includes information on data acquisition (such as parameters of the detectors for observational data and simulation for simulation data), as

well as administrative data about the experiments, data model, and measurement units. Other kinds of metadata are extracted from raw data, possibly through ontologies. Metadata may also denote data relationships and quality. Annotating the data, all metadata (accumulated or extracted) is critical to deducing experimental conclusions. The metadata and workflows often complement one another, necessitating combined management.⁷

Data and process integration. Data and its respective metadata may be integrated such that they can be manipulated as a unit. Moreover, newly collected data may be integrated with historic data representing different versions or aspects of the same experiment or belonging to different experiments in the same research track. The quality of the data, as well as its semantic interpretation, is crucial for data integration. Data quality can be achieved through data cleansing; semantic integration can be achieved through ontologies. Beyond data integration, process integration is often needed to simplify the overall flow of the experiment and unify partial results. For example, integration may be necessary or desirable for data mining algorithms that facilitate feature extraction and provenance. Process integration might necessitate creation or customization of middleware allowing for interoperability among different procedures and technologies. Automatic integration of data and processes is highly desirable for ease of use but also because querying information as a unit allows parallel and online processing of partial results. Moreover, scientists want transparent access to all data. Automatic integration assumes customizable tools implementing generic integration solutions appropriate for scientific data. A notably challenging task is how to identify commonalities in scientific experiments and data in order to create integration tools.

Data archiving. After they've met the expected standards of data content, scientists archive the data so other scientists are able to access and use it in their own research. The data must first be stored using robust, reliable storage technology, a mandatory

task, especially for raw observational data derived from experiments that cannot be replayed or replayed only at prohibitively high cost. The complete data set is usually archived on tape, with selected parts stored on disks. Portions of the data might also have to be cached temporarily during migration between tape and disk or between various computers.

Beyond archiving master copies, data replication on multiple sites may be necessary to accommodate geographically dispersed scientists. All the challenges of distributing and replicating data management come into play when coordinating the movement of large data volumes. Efforts are under way to manage these replication tasks automatically.⁴

Archiving scientific data is usually performed by storing all past data versions, as well as the respective metadata (such as documentation or even human communication like email). Nevertheless, the problem of organizing archives relates to the general research problem of data versioning, so solutions to the versioning problem can be adapted to archiving scientific data. Representative versioning solutions (such as the concurrent versions system) compute differences between sequential versions and use the differences for version reconstruction. Recent proposals targeting scientific data³ exploit the data's hierarchical structure in order to summarize and merge versions.

Data processing. Data is analyzed to extract evidence supporting scientific conclusions, ultimately yielding research progress. Toward this end, the data must undergo a series of procedures specified by scientists in light of the goal of their respective experiments. These procedures usually involve data clustering, mining, and lineage, leading to the inference of association rules and abnormalities, as well as to computation for feature identification and tracking.

Data analysis is often tightly correlated with data visualization, especially when it comes to simulation data. Scientists want a visual representation of the data to help them recognize coarse associations and abnormalities. Interleaving the steps involved in visualization and analysis yields the

right data regions for testing hypotheses and drawing conclusions. The key to efficient data processing is a carefully designed database and is why automated physical database design is the subject of recent research (discussed in the following section). In addition, there is an imminent need for online data processing (discussed in the second following section).


Automation

Errors and inefficiencies due to human-handled physical database design are common in both metadata management and data processing. Much recent research has focused on automating procedures for these two phases of scientific data management.


Metadata management. Metadata processing involves determining the data model, annotations, experimental setup, and provenance. The data model can be generated automatically by finding dependencies between different attributes of data.¹⁰ However, experimenters typically determine the model since this is a one-time process, and dependencies $A=\pi r^2$ are easily identified at the attribute level.

Annotations are meta-information about the raw scientific data and especially important if the data is not numeric. For example, annotations are used in biology and astronomy image data. Given the vast scale of scientific data, automatically generating these annotations is essential. Current automated techniques for gathering annotations from documents involve machine-learning algorithms, learning the annotations through a set of pre-annotated documents.¹⁴ Similar techniques are applied to images and other scientific data but must be scaled to terabyte or petabyte scale. Once annotations are built, they can be managed through a DBMS.

Experimental setups are generally recorded in notebooks, both paper and electronic, then converted to query-able digital records. The quality of such metadata is typically enforced through policies that must be as automated as possible. For example, when data is collected from instruments, instrument parameters can be recorded automatically in a database. For manually generated data, the policies must be enforced automatically.



Not having to read from the disk and write computation results back saves hours to days of scientific work, giving scientists more time to investigate the data.



For the ATLAS experiment, the parameters of the detectors, as well as the influence of external magnetic devices and collider configurations, are all stored automatically as metadata. Some policies can be enforced automatically through a knowledge base of logical statements; the rest can be verified through questionnaires. Many commercial tools are available for validating policies in the enterprise scenario, and the scientific community can borrow technology from them to automate the process (<http://www.compliancehome.com/>).

Provenance data includes experimental parameters and task history associated with the data. Provenance can be maintained for each data entry or for each data set. Since workload management tracks all tasks applied to the data, it can automatically tag it with task information. Hence, automating provenance is the most straightforward of the metadata-processing automation tasks. The enormous volume of automatically collected metadata easily complicates the effort to identify the relevant subset of metadata to the processing task in hand. Some research systems are capable of automatically managing a DBMS's provenance information.²

Data processing. Data processing depends on how data is physically organized. Commercial DBMSs usually offer a number of options for determining how to store and access it. Since scientific data might come in petabyte-scale quantities and many scientists work on the same data simultaneously, the requirements for efficient data organization and retrieval are demanding. Furthermore, the data might be distributed or replicated in multiple geographically dispersed systems; hence, network resources play an important role in facilitating data access. Possibly hundreds or thousands of scientists could simultaneously query a petabyte-scale database over the network, requiring more than 1GB/sec bandwidth.

To speed data access, the database administrator might have to tune several parameters, changing the data's logical design by normalizing the data or its physical design. The logical design is determined by the data model in the metadata-processing

phase of a science experiment. The physical design determines optimal data organization and location, caching techniques, indexes, and other performance-enhancing techniques. All depend on the data-access pattern, which is dynamic, hence, it changes much more frequently than logical design; physical design automation is therefore critical for efficient data processing.

Considering the number of parameters involved in the physical design of a scientific database, requiring the database administrator to specify and optimize the parameters for all these techniques is unreasonable. Data storage and organization must be automated.

All DBMSs today provide techniques for tuning databases. Though the provision of these techniques is a step in the right direction, existing tools are insufficient for four main reasons:

Precision. They require the query workload to be static and precise;

Relational databases. They consider only auxiliary structures to be built on relational databases and do not consider other types of data organization;

Static database. They assume a static database, so the statistics in the database are similar to the statistics at the time the tool is run; and

Query optimizer. They depend on the query optimizer to direct their search algorithms, making them slow for large workloads.

Recent database research has addressed these inherent DBMS limitations. For example, some techniques do not require prespecifying the workload,¹ and others make the cost model more efficient, enabling more thorough search in the data space.¹⁸ However, they also fall short in several areas; for example, they are not robust enough to change database statistics and do not consider data organization other than relational data. Likewise, data-organization methods for distributed data and network caches are nascent today. Automatically utilizing multiple processing units tuned for data-intensive workloads to scale the computation is a promising research direction, and systems (such as Gray-Wulf²⁴) apply this technique to achieve scalability.

Physical and logical design-automation tools must consider all parameters and suggest optimal organization. The tools must be robust to small variations in data and query changes, dynamically suggesting changes in the data organization when the query or data changes significantly.

Online Processing

Most data-management techniques in the scientific community are offline today; that is, they provide the full result of the computation only after processing an entire data set. However, the ever-growing scale of scientific data volume necessitates that even simple processes, one-time data movement, checksum computation, and verification of data integrity might have to run for days before completion.

Simple errors can take hours to be noticed by scientists, and restarting the process consumes even more time. Therefore, it is important that all processing of scientific data be performed online. Converting the processes from offline to online provides the following benefits:

Efficiency. Many operations can be applied in a pipeline manner as data is generated or move around. The operations are performed on the data when already in memory, which is much closer to the CPU than to a disk or tape. Not having to read from the disk and write computation results back saves hours to days of scientific work, giving scientists more time to investigate the data.

Feedback. Giving feedback to the operations performed on the scientific data is important, because it allows scientists to plan their analysis according to the progress of the operation. Modern DBMSs typically lack a progress indicator for queries, hence scientists running queries or other processes on DBMSs are typically blind to the completion time of their queries. This blindness may lead to canceling the query and issuing a different one or abandoning the DBMS altogether. DBMSs usually allow a query issuer to compute the “cost” of a query in a unit specific to the DBMS. This cost is not very useful to scientists, since it doesn’t correspond to actual running time or account for the complete set of resources (such as memory size, band-

width, and operation sharing) available to the DBMS for running the query. Operations, including querying/ updating data, should thus provide real-time feedback about the query progress to enable scientists to better plan their experiments.

Debugging. Scientific data is typically processed on multiprocessor systems, as scientific applications are often parallelizable and computation can thus scale to data volume. However, it is nearly impossible to detect all the problems of a parallel program at development time. Using source debuggers for parallel programming is infeasible, since debuggers change the timing of the programs, thereby hiding many problems. Debugging becomes even more difficult when programs execute complex tasks (such as queries with user-defined functions).

Some research DBMS prototypes provide feedback on query progress,¹³ though they are not yet incorporated into commercial systems, so the benefits are still not available to scientists. Similarly, tools that provide online visualization of progress for specific simulations are not generic enough for a variety of scientific experiments.

Computational steering. Building complex simulations is a challenge even in uniprocessor systems. After building them, the submitters-scientists often boot on suboptimal parameters, unaware that they’re indeed relying on suboptimal parameters until the entire simulation is over. Therefore, online processing, combined with online visualization, can simultaneously help debug such programs and parameters. The system’s operations should allow an observer to generate snapshots of the simulations or operations and, if possible, control the simulation to remove potential problems. Manual intervention in an otherwise automatic process is called “computational steering”; for example, in parallel programs, observers could decide which deadlocks to break or when simulations should change a parameter on the fly.

Software for computational steering includes the scientific programming environment SciRun (<http://www.sci.utah.edu/cibc/software/106-scirun.html>). Nevertheless, software must support simulations with pet-

abytes of data and execution of complex tasks. For software designers, it may sometimes be beneficial to model the simulations and data processing as event generators, using streaming and complex event-processing techniques to summarize data operations with little overhead or controllable accuracy guarantees.

Data and Process Integration

Large-scale experiments organized by scientists collect and process huge amounts of raw data. Even if the original data is reorganized and filtered in a way that keeps only the interesting parts for processing, these interesting parts are still big. The reorganized data is augmented with large volumes of metadata, and the augmented reorganized data must be stored and analyzed.

Scientists must collaborate with computer engineers to develop custom solutions supporting data storage and analysis for each experiment. In spite of the effort involved in such collaborations, the experience and knowledge gained this way is not generally disseminated to the wider scientific community or benefit next-generation experimental setups. Computer engineers must therefore develop generic solutions for storage and analysis of scientific data that can be extended and customized to reduce the computing overhead of time-consuming collaborations. Developing generic solutions is feasible, since many low-level commonalities are available for representing and analyzing experimental data.

Management of generic physical models. Experimental data tends to have common low-level features not only across experiments of the same science, but across all sciences. For example, reorganized raw data enhanced with metadata usually involves complex structures that fit the object-oriented model. Scientific data representation benefits from inheritance and encapsulation, two fundamental innovations of the object-oriented data model.

Beyond its complexity in terms of representation, scientific data is characterized by complex interdependencies, leading to complex queries during data processing and analysis. Even

though the object-oriented model is suitable for the representation of scientific data, it cannot efficiently optimize and support complex queries.

Nevertheless, most scientific data is represented by objects with strong commonalities with respect to their structural elements. DBMSs must be extended so they manage the common structural elements of scientific data representations as generic database objects, and database support for these objects must include efficient ways to store and index data.

Experimental data derived from simulations is frequently represented as meshes ranging from structured to unstructured and consisting of tetrahedra, hexahedra, or n -facets cells. For example, an earthquake simulation data set may be represented as an unstructured hexahedral mesh. A typical volume of earth, say, $100\text{km} \times 100\text{km} \times 30\text{km}$, is represented by a mesh consisting of roughly one billion nodes and one billion elements requiring about 50GB of storage; such a mesh is capable of resolving seismic waves up to 2Hz. Scientific data management would benefit greatly if DBMSs offered storage and indexing methods for meshes. Initial efforts toward supporting meshes in DBMSs were presented in research¹⁹ and commercial products.⁸

Multidimensional data also needs storage and indexing methods. Most scientific data is represented as multidimensional arrays, but support for multidimensional arrays in RDBMSs is poor. Computer engineers must produce custom solutions for manipulating multidimensional data, leading to many domain-specific data formats, including netCDF (<http://www.unidata.ucar.edu/software/netcdf/>) and HDF (<http://www.hdfgroup.org/>) for climate data; FITS (<http://heasarc.gsfc.nasa.gov/docs/heasarc/fits.html>) for astronomical data; and ROOT (<http://root.cern.ch/drupal/>) for high-energy physics data.

An experimental study⁵ showed that, even if using array primitives in RDBMSs, native file formats outperformed the relational implementation by a factor of 20 to as much as 80. Proposed scientific DBMSs^{6,23} provide multidimensional arrays as first-class types, aiming to bridge the gap be-

tween DBMSs and native files in the process. The multidimensional arrays are present in multidimensional online analytical processing (MOLAP) implementations from mainstream DBMSs that allow fast exploratory analysis of the data by pre-computing aggregations on multiple dimensions. However, MOLAP needs a significant amount of offline processing and an enormous amount of disk space to store the pre-computed aggregations, making them unsuitable for the enormous scale of scientific data. Attempts to support exploratory ad hoc OLAP queries on large data sets, including wavelets, promises to enable fast, powerful analysis of scientific data.²¹

A frequently used type of scientific data is time-and-space-based observations, meaning interesting data sets are trajectories in space and time. As trajectories are not inherently supported by databases, data points are usually stored individually and processed for line-fitting during scientific analysis. Line-fitting is a resource-consuming task and could be avoided if a DBMS inherently supported trajectories. Inherent support for trajectories is related to multidimensional array support, since trajectories are actually polylines, and each line can be approximated (fitted) by functions. Promising research results have been reported for managing trajectories.²²

DBMSs should inherently support new data types while accounting for the specialized use of the new types for representing scientific data. For example, the Hierarchical Triangular Mesh method¹¹ subdivides spherical surfaces so objects localized on a sphere can be indexed and queried efficiently. Scientific data is usually persistent, meaning it is rarely (if ever) changed or involved in complex associations. While developing support mechanisms (such as indexes) for new data types, priority must go to search rather than to update efficiency. The persistence of scientific data alleviates a major requirement, making it possible to develop efficient indexes for the new types.

Management of generic data processing. Scientific data processing differs from experiment to experiment and discipline to discipline. No matter how wide the scope of processing

and overall heterogeneity, processing frequently encompasses generic procedures.


It is most common that scientific data is searched in order to find interesting regions with respect to prespecified characteristics, data regions, and abnormalities. Moreover, metadata processing consists of data annotation, as well as feature extraction. Data tagged with parameter values refers to the condition of the experiment and is mined to deduce common characteristics and behavior rules.

Generic processes that produce metadata (such as those just mentioned) must be supported inherently by the DBMS for all generic physical models in a parameterized manner, thus bringing processing close to the data and leading to reduced data movement and reorganization, along with efficient processing execution in the DBMS. Each generic physical model must include templates supporting the generic procedures for the model in a customizable manner. For example, the particles in the ATLAS experiment are tracked using spatio-temporal attributes. Even though the data sets are enormous, only a small amount of space and time are populated by particles. Therefore, the data sets would benefit from a generic DBMS customizable procedure supporting compression.


Scientists would benefit even more if the definition and customization of the templates could be performed using a declarative language. Such a language would give users intuitive guidance as to the specification of the customization procedure, as well as to the combination and pipelining of multiple procedures. In this way, the processing burden would be leveraged to the DBMS, and scientists would not have to function as computer engineers.

File Management

The vast majority of scientific data is stored in files and manipulated through file systems, meaning all processing, from search to computation, is performed in the content of the files. Sophisticated frameworks have been proposed to manage the files over a large number of disks, including storage resource management



DBMSs must be extended so they manage the common structural elements of scientific data representations as generic database objects, and database support for these objects must include efficient ways to store and index data.



technology (<https://sdm.lbl.gov/srm-wg/index.html>).

Existing persistent scientific data in files is huge and will not be moved to databases, even if they support efficient scientific experimentation. Moreover, the tradition in applications that manipulate scientific data files is long, and implementing the same functionality in modules that are plug-able on DBMSs needs further effort. A long tradition and the need for plug-able capabilities mean that a full-fledged querying mechanism for files, similar to DBMSs, is needed. Such a querying mechanism can be constructed in either of two ways: enhance current DBMSs so they uniformly manage both structured data and unstructured data in files; and create a management layer on top of both the DBMS and the file system to enable transparent querying of structured and unstructured data. Each approach has advantages and disadvantages.

Enhancing a DBMS to manage files and data means that all mechanisms in the system should be extended for files. Querying on files is assumed to be efficient since it would benefit from sophisticated database structures (such as indexes, autonomic database organization, and database query planning). Moreover, querying structured and unstructured data is an opportunity for tight interaction among queries and query results and refined optimization in intermediate querying steps.

Extending DBMSs to manage files is a challenge for the data-management community since it entails reconsideration of many database protocols and a total rebuild of all database procedures with new enhancements for unstructured data. Yet it is inevitable that such a breakthrough in the functionality of DBMSs will involve substantial redundancy, since a big part of database management (such as transaction management) is useless in the manipulation of scientific data. Recent research efforts seeking to integrate scientific files into DBMSs include Netlobs, a netCDF cartridge for Oracle (http://datafedwiki.wustl.edu/images/f/ff/Dews_poster_2006.ppt) and Barrodale Computing Services, Ltd., on Postgres ([JUNE 2010 | VOL. 53 | NO. 6 | COMMUNICATIONS OF THE ACM 77](http://www.</p>
</div>
<div data-bbox=)

barrodale.com/) but are not generic enough for other file formats (such as HDF and ROOT). Aiming for file manipulation, the Data Format Description Language Work Group project (<http://forge.gridforum.org/projects/dfdl-wg>) is developing an XML-based language for describing the metadata of files. However, these approaches do not provide complete support for DBMS features on the files. MapReduce-based techniques for processing data stored in files^{17,20} do not replicate DBMS functionalities and are mainly used for batch processing of files.

A management level on top of the database and file system that manages structured and unstructured data separately but transparently seems more feasible in the near future. The challenge in developing this approach is twofold. First, it is necessary to construct middleware through which users define their queries in a declarative high-level manner, but the middleware must include a mechanism that transcribes queries as input for the DBMS or file system and routes it appropriately. Second, a query mechanism dedicated to the file system must be developed; the benefit of a separate file-querying mechanism is that it includes only procedures targeted at querying, thereby avoiding implications due to complicated database mechanisms—insert, delete, update—serving database operations. However, the procedures involved in the querying mechanism must be designed and implemented from scratch, an approach that precludes uniform querying of both structured and unstructured data; it also means limited uniform query optimization and limited query-execution efficiency. Future grid middleware promises to support such integration (<http://www.ogsadai.org.uk/>), though the related research is still only nascent.

Conclusion

Scientific data management suffers from storage and processing limitations that must be overcome for scientific research to take on demanding experimentation involving data collection and processing. Future solutions promise to integrate automation, online processing, integration,

and file management, but data manipulation must still address the diversity of experimentation tasks across the sciences, the complexity of scientific data representation and processing, and the volume of collected data and metadata. Nevertheless, data-management research in all these areas suggests the inherent management problems of scientific data will indeed be addressed and solved.

Acknowledgments

We would like to express our gratitude to Miguel Branco of CERN for contributing the dataflow of the ATLAS experiment, allowing us to demonstrate a scientific application requiring both observational and simulation data. We would also like to acknowledge the European Young Investigator Award by the European Science Foundation (<http://www.esf.org/>).

References

1. Bruno, N. and Chaudhuri, S. Online autoadmin: Physical design tuning. In *Proceedings of the ACM International Conference on Management of Data* (Beijing, June 11–14). ACM Press, New York, 1067–1069.
2. Buneman, P., Chapman, A., and Cheney, J. Provenance management in curated databases. In *Proceedings of the ACM SIGMOD International Conference on Management of Data* (Chicago, June 27–29). ACM Press, New York, 2006, 539–550.
3. Buneman, P., Khanna, S., Tajima, K., and Tan, W. Archiving scientific data. In *Proceedings of the ACM SIGMOD International Conference on Management of Data* (Madison, WI, June 3–6). ACM Press, New York, 2002, 1–12.
4. Chervenak, A.L., Schuler, R., Ripeanu, M., Amer, M.A., Bharathi, S., Foster, I., Iamnitchi, A., and Kesselman, C. The Globus Replica Location Service: Design and experience. *IEEE Transactions on Parallel Distributed Systems* 20, 9 (Sept. 2009), 1260–1272.
5. Cohen, S., Hurley, P., Schulz, K.W., Barth, W.L., and Benton, B. Scientific formats for object-relational database systems: A study of suitability and performance. *SIGMOD Records* 35, 2 (June 2006), 10–15.
6. Cudre-Mauroux, P., Kimura, H., Lim, K., Rogers, J., Simakov, R., Soroush, E., Velikhov, P., Wang, D.L., Balazinska, M., Becla, J., DeWitt, D., Heath, B., Maier, D., Madden, S., Patel, J., Stonebraker, M., and Zdonik, S. A demonstration of SciDB: A science-oriented DBMS. *Proceedings of VLDB Endowment* 2, 2 (Aug. 2009), 1534–1537.
7. Davidson, S.B. and Freire, J. Provenance and scientific workflows: Challenges and opportunities. In *Proceedings of the ACM SIGMOD International Conference on Management of Data* (Vancouver, B.C., June 9–12). ACM Press, New York, 1345–1350.
8. Gray, J. and Thomson, D. *Supporting Finite-Element Analysis with a Relational Database Backend, Parts i–iii*. MSR-TR-2005-49, MSR-TR-2006-21, MSR-TR-2005-151. Microsoft Research, Redmond, WA, 2005.
9. Hey, T., Tansley, S., and Tolle, K. *The Fourth Paradigm: Data-Intensive Scientific Discovery*. Microsoft, Redmond, WA, Oct. 2009.
10. Ilyas, I.F., Markl, V., Haas, P., Brown, P., and Aboulmaga, A. CORDS: Automatic discovery of correlations and soft functional dependencies. In *Proceedings of the ACM SIGMOD International Conference on Management of Data* (Paris, June 13–18). ACM Press, New York, 2004, 647–658.
11. Kunszt, P.Z., Szalay, A.S., and Thakar, A.R. The hierarchical triangular mesh. In *Proceedings of the*

- MPA/ESO/MPE Workshop (Garching, Germany, July 31–Aug. 4). Springer, Berlin, 2000, 631–637.
12. Liu, D.T., Franklin, M.J., Abdulla, G.M., Garlick, J., and Miller, M. Data-preservation in scientific workflow middleware. In *Proceedings of the 18th International Conference on Scientific and Statistical Database Management* (July 3–5). IEEE Computer Society, Washington, DC, 2006, 49–58.
13. Mishra, C. and Koudas, N. A lightweight online framework for query progress indicators. In *Proceedings of the IEEE International Conference on Data Engineering* (Istanbul, Apr. 15–20). IEEE Press, 1292–1296.
14. Müller, A. and Sternberg, M. Structural annotation of the human genome. In *Proceedings of the German Conference on Bioinformatics* (Braunschweig, Germany, Oct. 7–10). German Research Center for Biotechnology, Braunschweig, 2001, 211–212.
15. Ngu, A.H., Bowers, S., Haasch, N., McPhillips, T., and Critchlow, T. Flexible scientific workflow modeling using frames, templates, and dynamic embedding. In *Proceedings of the 20th International Conference on Data Engineering and Statistical Database Management* (Hong Kong, July 9–11). Springer-Verlag, Berlin, Heidelberg, 2008, 566–572.
16. Office of Data Management Challenge. *Report from the DOE Office of Science Data Management Workshops*, Mar.–May 2004; <http://www.er.doe.gov/ascr/ProgramDocuments/Docs/Final-report-v26.pdf>
17. Olston, C., Reed, B., Srivastava, U., Kumar, R., and Tomkins, A. Pig Latin: A not-so-foreign language for data processing. In *Proceedings of the ACM SIGMOD International Conference on Management of Data* (Vancouver, B.C., June 9–12). ACM Press, New York, 2008, 1099–1110.
18. Papadomanolakis, S., Dash, D., and Ailamaki, A. Efficient use of the query optimizer for automated physical design. In *Proceedings of the 33rd International Conference on Very Large Data Bases* (Vienna, Austria, Sept. 23–27). VLDB Endowment, 2007, 1093–1104.
19. Papadomanolakis, S., Ailamaki, A., Lopez, J.C., Tu, T., O'Hallaron, D.R., and Heber, G. Efficient query processing on unstructured tetrahedral meshes. In *Proceedings of the ACM SIGMOD International Conference on Management of Data* (Chicago, June 27–29). ACM Press, New York, 2006, 551–562.
20. Pike, R., Dorward, S., Griesemer, R., and Quintan, S. Interpreting the data: Parallel analysis with Sawzall. *Scientific Programming* 13, 4 (Oct. 2005), 277–298.
21. Shahabi, C., Jahangiri, M., and Banaei-Kashani, F. ProDA: An end-to-end wavelet-based OLAP system for massive datasets. *Computer* 41, 4 (Apr. 2008), 69–77.
22. Spaccapietra, S., Parent, C., Damiani, M.L., de Macedo, J. A., Porto, F., and Vangenot, C. A conceptual view on trajectories. *Data Knowledge Engineering* 65, 1 (Apr. 2008), 126–146.
23. Stonebraker, M., Bear, C., Çetintemel, U., Cherniack, M., Ge, T., Hachem, N., Harizopoulos, S., Lifter, J., Rogers, J., and Zdonik, S.B. One size fits all? Part 2: Benchmarking studies. In *Proceedings of the Conference on Innovative Data Systems Research* (Asilomar, Jan. 7–10, 2007), 173–184.
24. Szalay, A., Bell, G., VandenBerg, J., et al. GrayWulf: Scalable clustered architecture for data-intensive computing. In *Proceedings of the Hawaii International Conference on System Sciences* (Waikoloa, Jan. 5–8). IEEE Computer Society Press, 2009, 1–10.

Anastasia Ailamaki (natassa@epfl.ch) is director of the Data-Intensive Applications and Systems Laboratory and a professor at Ecole Polytechnique Fédérale de Lausanne, Lausanne, Switzerland, and an adjunct professor at Carnegie Mellon University, Pittsburgh, PA.

Verena Kantere (verena.kantere@epfl.ch) is a postdoctoral researcher in the Data-Intensive Applications and Systems Laboratory at Ecole Polytechnique Fédérale de Lausanne, Lausanne, Switzerland.

Debabrata Dash (debabrata.dash@epfl.ch) is Ph.D student in the Data-Intensive Applications and Systems Laboratory at Ecole Polytechnique Fédérale de Lausanne, Lausanne, Switzerland.