From SkyServer to SciServer

By
ALEXANDER S. SZALAY

Twenty years ago, work commenced on the Sloan Digital Sky Survey. The project aimed to collect a statistically complete dataset over a large fraction of the sky and turn it into an open data resource for the world's astronomy community. There were few examples to learn from, and those of us who worked on it had to invent much of the system ourselves. The project has made fundamental changes to astronomy, and we are now faced with the problem of ensuring that the data will be preserved and kept in active use for another 20 years. In redesigning this very large, open archive of data, we made a system that is able to serve a much broader set of communities. In this article, I discuss what we have learned by rebuilding a massive dataset that is available to an increasingly sophisticated set of users, and how we have been challenged and motivated to incorporate more of the patterns of data analytics required by contemporary science.

Keywords: astronomy; open data; sky surveys; databases; collaborative science

The unprecedented amount of observational, experimental, and simulation data is transforming the nature of scientific research. As more and more datasets are becoming public, those of us who are interested in data ubiquity need to find the right ways not only to make the data public, but accessible and usable. Yet our techniques have not kept up with this evolution. Traditionally scientists were moving the data to their computers to perform the analyses. With increasingly large amounts of data this is becoming difficult; as a result, scientists are learning how to "move the analysis to the data," that is, executing the analysis code on computers co-located with the

Alexander S. Szalay is the Bloomberg Distinguished Professor at the Johns Hopkins University. He is a cosmologist, also working on open scientific data. He built the archive of the Sloan Digital Sky Survey. He is a fellow of the American Academy of Arts and Sciences.

Correspondence: szalay@jhu.edu

DOI: 10.1177/0002716217745816

data repositories. In a digital world, we have to rethink not only the "data lifecycle" as most datasets are accessible via services, we also need to consider the "service lifecycle."

The data from Sloan Digital Sky Survey (SDSS) is one of the first examples of a large open scientific dataset. The data have been in the public domain for more than a decade (Szalay et al. 2000). It is fair to say that the project and its archive have changed astronomy forever, showing that a whole community of scientists are willing to change their approach to data and analytics, if the data are of high quality and presented in an intuitive fashion. The continuous evolution and curation of the system over the years has been an intense effort, and has given us a unique perspective on the challenges involved in operating open archival systems.

The SDSS is one of the first major open eScience archives. Tracking its evolution has the potential to help the whole science community to understand the long-term curation of such data, and see what common lessons emerged for other disciplines facing similar challenges. These new archives not only serve flat files, like a spreadsheet or a text document and simple digital objects like images and videos, but they also present complex services. The toolkits change, the service standards evolve, and even though some services may have been cutting edge 10 years ago, today they may be dated. To support the increasingly sophisticated client-side environments, the services need active curation at regular intervals.

Scientists in many disciplines would like to compare the results of their experiments to data emerging from computer simulations based on "first principles." Starting from a simple set of initial conditions, we apply the laws of nature to move the system forward to a state comparable to our observations and experiments. This tells us whether we have used the correct laws and approximations, and about the correctness of the initial assumptions. This requires not only sophisticated simulations and models, but that the results of the simulations also be made available publicly, through an easy-to-use portal. Turning simulations into open numerical laboratories where anyone can perform their own experiments and integrating and comparing experiments to simulations are nontrivial challenges in data management. Not every dataset from simulations has the same lifecycle. Some results are transient and need to be stored for a short while to analyze, while others become community references, with a useful lifetime of a decade or more.

In many areas, like environmental science, another challenge is the enormous complexity of the datasets involved. Various physical scales interact in a complex fashion; we have physical, chemical, biological factors all contributing to the observed phenomena. For example, the processes in the soil are affected by large-scale weather patterns, rainfall, on scales of many kilometers; but the local accumulation of water depends on local geographic features on meter scales, the soil properties are affected by the chemical composition and grain size on centimeter scales, and the decomposition is dominated by the microbes on the scales of a few microns. Much of these data reside in small files, such as the spreadsheets and tables collected in laboratories, in contrast to the large data collections like SDSS. These form the "long tail" of scientific data. Often, scientists would

like to cross-correlate the data in these small objects with each other as well as with the large online databases.

The progression through these challenges forms the story told in this article. We feel strongly that the framework developed for astronomy, and used for over a decade by the SDSS, captures the way scientists should approach scientific data, and our tools form a set of generic building blocks out of which many new applications can be built. In the sections below I describe these components, how they fit together, and how they can be generalized to solve a variety of problems.

Our data and services span a wide range in terms of their age: the SDSS data are quite far along in their lifecycle; after 20 years they face a set of different curation issues than services that have been in operation for 5 years (turbulence, cosmological simulations, sensors), and different from newly built applications. We discuss the history of our ongoing efforts, and describe our goals, and the objectives and methods applied to the problems.

SDSS is the first among the large scale eScience projects where the instruments are now approaching the end of their life, but the data will still be used, possibly for decades to come. We find ourselves at a place where we have to invent the best solution that serves the long-term needs of our user base.

The Origins

The Sloan Digital Sky Survey

The SDSS was one of the first large-scale digital surveys of the sky. The goal was to perform a high-resolution, five-color imaging of the northern sky, and based upon our own images, collect spectra of the brightest one million galaxies and a few hundred thousand stars and quasars. All data were to be open and public, after a six-month proprietary period.

The project was started in 1992, and expected to end in 2000. The total budget was to be $25 million, about $10 million on the telescope, $10 million on the instruments, $3 million on the mountain operations, and $2 million on the software.

The telescope started operating in 2001, and we finally completed the survey as originally proposed by 2008. The final cost was more than $100 million, with close to a third of this spent on software development, data processing, and data management. The original projections for the data in the mid-1990s were that we would collect about 10 terabytes (TB) of raw imaging data, process them as they arrive, with maybe an additional full reprocessing toward the end. We projected the total volume of the database to be about 0.5TB, which was quite a big number in 1992. The final tally of all the low-level data that need to be preserved for long-term use is about 150TB, the current size of the main database is 15TB, with an additional 20TB of supplementary databases, such as additional time-domain data. This was all made possible with the eventual delays in the survey, where Moore's Law (the power of computers doubles every 18 months) helped us to

reprocess the data much easier as we understood various systematic errors better, and Kryder's Law (the capacity of disks is doubling every 13 months) enabled us to store much more of the data as they were collected and processed. One of the lessons learned for future surveys was the fact that in projects like the SDSS the *capital investment is in the software and the computational hardware became disposable*.

The SDSS was amazingly successful. There are now more than seven thousand refereed papers published, with well over 350,000 citations. Many more papers were from outside the collaboration than by the survey participants, and the published papers have exceeded our imagination.

Initially, there was a lot of distrust in the astronomy community whether SDSS would truly release their data as promised. It took several years to convince the astronomers that we stood by our promises—we have never missed a data release. In the end, most of the proprietary periods were close to zero, and those of us who were creating the dataset were almost always getting the data at the same time as the public. In the beginning, people did not believe that we were able to process the data well enough in an automated environment. Much of the astronomy software at the time still required that an astronomer directly issue interactive commands. This was unacceptable for a uniform processing of a large fraction of the sky, and represented one of the biggest unforeseen challenges for the project. Creating such an automated pipeline that required essentially no human interaction to process millions of images required much more code to be written than previously envisaged.

There was very little precedent that we could rely upon, and we had to make things up as we went along, rather quickly, as the data went online and usage grew quickly. We had to figure out how to deal with reproducibility. New data were added every few nights. We decided to adopt a model where the public data releases happened once a year, and once released, they never changed. This meant that a query submitted to a particular data release, say DR3, will always return the same result. While DR4 and later contain all of DR3, and more, each of the data releases was treated as a separate edition of a book. When a new "edition" was released, we did not take the old ones off the shelf. This enabled students, who started their thesis with a particular data release, to remain consistent and papers published on an earlier data release can be fully reproduced by rerunning the queries on the original version of the database. Today we are at DR15 and counting. All data releases are still up and available at Johns Hopkins University. We have completed the imaging survey of the available sky, and turned off the main imaging camera and now are only taking spectra.

The data releases presented an interesting challenge. Data come in at an approximately linear rate, assuming no significant changes are made to the detectors. This means that from year one to year two the data are doubling, but after that the relative change is much smaller. The total amount of data that we have to store is approximately quadratic: $1 + 2 + \ldots N = N(N - 1)/2$. Of course, we always store several copies of the most current data: right now, we are serving six different instances of DR13 and later. We can relax this somewhat for the older, less used versions. As the price of storage is constantly dropping, with larger and

larger disks, we found that the second year was the hardest to accommodate from the perspective of hardware expenditures. The evolution of storage technologies has far outpaced the rate of 20 years' linear-rate data collection.

Jim Gray and the SDSS archive

Alex Szalay and Ani Thakar, in collaboration with Jim Gray (Microsoft), have spent much of the last two decades working on the archive for the SDSS. The archive was originally built on top of an emerging database technology, but after a few years it became clear that the users would want to have a very flexible query environment with a lot of ad hoc queries, which this system could not support well enough. As a result, we have started to develop our own (very limited) additions to the query capabilities of the system.

It was around this time, when we met Jim Gray of Microsoft. Jim liked to say that the "best collaborators are the desperate ones," as they are ready to change the way they approach a problem. We were desperate at that point. After a few meetings, Jim advocated for a more traditional relational database, consisting of tables of columns and rows—a much more mature technology. He made the point that a few programmers in an academic environment cannot successfully compete with the thousands of developers at Microsoft, Oracle, and IBM, and we should spend our efforts on creating the additional "business logic" related to astronomy, and use an off-the-shelf commercial platform with a robust engine. These are all based on SQL, the Structured Query Language: the standard, portable way to query databases. This advice set us on the trajectory that we have followed ever since.

The project has revolutionized not just professional astronomy but also the public's access to it. Although a substantial portion of the astronomy community is using the SDSS archive on a daily basis, the archive has also attracted a wide range of users from the public (Singh et al. 2006): a scan of the logs showed more than 4M distinct IP addresses accessing the site. The total number of professional astronomers worldwide is only about fifteen thousand. Furthermore, the collaborative CasJobs interface has more than eight thousand registered users—almost half of the professional astronomy community.

SDSS (2000–2005) and its successors SDSS-II (2005–2008) and SDSS-III (2008–2014) have an extraordinary legacy of mapping structure across a vast range of scales, from asteroids in our own solar system to quasars more than 10 billion light years away. These surveys have produced data that have supported 7,000 papers with more than 350,000 citations. The SDSS has several times been named the highest impact project, facility, or mission in the field of astronomy, as judged by number of citations of associated refereed journal articles (Banks 2009; Madrid and Macchetto 2009). The SDSS was the source of the most highly cited astronomy article in the years 2000, 2002, 2005, and 2008 (Frogel 2010). Within the collaboration there have been more than 120 SDSS-based PhD theses, and outside the collaboration there have been many more. Its publicly available, user-friendly tools have fueled a large number of undergraduate and even high-school projects and research papers.

In 2007 we played a key role in launching the Galaxy Zoo citizen science project (Lintott et al. 2008) in which online volunteers—members of the public, most of whom had no prior experience with scientific research—were asked to visually classify SDSS images of nearly a million galaxies. Today, Galaxy Zoo has more than 200,000 volunteers, who have collectively classified each of the million galaxies between 9 and 50,000 times. Galaxy Zoo has been featured by many of the world's best-known and respected news organizations (BBC, the *New York Times, Nature*, etc.), showing how active scientific research can attract a large and involved nonexpert population. But Galaxy Zoo users have contributed more than just raw image classifications. One of the most unexpected and successful parts of the project was the way in which citizen scientists used SkyServer tools to learn more about the galaxies they were asked to classify. In two cases, these efforts by citizen scientists led to published original research in astronomy journals (Lintott et al. 2009; Cardamone et al. 2009).

The 2.5-meter Sloan telescope in Apache Point, New Mexico, remains the most powerful wide-field spectroscopic survey facility in the world today. To capitalize on this resource, a collaboration of 186 astronomers and physicists from sixty-five institutions have organized the SDSS-IV program, conducting a broad survey of our Milky Way Galaxy, the population of nearby galaxies in the local universe, and the large-scale structure of the universe as a whole. SDSS-IV will operate from July 2014 to July 2020. It will marshal imaging data from multiple telescopes and wavelength regimes to identify targets for follow-up spectroscopy.

Evolution during the Early Years

The SkyServer usage log database

One of the most useful byproducts of the SDSS data has been the usage logs that we have kept since the very beginning of the project: every web hit and every single query have been logged since the archive was opened. The log database today is over 3TB, and contains rich historical information about how astronomers learned to access a virtual telescope (Singh et al. 2006). This has resulted in an amazingly rich and useful resource not only for SDSS scientists and project managers, but since the dataset is available to anyone, many other projects and researchers have found it extremely valuable. Next generation large astronomy surveys like Pan-STARRS (Heasley et al. 2007) and LSST (Becla et al. 2006) have used this data to plan their data management infrastructure and services, and several other groups in astronomy and computer science have downloaded the entire dataset for analysis. The SDSS log data was the subject of a PhD dissertation at Drexel University in Human-Computer Interaction research (Zhang 2011). We receive on average one or two requests per month to download the SDSS log data, especially the SQL query logs since this is perhaps the only such large dataset of SQL usage in existence. The SDSS has several mirrors over the world (UK, Brazil, India, China, Hungary). Their logs are harvested every night

and aggregated into our main log database. On the main SDSS SkyServer web page there is a link to some cumulative counts from the log DB. The most current values are 2.4 billion web hits, and 364 million free-form SQL queries.

Parallel loader environment

The raw data are transformed into a common loadable format by a set of specific plug-ins. The data come in blocks, typically a few tens of gigabytes (GBs) at a time. Each block is transformed into a set of files in a single directory tree, with checksum files stored in each directory. For larger datasets this is a brutal data-parallel operation. The results of the data transformation process are picked up by the parallel loader (Szalay, Thakar, and Gray 2008). The loader scales to an arbitrary number of machines. It performs a two-phase load. First, for each block, we create an empty database with the same schema as the main system, and load the whole block. During the load process, broken into tasks, steps, and phases, we generate a detailed log at each granularity. The state of all jobs can be tracked visually using the load monitor interface.

Some of the tasks in the workflow described by a DAG (directed acyclic graph) perform a very detailed integrity check and data scrubbing, looking for out-of-band data. The data rows in each block get tagged by a load-ID unique to the block, so that combining this with the loader logs allows us to track each row's provenance. The two-phase load has proven to be invaluable, as data errors were caught well before the bad data could have contaminated the main database. It turns out that the load performance of a typical database server, running on a good file system, is not I/O- but rather CPU-limited, due to the various page formatting and checksum calculations. We found that on a high-end SQL server machine, using an array of SSDs and thirty-two cores, we were able to achieve load speeds in excess of 1GB per second using thread parallelism. Once data are in a DB page format, copying the DB files to other machines is only limited by the hardware performance.

The web interface

Early on we decided to move away from the solely form-based interfaces to the archive, where users are only allowed to enter certain parameters on a particular predefined search pattern through a web page, but they are not allowed to change the pattern itself. We decided to have a highly visual interactive front end, based on the available browsers at the time. This era is still remembered as "browser hell," as the existing web browsers had rather incompatible functionalities and commands. Internet Explorer and Netscape were still not capable of stylesheets, their javascript implementations were rather different. In the end, we built the website using our own abstract API for rendering various items, which were mapped onto their native implementations when a web page was loaded. This approach saved us a lot of headaches. Some of the functions are still there, but we are in the process of gradually replacing them with HTML5 canvas,

and other more modern components. Based upon Jim Gray's Terraserver experience, and the emerging MapQuest, we created a clickable map of the sky, with image mosaics built server side from precomputed color images.

Free-form SQL queries

After about a year we visited the National Center for Biological Information (NCBI) for a few days. During this visit, David Lipman, the NCBI director showed us an article arguing against form-based interfaces to biological databases. The author felt that these interfaces, designed by a programmer and not by a scientist at the cutting edge, restrict the patterns of how data can be explored, thus limiting the scope of possible science. He suggested that there should be a back-door, enabling "anything and everything goes" type of creative query.

We decided to open the database for free-form SQL queries. Many people cautioned us against this, arguing that no astronomer would want to write SQL queries and that we would be constantly hit with denial of service attacks. We did it anyway and much to our amazement we found that neither of those predictions came true: astronomers embraced SQL remarkably quickly, and there were no major abuses of the interface.

To help astronomers to learn SQL, we posted the twenty queries that came out of early discussions. We first displayed the original question or problem definition written in plain English, then showed the SQL implementation that executed the query. Finally, in about a page or so we explained why the query was written the way it was shown. This enabled the astronomers to look for a query that was close enough to what they wanted to do, first do a cut and paste, run it, and start modifying it step-by-step, until they arrived at their results. Over the years we have added another fifteen query patterns to the pool. We have also built a step-by-step tutorial to teach the basics of the SQL.

Maturity and Production

The CasJobs/MyDB collaborative environment

As traffic on the SDSS archive grew, many users were running repeated queries and extracting a few million rows of data. The DB server delivered such datasets in 10 seconds, but it took several minutes to transmit the data through the slow wide-area networks. We realized that if users had their own databases at the server, then the query outputs could go through a high-speed connection, directly into their local databases. This improved system throughput by a factor of ten. Furthermore, we have built an asynchronous (batch) mode that enabled queries to be queued for execution and results to be retrieved later at will.

The CasJobs/MyDB batch query workbench environment was born as a result of combining these "take the analysis to the data" and asynchronous query execution concepts. The name "CasJobs" comes from "CAS (Catalog Archive Server)"

and (batch) "jobs." CasJobs builds a flexible shell on top of the large SDSS database. Users are able to conduct sophisticated database operations in their own space: they can create new tables, perform joins with the main DB, write their own functions, upload their own tables, and extract their value-added datasets to their home environment. The system was an overnight success.

For redundancy, we had three identical servers containing the active databases. By studying the usage patterns, we realized that the query length distribution was well represented by a power law. Hence, we split the traffic into multiple queues served by different servers, each handling the same aggregate workload (O'Mullane et al. 2004). Each query can be submitted to a "fast," "medium," or "long" queue, returning the result into a MyDB table. The user can then process the derived result further, run a multistep workflow, or extract the data. Everything that a user does is logged. This set of user-controlled databases form a very flexible tier on top of the rigid schema of the archive. This resolves the long-standing tension between stability and integrity of the core data and the flexibility for user creativity.

As users became familiar with the system, there were requests for data sharing. As a result, we added the ability to create groups and to make individual tables accessible to certain groups. This led to a natural self-organization, as groups working on a collaborative research project used this environment to explore and build their final, value-added data for eventual publication. GalaxyZoo, which classified over a million SDSS galaxies through a user community of 300,000, used CasJobs to make the final results world-visible, and CasJobs also became a de facto platform for publishing data. We added the capability for users to upload their own datasets and import them into their MyDBs for correlation with the SDSS.

SQL extensions

Over the years, we have developed a design pattern to add domain specific extensions to the SQL server, using CLI integration. Our code for spatial indexing was used in the "shrink-wrap" production version of SQL Server 2005 (Fekete, Szalay, and Gray 2006; Budavári, Szalay, and Fekete 2010). The idea is to take a class library written in one of the .NET languages (C++, Java,C#), store a binary instance of the class as a binary datatype, and expose the object methods as user-defined functions (UDFs). The SQL server makes this very convenient, since unlike many other database platforms like MySQL, it allows for table-valued UDFs. One can then pass the binary object as a parameter to the function and execute the method, or access the property.

We have 236 UDFs supporting detailed astronomy knowledge, like conversion of cosmological coordinates in a curved space to angles and radial distances. Also, we have built an astronomy-specific spatial index, representing spherical polygons with extreme accuracy over the whole sky, with a relational algebra over the regions, and fast indexing capabilities to find several million points per spherical region in a second.

For large numerical simulations much of the data are in multidimensional floating point arrays. We have built such a User Defined Type for SQL Server, which is used for all our simulation databases (Dobos et al. 2011). We will develop a generic module that repartitions the data in a large array into smaller blocks organized along a space-filling curve, adds the custom metadata header, and writes these out in native binary format for optimal SQL server load performance.

Schema and metadata framework

The schema for the database is contained in a set of DDL files. These files are quite complex; they not only contain the code to generate the database and the associated stored procedures and user defined functions, but in the comment fields of the scripts they contain rich metadata describing the schema elements, including physical units, enumerations, indexes, primary keys, and short and long descriptions. A parser can extract this information at different granularities (object, column) and create a set of metadata tables that can be automatically loaded into the database. This ensures that all the schema and related metadata are handled together in an automated fashion, similar to the approach originally employed by Donald Knuth, when he created TeX. The database will then contain all the up-to-date metadata and these can be queried and displayed using simple functions and dynamic web services. This tool is quite robust and mature and has been in use for more than 14 years.

Branching Out to Other Disciplines

The SkyServer genealogy

The template for the SDSS archive is now being used within astronomy by several projects and institutions beyond JHU (STScI, Fermilab, Caltech, Edinburgh, Hawaii, Portsmouth, and Budapest). The technologies and concepts used for the SDSS archive have also been used beyond astronomy. Using the same template, we have built databases for a growing number of other disciplines. Such databases include those for turbulence (Li et al. 2008), radiation oncology (McNutt et al. 2008), environmental sensing and carbon cycle monitoring (Szlavecz et al. 2006), and, most recently, a prototype for high-throughput genomics (Wilton et al. 2015). The databases built for cosmological simulations are revolutionizing how astronomers interact with the largest simulations.

Open numerical laboratories

Worldwide, there is an ongoing effort to build an exascale computer. However, fewer and fewer codes will scale to millions of cores, and as a result, fewer people will use these ever larger machines. There will be an increasing gap between the wide science community and the top users. It will be increasingly important to

create science products that can be used by a much wider pool of users, otherwise community support will dwindle. There is already an increasing demand from the broader science community to access the largest numerical simulations. While only our largest supercomputers are capable of creating such simulations, their analysis, especially if the data will be publicly accessible, requires a different type of architecture.

To date, the usual way of analyzing somebody else's simulation is to download the data. With PB scale datasets, this will not work. We are experimenting with a new, immersive metaphor for interacting with large simulations by using a large number of virtual sensors that can be placed in a simulation, anywhere at any timestep. They can also be set to send a data stream in real physical quantities. Imagine how scientists could launch mini accelerometers into simulated tornadoes, emulating the movie *Twister*. We have successfully implemented this metaphor for our turbulence data, and are now porting it to the cosmology simulations.

In this approach, one can create a so-called immersive environment, in which the users can insert virtual sensors into the simulation data. These sensors can then feed data back to the user. They can provide a one-time measurement, they can be pinned to a physical (Eulerian) location, or they can "go with the flow" as comoving Lagrangian particles. By placing the sensors in different geometric configurations, users can accommodate a wide variety of spatial and temporal access patterns. The sensors can feed back data on multiple channels, measuring different fields in the simulation.

This pattern also enables the users to run time backward, something that is impossible in a direct simulation involving dissipation. Imagine that the snapshots are saved frequently enough that one can interpolate particle velocities smoothly. Sensors can back-track their original trajectory and one can see where they came from, all the way back to the initial conditions. This simple interface can provide a very flexible, yet powerful, way to do science with large datasets from anywhere in the world. The availability of such a 4D dataset "at your fingertips" and the ability to make "casual" queries from anywhere are beginning to change how we think about the data. Researchers can come back to the same place in space and time and be sure to encounter the same values.

The Twister metaphor mentioned above was implemented in the Turbulence DB eight years ago. The Turbulence DB is the first space-time database for turbulent flows, containing the output of large simulations, publicly available to the research community (Perlman et al. 2007). The 27TB database contains the entire time-history of a 1024^3 mesh point pseudo-spectral Direct Numerical Simulation of forced Navier-Stokes equations representing isotropic turbulence. One thousand and twenty-four time-steps are stored, covering a full "large-eddy" turnover time of model evolution. We have built a prototype that serves requests over the web for velocities, pressure, various space derivatives of velocity and pressure, and interpolation functions. The data and their interface are used by the turbulence research community and have led to about 100 publications to date. To date we have delivered more than 36 trillion data points to the user community. In a recent paper on MHD, trajectories were computed by moving the

particles backward in time, something that is impossible to do in an in situ computation and only enabled by interpolation over the database (Eyink et al. 2013).

A similar transformation is happening in cosmology. The SDSS SkyServer framework was reused for the Millennium simulation database (Lemson and the Virgo Consortium 2006). The database has been in use for more than 10 years, has hundreds of regular users, and has been used in nearly 700 publications. The database contains value added data from a simulation originally only containing 10B dark matter particles. A semi-analytical recipe was used to create mock galaxies in the simulations, and their hierarchical mergers were tracked in the database. The merger history was used to assign a plausible star formation rate to each galaxy, which in turn can be used to derive observable physical properties. The database contains several such semi-analytic scenarios and has been expanded with data from three other simulations, one of which contains 300 billion particles.

Environmental science

Environmental data are complex; combine biological, physical, and geological measurements; and are heterogeneous in space and time. The data are fragmented, and as various scientists focus on specific variables and store data in isolated file systems, integration becomes a significant challenge. A great deal of effort has been spent to make environmental data more accessible. A common feature of these networks is that they have largely focused on data accessibility through metadata catalogs where investigators can search data by keyword, project name, investigator name, and so on.

Our pilot system focused on integrating data on various spatial and temporal scales to answer science questions related to the soil ecosystem. LifeUnderYourFeet (Szlavecz et al. 2006) has been continuously collecting soil moisture and temperature data since 2008, and soil respiration data since 2010. We used the SciServer framework to integrate data at national and local spatial scales and to correlate soil measurements in space and time for various climatic, atmospheric, meteorological, and anthropogenic conditions and scales.

Toward a Sustainable Solution: The SciServer

Consolidating the evolution

Over the first 12 years of the SDSS archive we have incrementally evolved the system, avoiding major architectural changes. The SDSS data with all the additional science projects have been created at a cost of well over $100 million. They are widely used by a diverse community, and are generating new papers and supporting original research every day.

But the services are showing signs of aging; while the data are still very much alive, they will still be used 15 years from now. To prepare for the future, we need

to consolidate and reengineer the services, to make them more sustainable and inexpensive to operate. To do this, we have endeavored to convert the SkyServer to the SciServer, a generic, modular set of building blocks that can be connected in several ways.

New building blocks

FileDB. Relational databases have shown their value to the scientific community. The SDSS Database (Thakar et al. 2008) was a forerunner, showing how the community was willing to take the step of learning SQL to access a database. However, data volumes are reaching the limits of what can be managed within relational databases with reasonable effort. For example, it takes a week to load a typical Turbulence database. To avoid this bottleneck, we built a system that allows raw data from the database to be linked, using indexes, without ingesting them into the database. We wrote custom functions that can access the file system, but can be called from ordinary SQL. These functions are exposed as table-valued, user-defined functions and are accessible through standard SQL queries. Their performance is as good as native DB calls.

ScratchDB. We have enabled the CasJobs system to have many other contexts, not just the SDSS data versions (right now we have all the previous data releases from DR1 through DR9) but also other astronomical collections. We will also bring the simulations and environmental datasets into the federation. Uploaded and derived data (and the related metadata) will automatically show up in the user's MyDB. For large scale intermediate data, the small user space is not quite enough. For example, a custom cross-match of large astronomical catalogs, like SDSS and GALEX, might require several 100 GBs if not TBs of disk space. This cannot be done today. We resolve this problem with a new MyScratch layer between the static contexts and the MyDBs, with tens of TB of storage, both in flat files and as large databases.

Advanced scripting. Our users, both in SDSS and in the numerical laboratories have become quite artful in using database queries. They use SQL tools not as a hammer, but rather as a violin, and they generate "nice music." But with the emergence of Python, sophisticated machine learning algorithms, libraries, and packages have become available, and the users are now keen to use these with the same ease of interactivity as SQL. A typical use case would start with an SQL query returning tens of thousands of objects with a particular spectral property. However, the user would then like to go back to the raw data (spectra in this case) and run her own tools and algorithm written in Python.

To facilitate this, we built two add-ons: one is SciServer Compute, a set of servers providing about 100 virtual machines, always available, that can be used to start Jupyter/iPython notebooks, within Docker containers. These are preconfigured with the database interface tools, and users can run their SQL queries out of Python. Furthermore, all the raw data files of SDSS (about 150TB) are

wrapped into a data container, so access is trivial. The Jupyter environment also enables Matlab and R, which are relevant for our engineering and Biostats/genomics users. Several of our interactive numerical laboratories (Turbulence, Ocean Circulation, N-body) are now using both Python and Matlab bindings.

SDSS Futures

Consolidation of the SDSS versions

We aim to integrate the SDSS-IV results with the legacy data from earlier stages: SDSS-I, SDSS-II, and SDSS-III, including a large (14,555 square degree) imaging survey of the sky with follow-up optical and near-infrared spectroscopy. Currently, because the SDSS-III project proceeded under a different organizational structure than SDSS-II, the SDSS products have branched into two distribution sites. For SDSS-IV, we plan to reintegrate this distribution under a single archive that includes all the legacy data and documentation, as well as the new data, integrated under the reengineered and enhanced version of SciServer. The proven flexibility and extensibility of the Sky/SciServer framework makes it possible to integrate these new data in a coherent and scientifically powerful fashion. The total data volume of the survey, combined with the legacy data, is projected to be around 400TB, with the final reduced catalogs around 15 to 20TB. In addition, these final reduced catalogs will consist of several different flavors of data—optical and near-infrared spectra and several different types of imaging data (optical, near- and mid-infrared). Finally, the combination of imaging and spectroscopic coverage maps will form a complex pattern on the sky that will need to be described quantitatively for science, and that the spatial tools of the SkyServer have been designed to track.

The data lifecycle

We often talk about the data lifecycle, and its phases. As the SDSS project is probably nearing its data acquisition end, we have to think carefully about the long-term sustainability of the data archive, and how it will be curated and preserved. Given that its usage shows no signs of decreasing, we need to consider that the data will support good science for another 20 years. How can we support such a long lifecycle, where does the support come from, and where will the data reside? It is time to start thinking about what happens to the data after the sunset of the observations.

We can see three distinct phases. In Phase 1 observations are still happening. As long as the SDSS telescope is still taking data, the archive is part of an active data collection effort. Phase 2 starts once the telescope is shut down. The archive needs to be kept alive, but the data do not grow any longer. Over a five-year period during this phase we need to consolidate the services as much as possible. This must be done by the team currently operating the archive. During the following five years of Phase 3, the archive must be handed off to an organization

that can operate on a good economy of scale, and whose sustained existence is guaranteed, independent of the individual datasets. One of the possibilities we are considering is to identify a set of university libraries, which are willing to undertake this task of maintaining the archive and operate a help desk. This phase will continue as long as there is continued use of the archive and one can justify its existence based upon scientific value generated.

The service lifecycle

However, during the 20 years that we have been working on the SDSS archive and now the SciServer, we have learned about the service lifecycle as well. The SDSS archive and now the SciServer are much more than just a simple file-store. The data are served through a set of sophisticated, smart services, which offer a lot of server-side functionality. In 2001 we built the first web services deployed in a science setting, but now many of these APIs and interfaces are obsolete. Computing has undergone several major paradigm shifts. Over the last 20 years we went from a lot of different technologies, with their own special acronyms: CORBA to The Computational Grid, to Web Services, Grid Services, then the Cloud, and most recently to Data Lakes. No matter what, this dynamic evolution is going to continue, and it is difficult to predict what the world of distributed computing will look like even five years from now.

There is also natural aging. Technology has improved significantly since we built our first services; the first web services in science were built for SDSS by us. While several improvements have been implemented over time, it is important to rethink the methodologies in the context of the new, evolving Internet. Smarter client-side web interfaces are possible today using HTML5 and JavaScript, which are standard and quickly became widely accepted. These modifications will enable our new infrastructure to perform some of the processing steps in the browsers rather than overloading the servers. Smarter clients will work efficiently with new services; by now, REST has replaced SOAP almost everywhere. Asynchronous messaging protocols will make the infrastructure more robust against the glitches in communications. Behind the web server we will build a universal application layer that uses proper scheduling mechanisms to handle the large volume of complex user jobs. Load balancing will be realized on all levels by partitioning and parallel execution of the tasks over a cluster of database servers.

From queries to file extractions, everything will be prioritized and executed in the most efficient way by schedulers that keep track of data locality and use the closest copies in the distributed database system. The next generation execution environment will be based on workflows, whose state can be persisted in a database. Thus long-running and expensive scientific analyses can be suspended and resumed, making the framework more resilient and the system management much easier.

All that we can do today is to prepare for these changes to come, and reorganize the underlying services and APIs in such a way, that they are maximally modular and independent, so that future upgrades and improvements will be as

painless as possible. Building the database schema to be maximally portable enabled us to move from one database system to another, until we settled on SQL Server. In the SciServer we are extending this philosophy, and we have further modularized the whole environment, and incorporated design patterns going beyond astronomy. That it was very easy to bring new science use cases, even related to social science, into the SciServer validated our approach beyond our initial hopes.

Community response

In just a few years these datasets have earned the trust of the astronomy community and have been heavily used. Starting with the Turbulence project, we introduced the notion of interactive, database-centric tools into other science domains. The initial reaction from the turbulence community was rather skeptical: they felt that they could analyze their own data more effectively than through our database approach. However, others in the research community did not feel this way.

Many researchers started to access the data in our system and do their research in the open numerical laboratory. For instance, experimentalists could place tracer particles as measurement devices inside the numerical space-time data in our numerical laboratory and calibrate their measurement techniques. Mathematicians could find seeds of possible singularities in the partial differential equations. These scientists represent a cross-section of the research community that had real difficulties accessing large datasets from simulations prior to the JHTDB. The availability of our open numerical laboratory has led to many results and papers by researchers all over the world, having been used for over one hundred published papers on turbulence, for example. In 2015 the number of points has exceeded 12 trillion, and recently, it had reached 56 trillion.

A similar transformation is happening in cosmology. The SDSS SkyServer framework was reused for the millennium simulation database (Lemson and the Virgo Consortium 2006). The database has been in use for more than eight years, and has hundreds of regular users, and has been used in nearly seven hundred publications. The set-oriented SQL makes it remarkably easy to formulate very complex aggregate queries over the temporal history of various subsets of galaxies and create samples that can be compared directly to observations.

Conclusion: The Cost and Price of Data

In every new community with which we have engaged, it takes about three to five years to overcome the initial skepticism, and for us to demonstrate that our interactive approach to large-scale problems is more scalable than the traditional ones. We have to earn the trust of the community—by giving them open access

to high quality data and easy to use tools that mesh well with how they analyze their data.

It is also clear that none of the domain communities understand the subtle differences between the value of data, and the cost of data, and the cost of archiving. The value of data is relatively easy to grasp, we make new discoveries based upon these datasets, write new papers, share them, combine them with other datasets, and they provide a solid foundation for reproducible results.

It is much harder to define the price of data. On one hand, one can argue that the price of the data is the cost to build and run the instruments. Many of today's large data collections in this sense have cost hundreds of millions of dollars (SDSS), if not billions (LHC). On the other hand, one can argue that a typical NSF grant of $100,000/year is considered high quality if it produces two papers in a good, refereed journal annually. By this token, the value of a paper is about $50,000. Of course, not all science support goes into the individual grants, at least an equal amount goes into various national facilities, both physical and computational. Let us double this number, and estimate the value of a good scientific paper to be about $100,000. By this measure, the SDSS data have, to date, resulted in more than 7,000 refereed publications, and this has a "monetized value" of $700 million. At the same time, the total cost of all the SDSS projects combined has been less than $200 million, making it very cost-efficient.

We also need to consider the cost of archiving. On one hand, we can calculate the physical costs, power, disk drives, curation personnel, servers, and so on. In astronomy, the typical annual operating cost of a telescope is around 5 to 10 percent of the capital investment. Everyone accepts this. At the same time, we are still shocked if the cost of maintaining an archival dataset is a few hundred thousand dollars, often a small fraction of 1 percent of the capital cost of acquiring it. Yet these archival datasets will generate a disproportionally high value in terms of new publications, for several decades to come.

These large, open datasets, analyzed by a much broader range of scientists than ever before, using all the tools of the computer age, are creating a new way to do science. We cannot predict where they will lead, but it is already clear that these technologies have brought and will bring about dramatic changes in the way we do open science and make new discoveries.

References

Banks, Michael. March 2009. Impact of sky surveys. *Physics World*.

Becla, Jacek, Andrew Hanushevsky, Sergei Nikolaev, Ghaleb Abdulla, Alexander S. Szalay, Maria Nieto-Santisteban, Ani R. Thakar, and Jim Gray. 2006. Designing a multi-petabyte database for LSST. *Proceedings of the SPIE* 6270:62700R.

Budavári, Tamas, Alexander S. Szalay, and George Fekete. 2010. Searchable sky coverage of astronomical observations: Footprints and exposures. *Publications of the Astronomical Society of the Pacific* 122:1375–88.

Cardamone, Caroline N., Kevin Schawinski, Marc Sarzi, Steven P. Bamford, Nicola Bennert, C. Megan Urry, Christopher J. Lintott, William C. Keel, John Parejko, Robert C. Nichol, et al. 2009. Galaxy Zoo

green peas: Discovery of a class of compact extremely star-forming galaxies. *Monthly Notices of the Royal Astronomical Society* 399:1191–1205.

Dobos, Laszlo, Istvan Csabai, Milos Milovanovic, Tamas Budavari, Alexander. S. Szalay, Marko Tintor, Jose Blakeley, Andrija Jovanovic, and Dragan Tomic. 2011. Array requirements for scientific applications and an implementation for Microsoft SQL Server. Paper presented at EDBT/ICDT Workshop on Array Databases, Uppsala, Sweden.

Eyink, Gregory, Ethan Vishniac, Cristian Lalescu, Hussein Aluie, Kalin Kanov, Kai Bürger, Randal Burns, Charles Meneveau, and Alexander S. Szalay. 2013. Flux-freezing breakdown in high-conductivity magnetohydrodynamic turbulence. *Nature* 497:466–69.

Fekete, George, Alexander S. Szalay, and Jim Gray. 2006. Using table valued functions in SQL Server 2005. Paper presented at the MSDN Development Forum.

Frogel, Jay A. 2010. Astronomy's greatest hits: The 100 most cited papers in each year of the first decade of the 21st century (2000–2009). *Publications of the Astronomical Society of the Pacific* 122 (896): 1214–35.

Heasley James N., Maria Nieto-Santisteban, Alexander S. Szalay, and A. Thakar. 2007. The Pan-STARRS object data manager database. *Bulletin of the American Astronomical Society* 38:124.

Lemson, Gerard, and the Virgo Consortium. 2006. Halo and galaxy formation histories from the millennium simulation: Public release of VO-oriented and SQL-queryable database for studying the evolution of galaxies in the ΛCDM cosmogony. Available from https://arxiv.org/abs/astro-ph/0608019.

Li, Y., Eric Perlman, Minping Wan, Yunke Yang, Charles Meneveau, Randal Burns, Shi-Yi Chen, Alexander S. Szalay, and Gregory Eyink. 2008. A public turbulence database cluster and applications to study Lagrangian evolution of velocity increments in turbulence. *Journal of Turbulence* 9 (31): 1–29.

Lintott, Christopher. J., Kevin Schawinski, William C. Keel, Hanny Van Arkel, Nicola Bennert, Ed Edmondson, Daniel Thomas, Daniel J. B. Smith, Peter D. Herbert, Matt J. Jarvis, Shanil Virani, Dan Andreescu, Steven P. Bamford, Kate Land, Phil Murray, Robert C. Nichol, M. Jordan Raddick, Anze Slozar, Alexander S. Szalay, and Jan Vandenberg. 2009. Galaxy Zoo: Hanny's Voorwerp, a quasar light echo? *Monthly Notices of the Royal Astronomical Society* 399:129–40.

Lintott, Christopher J., Kevin Schawinski, Anze Slosar, Kate Land, Steven P. Bamford, Daniel Thomas, M. Jordan Raddick, Robert C. Nichol, Alexander S. Szalay, Dan Andreescu, Phil Murray, and Jan Vandenberg. 2008. Galaxy Zoo: Morphologies derived from visual inspection of galaxies from the Sloan Digital Sky Survey. *Monthly Notices of the Royal Astronomical Society* 389:1179–89.

Madrid, Juan P., and F. Duccio Macchetto. 2009. High-impact astronomical observatories. *Bulletin of the American Astronomical Society* 41:913–14.

McNutt, Todd R., Thomas Nabhani, Alexander S. Szalay, Theodore Deweese, and John Wong. 2008. Oncospace: EScience technology and opportunities for oncology. *Medical Physics* 35:2900–2901.

O'Mullane, William, Jim Gray, Nolan Li, Tamas Budavari, Maria Nieto-Santisteban, and Alexander S. Szalay. 2004. Batch query system with interactive local storage for SDSS and the VO. In *Proceedings of the ADASS XIII, ASP Conference Series*,eds. F. Ochsenbein, Marc Allen, and Daniel Egret, 372–76.

Perlman, Eric, Randal Burns, Yi Li, and Charles Meneveau. 2007. Data exploration of turbulence simulations using a database cluster. In *Proceedings of the Supercomputing Conference* (SC'07). doi:10.1145/1362622.1362654.

Singh, Vic, Jim Gray, Ani R. Thakar, Alexander S. Szalay, Jordan Raddick, Bill Boroski, Svetlana Lebedeva and Brian Yanny. 2006. *SkyServer traffic report: The first five years*. Microsoft Technical Report, MSR-TR-2006-190. Redmond, WA: Microsoft.

Szalay, Alexander S., Peter Kunszt, Ani R. Thakar, Jim Gray, Donald Slutz, and Robert Brunner. 2000. Designing and mining multi-terabyte astronomy archives: The Sloan Digital Sky Survey. In *Proceedings of SIGMOD 2000 Conference*, 451–62.

Szalay Alexander, Ani R. Thakar, and Jim Gray. 2008. The sqlLoader data-loading pipeline. *Computing in Science and Engineering* 10:38–48.

Szlavecz, Katalin, Andreas Terzis, E. Razvan Musăloiu, Josh Cogan, Sam Small, Stuart Ozer, Randal Burns, Jim Gray, and Alexander S. Szalay. 2006. *Life under your feet: An end-to-end soil ecology sensor network, database, web server, and analysis service*. Microsoft Technical Report, MSR-TR-2006-90. Redmond, WA: Microsoft.

Thakar Ani R., Alexander S. Szalay, George Fekete, and Jim Gray. 2008. The catalog archive server database management system. *Computing in Science and Engineering* 10:30–37.

Wilton, Richard, Tamas Budavari, Ben Langmead, Sarah J. Wheelan, Steven L. Salzberg, and Alexander S. Szalay. 2015. Arioc: High-throughput read alignment with GPU-accelerated exploration of the seed-and-extend search space. *PeerJ* 3. doi:10.7717/peerj.808.

Zhang, Jian. 2011. Data use and access behavior in eScience: Exploring data practices in the new data-intensive science paradigm. PhD thesis, Drexel University, Philadelphia, PA.