Designing and Mining Multi-Terabyte Astronomy Archives: The Sloan Digital Sky Survey

Alexander S. Szalay
Peter Kunszt
Ani Thakar
Dept. of Physics and Astronomy, The Johns Hopkins University, Baltimore, MD 21218

Jim Gray
Don Slutz
Microsoft Research, San Francisco, CA 94105
Robert J. Brunner,
Dept. of Astronomy, California Institute of Technology, Pasadena, CA 91125

June 1999
Revised Feb 2000

Designing and Mining Multi-Terabyte Astronomy Archives: The Sloan Digital Sky Survey

Alexander S. Szalay, Szalay@jhu.edu, Peter Z. Kunszt, Kunszt@pha.jhu.edu, Ani Thakar, Thakar@pha.jhu.edu
Dept. of Physics and Astronomy, The Johns Hopkins University, Baltimore, MD 21218
Jim Gray, Gray@Microsoft.com, Don Slutz Dslutz@Microsoft.com
Microsoft Research, San Francisco, CA 94105
Robert J. Brunner, RB@astro.caltech.edu, California Institute of Technology, Pasadena, CA 91125

ABSTRACT

The next-generation astronomy digital archives will cover most of the sky at fine resolution in many wavelengths, from X-rays, through ultraviolet, optical, and infrared. The archives will be stored at diverse geographical locations. One of the first of these projects, the Sloan Digital Sky Survey (SDSS) is creating a 5-wavelength catalog over 10,000 square degrees of the sky (see http://www.sdss.org/). The 200 million objects in the multi-terabyte database will have mostly numerical attributes in a 100+ dimensional space. Points in this space have highly correlated distributions.

The archive will enable astronomers to explore the data interactively. Data access will be aided by multidimensional spatial and attribute indices. The data will be partitioned in many ways. Small *tag* objects consisting of the most popular attributes will accelerate frequent searches. Splitting the data among multiple servers will allow parallel, scalable I/O and parallel data analysis. Hashing techniques will allow efficient clustering, and pair-wise comparison algorithms that should parallelize nicely. Randomly sampled subsets will allow debugging otherwise large queries at the desktop. Central servers will operate a data pump to support sweep searches touching most of the data. The anticipated queries will require special operators related to angular distances and complex similarity tests of object properties, like shapes, colors, velocity vectors, or temporal behaviors. These issues pose interesting data management challenges.

Keywords

Database, archive, data analysis, data mining, astronomy, scaleable, Internet,

Introduction

Astronomy is about to undergo a major paradigm shift. Data gathering technology is riding Moore's law: data volumes are doubling every 20 months. Data sets are becoming larger, and more homogeneous. For the first time data acquisition and archiving is being designed for online interactive analysis. In a few years it will be much easier to download a detailed sky map or object class catalog, than wait several months to access a telescope. In addition, the online data detail and quality is likely to rival that generated by the typical telescopes.

Several multi-wavelength projects are under way: SDSS, GALEX, 2MASS, GSC-2, POSS2, ROSAT, FIRST and DENIS. Each is surveying a large fraction of the sky. Together they will yield a *Digital Sky*, of interoperating multi-terabyte databases. In time, more catalogs will be added and linked to the existing ones. Query engines will become more sophisticated, providing a uniform interface to all these datasets. In this era, astronomers will have to be just as familiar with mining data as with observing on telescopes.

The Sloan Digital Sky Survey

The Sloan Digital Sky Survey (SDSS) will digitally map about half of the Northern sky in five spectral bands from ultraviolet to the near infrared. It is expected to detect over 200 million objects. Simultaneously, it will measure redshifts for the brightest million galaxies (see http://www.sdss.org/).

The SDSS is the successor to the Palomar Observatory Sky Survey (POSS), which has provided a standard reference data set to all of astronomy for the last 40 years. Subsequent archives will augment the SDSS and will interoperate with it. The SDSS project must not only build the survey hardware, it must also design and implement the software to reduce, calibrate, classify, index, and archive the data so that many scientists can use it.

The SDSS will revolutionize astronomy, increasing the amount of information available to researchers by several orders of magnitude. The SDSS archive will be large and complex: including textual information, derived parameters, multi-band images, spectra, and temporal data. The catalog will allow astronomers to study the evolution of the universe

in great detail. It is intended to serve as the standard reference for the next several decades. After only a month of operation, SDSS found two of the most distant known quasars and several methane dwarfs. With more data, other exotic properties will be easy to mine from the datasets.

The potential scientific impact of the survey is stunning. To realize this potential, data must be turned into knowledge. This is not easy – the information content of the survey will be larger than the entire text contained in the Library of Congress.

The SDSS is a collaboration between the University of Chicago, Princeton University, the Johns Hopkins University, the University of Washington, Fermi National Accelerator Laboratory, the Japanese Participation Group, the United States Naval Observatory, and the Institute for Advanced Study, Princeton, with additional funding provided by the Alfred P. Sloan Foundation, NSF and NASA. The SDSS project is a collaboration between scientists working in diverse areas of astronomy, physics and computer science. The survey will be carried out with a suite of tools developed and built especially for this project – telescopes, cameras, fiber spectrographic systems , and computer software.

SDSS constructed a dedicated 2.5-meter telescope at Apache Point, New Mexico, USA. The telescope has a large, flat focal plane that provides a 3-degree field of view. This design balances the areal coverage of the instrument against the detector's pixel resolution.

Figure 1: The SDSS photometric camera with the 5x6 CCD array contains 120 million pixels. The CCDs in each row have a different filter attached. As the earth rotates, images migrate across the CCD array. The array shifts in synchrony with this movement, giving a 55 second exposure for each object in 5 spectral bands.

The survey has two main components: a photometric survey, and a spectroscopic survey. The photometric survey is produced by drift scan imaging of 10,000 square degrees centered on the North Galactic Cap using five broad-band filters that range from the ultra-violet to the infra-red. The effective exposure is 55 sec, as a patch of sky passes over the focal plane with the earth's rotation. The photometric imaging uses an array of 30x2Kx2K Imaging CCDs, 22 astrometric CCDs, and 2 Focus CCDs. Its 0.4 arcsecond pixel size provides a full sampling of the sky. The data rate from the camera's 120 million pixels is 8 Megabytes per second. The cameras can only be used under ideal atmospheric conditions, but in five years the survey expects to collect 40 terabytes of data from its survey of the northern sky.

The spectroscopic survey will target a million objects automatically chosen from the photometric survey. The goal is to survey a statistically uniform sample of visible objects. Due to the expansion of the universe, the Doppler-shift in the spectral lines of the galaxies is a direct measure of their distance. This spectroscopic survey will produce a three-dimensional map of galaxy distribution, for a volume several orders of magnitude larger than current maps.

The primary targets will be galaxies, selected by a magnitude and surface brightness limit in the r band. This sample of 900,000 galaxies will be complemented with 100,000 very red galaxies, selected to include the brightest galaxies at the cores of galaxy clusters. An automated algorithm will select 100,000 quasar candidates for spectroscopic follow-up, creating the largest uniform quasar survey to date. Selected objects from other catalogs taken at different wavelengths (e.g., FIRST, ROSAT) will also be targeted.

The spectroscopic observations will be done in overlapping 3? circular "tiles". The tile centers are determined by an optimization algorithm, which maximizes overlaps at areas of highest target density. The spectroscopic survey uses two multi-fiber medium resolution spectrographs, with a total of 640 optical fibers. Each fiber is 3 seconds of arc in that diameter providing spectral coverage from 3900 - 9200 Å. The system can measure 5000 galaxy spectra per night. The total number of galaxy spectra known to astronomers today is about 100,000. In only 20 nights of observation, SDSS will double this.

Whenever the Northern Galactic cap is not accessible, SDSS will repeatedly image several areas in the Southern Galactic cap to study fainter objects and identify variable sources.

SDSS has also been developing the software necessary to process and analyze the data. With construction of both hardware and software largely finished, the project has now entered a year of integration and testing. The survey itself will take 5 to 7 years to complete, depending on weather.

The Data Products

The SDSS will create four main data sets: (1) a photometric catalog, (2) a spectroscopic catalog, (3) bitmap images in five color bands, and (4) spectra.

The photometric catalog is expected to contain about 500 distinct attributes for each of one hundred million galaxies, one hundred million stars, and one million quasars. These include positions, fluxes, radial profiles, their errors, and information related to the observations. Each object will have a bitmap "atlas image" in each of the five filters.

The spectroscopic catalog will contain identified emission and absorption lines, and one-dimensional spectra for one million galaxies, 100,000 stars, and 100,000 quasars. Derived custom catalogs may be included, such as a photometric galaxy cluster catalog, or quasar absorption line catalog. In addition there will be a compressed 1TB Sky Map. As shown in Table 1, the total size o f these products is about 3TB.

The SDSS will release this data to the public after a period of thorough verification. This public archive is expected to be the standard reference catalog for the next several decades. This long lifetime presents design and legacy problems. The design of the SDSS archival system must allow the archive to grow beyond the actual completion of the survey. As *the* reference astronomical data set, each subsequent astronomical survey will want to cross-identify its objects with the SDSS catalog, requiring that the archive, or at least a part of it, be dynamic with a carefully defined schema and metadata.

Table 1. Sizes of various SDSS datasets		
Product	**Items**	**Size**
Raw observational data	-	40,000 GB
Redshift Catalog	10^6	2 GB
Survey Description	10^5	1 GB
Simplified Catalog	$3x10^8$	60 GB
1D Spectra	10^6	60 GB
Atlas Images	10^9	1,500 GB
Compressed Sky Map	$5x10^5$	1,000 GB
Full photometric catalog	$3x10^8$	400 GB

The SDSS Archives

Observational data from the telescopes is shipped on tapes to Fermi National Accelerator Laboratory (FNAL) where it is reduced and stored in the *Operational Archive (OA)*, accessible to personnel working on the data processing.

Data reduction and calibration extracts astronomical objects and their attributes from the raw data. Within two weeks the calibrated data is published to the *Science Archive (SA)* ac-

Figure 2. *A conceptual data-flow diagram of the SDSS data.* Telescope data (T) is shipped on tapes to FNAL, where it is processed into the Operational Archive (OA). Calibrated data is transferred into the Master Science Archive (SA) and then to Local Archives (LA). The data gets into the public archives (MPA, PA) after approximately 1-2 years of science verification. These servers provide data for the astronomy community, while a WWW server provides public access.

cessible to all SDSS collaborators. The Science Archive contains calibrated data organized for efficient use by scientists. The SA provides a custom query engine built by the SDSS consortium that uses multidimensional indices and parallel data analysis. Given the amount of data, most queries will be I/O limited, thus the SA design is based on a scalable architecture of many inexpensive servers, running in parallel.

Science archive data is replicated to many Local Archives (LA) managed by the SDSS scientists within another two weeks. The data moves into the public archives (MPA, PA) after approximately 1-2 years of science verification, and recalibration (if necessary).

The astronomy community has standardized on the FITS file format for data interchange [18]. FITS includes an ASCII representation of the data and metadata. All data exchange among the archive is now in FITS format. The community is currently considering alternatives such as XML.

Accessing The Archives

Both professional and amateur astronomers will want access to the archive. Astronomy is a unique science in that there is active collaboration between professional and amateur astronomers. Often, amateur astronomers are the first to see some phenomenon. Most of the tools are designed for professional astronomers, but a public Internet server will provide public access to all the published data. The public will be able to see project status and see various images including the 'Image of the Week'.

The Science Archive and public archives both employ a three-tiered architecture: the user interface, an intelligent query engine, and a data warehouse. This distributed approach provides maximum flexibility, while maintaining portability, by isolating hardware specific features. Both the Science Archive and the Operational Archive are built on top of Objectivity/DB, a commercial object oriented database system.

Analyzing this archive requires a parallel and distributed query system. We implemented a prototype query system. Each query received from the user interface is parsed into a query execution tree (QET) that is then executed by the query engine, which passes the requests to Objectivity/DB for actual execution. Each node of the QET is either a query or a set-operation node, and returns a bag of object-pointers upon execution. The multi-threaded query engine executes in parallel at all the nodes at a given level of the QET.

Results from child nodes are passed up the tree as soon as they are generated. In the case of blocking operators like aggregation, sort, intersection, and difference, at least one of the child nodes must be complete before results can be sent further up the tree. In addition to speeding up the query processing, this as-soon-as-possible data push strategy ensures that even in the case of a query that takes a very long time to complete, the user starts seeing results almost imme-
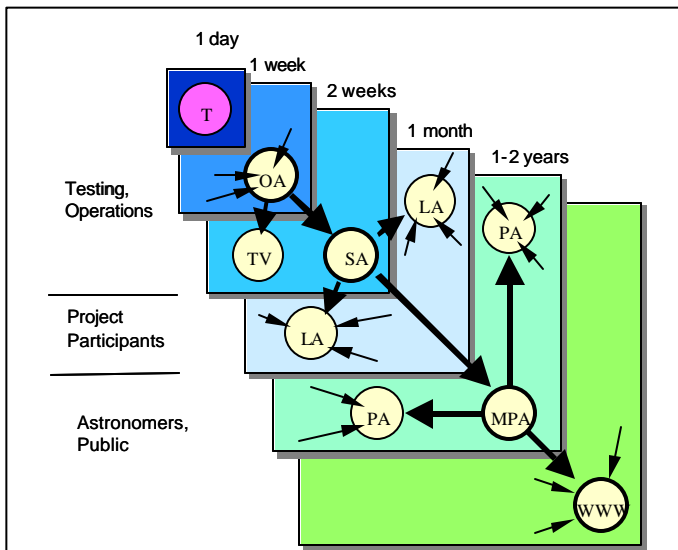
diately, or at least as soon as the first selected object percolates up the tree.

We have been very pleased with Objectivity/DB's ability to match the SDSS data model. For a C++ programmer, the object-oriented database nicely fits the application's data structures. There is no impedance mismatch [18]. On the other hand, we have been disappointed in the tools and performance. The sequential bandwidth is low (about 3 MBps/cpu while the devices deliver more than 10 times that speed) and the cpu overhead seems high. OQL has not been useable, nor have we been able to get the ODBC tools to work. So we have had to do record-at-a-time accesses. There is no parallel query support, so we have had to implement our own parallel query optimizer and run-time system.

Despite these woes, SDSS works on Objectivity/DB and is in pilot use today. Still, we are investigating alternatives. We have designed a relational schema that parallels the SDSS schema. Doing this has exposed some of the known problems with SQL: no support for arrays, poor support for user-defined types, poor support for hierarchical data, and limited parallelism. Still, the schema is fairly simple and we want to see if the better indexing and scanning technology in SQL systems, together with the use of commodity platforms, can offset the language limitations and yield a better cost-performance solution.

In order to evaluate the database design, we developed a list of 20-typical queries that we are translating into SQL. The database schema and these queries are discussed in Section 4. Preliminary results indicate that the parallelism and non-procedural nature of SQL provides real benefits. Time will tell whether the SQL OO extensions make it a real alternative to the OODB solution. You can see our preliminary schema, the 20-queries, and the SQL for them at our web site: http://www.sdss.jhu.edu/SQL.

Typical Queries

The astronomy community will be the primary SDSS user. They will need specialized services. At the simplest level these include the on-demand creation of (color) finding charts, with position information. These searches can be fairly complex queries on position, colors, and other parts of the attribute space.

As astronomers learn more about the detailed properties of the stars and galaxies in the SDSS archive, we expect they will define more sophisticated classifications. Interesting objects with unique properties will be found in one area of the sky. They will want to generalize these properties, and search the entire sky for similar objects.

The most common queries against the SDSS database will be very simple - finding objects in a given small sky region. Another common query will be to distinguish between rare and typical objects, based upon their colors and sizes. Other types of queries will be non-local, like "*find all the quasars brighter than magnitude 22, which have a faint blue galaxy within 5 arcseconds on the sky*". Yet another type of a query

is a search for gravitational lenses: "*find objects within 10 arcseconds of each other which have identical colors, but may have a different brightness*". This latter query is a typical high-dimensional query, since it involves a metric distance not only on the sky, but also in color space. It also shows the need for approximate comparisons and ranked results.

We can make a few general statements about the expected queries: (1) Astronomers work on the surface of the celestial sphere. This contrasts with most spatial applications, which operate in Cartesian 2-space or 3-space. (2) Most of the queries require a linear or at most a quadratic search (single-item or pair-wise comparisons). (3) Many queries are clustering or top rank queries. (4) Many queries are spatial involving a tiny region of the sky. (5) Almost all queries involve user-defined functions. (6) Almost all queries benefit from parallelism and indices. (7) It may make sense to save many of the computed attributes, since others may be interested in them.

Special operators are required to perform these queries efficiently. Preprocessing, like creating regions of mutual attraction, appears impractical because there are so many objects, and because the operator input sets are dynamically created by other predicates.

Geometric Data Organization

Given the huge data sets, the traditional astronomy approach of Fortran access to flat files is not feasible for SDSS. Rather, non-procedural query languages, query optimizers, database execution engines, and database indexing schemes must replace traditional file processing.

This "database approach" is mandated both by computer efficiency (automatic parallelism and pipelining), and by the desire to give astronomers better analysis tools.

The data organization must support concurrent complex queries. Moreover, the organization must efficiently use processing, memory, and bandwidth. It must also support adding new data to the SDSS as a background task that does not disrupt online access.

It would be wonderful if we could use an off-the-shelf object-relational, or object-oriented database system for our tasks. We are optimistic that this will be possible in five years – indeed we are working with vendors toward that goal. As explained presently, we believe that SDSS requires novel spatial indices and novel operators. It also requires a dataflow architecture that executes queries and user-methods concurrently using multiple disks and processors. Current products provide few of these features. But, it is quite possible that by the end of the survey, some commercial system will be adequate.

Spatial Data Structures

The large-scale astronomy data sets consist primarily of records containing numeric data, maps, time-series sensor logs,

and images. The vast majority of the data is essentially geometric. The success of the archive depends on capturing the spatial nature of this large-scale scientific data.

The SDSS data has high dimensionality - each item has thousands of attributes. Categorizing objects involves defining complex domains (classifications) in this N-dimensional space, corresponding to decision surfaces.

The SDSS teams are investigating algorithms and data structures to quickly compute spatial relations, such as finding nearest neighbors, or other objects satisfying a given criterion within a metric distance. The answer set cardinality can be so large that intermediate files simply cannot be created. The only way to analyze such data sets is to pipeline the answers directly into analysis tools. This *data flow* analysis has worked well for parallel relational database systems [2. 3, 4, 5, 9]. We expect that the implementation of these data river ideas will link the archive directly to the analysis and visualization tools.

The typical search of these multi-terabyte archives evaluates a complex predicate in k-dimensional space, with the added difficulty that constraints are not necessarily parallel to the axes. This means that the traditional indexing techniques, well established with relational databases, will not work, since one cannot build an index on all conceivable linear combinations of attributes. On the other hand, one can use the facts that the data are geometric and that every object is a point in this k-dimensional space [11,12]. Data can be quantized into containers. Each container has objects of similar properties, e.g. colors, from the same region of the sky. If the containers are stored as contiguous disk pages, data locality will be high - if an object satisfies a query, it is likely that some of the object's "friends" will as well. There are non-trivial aspects of how to subdivide the containers, when the data has large density contrasts [6].

These containers represent a coarse-grained density map of the data. They define the base of an index tree that tells us whether containers are fully inside, outside or bisected by our query. Only the bisected container category is searched, as the other two are wholly accepted or rejected. A prediction of the output data volume and search time can be computed from the intersection volume.

Indexing the Sky

There is great interest in a common reference frame for the sky that can be used by different astronomical databases. The need for such a system is indicated by the widespread use of the ancient constellations – the first spatial index of the celestial sphere. The existence of such an index, in a more computer friendly form will ease cross-referencing among catalogs.

A common scheme, that provides a balanced partitioning for all catalogs, may seem to be impossible; but there is an elegant solution, that subdivides the sky in a hierarchical fashion.
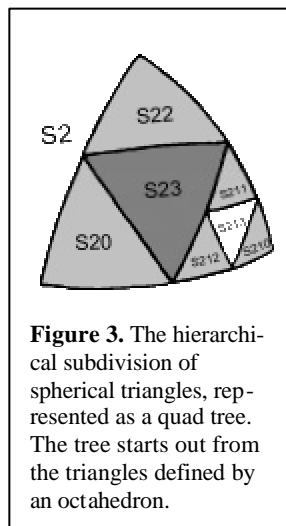

Figure 3. The hierarchical subdivision of spherical triangles, represented as a quad tree. The tree starts out from the triangles defined by an octahedron.

Instead of taking a fixed subdivision, we specify an increasingly finer hierarchy, where each level is fully contained within the previous one. Starting with an octahedron *base set,* each spherical triangle can be recursively divided into 4 sub-triangles of approximately equal areas. Each sub-area can be divided further into additional four sub-areas, ad infinitum. Such hierarchical subdivisions can be very efficiently represented in the form of quad-trees. Areas in different catalogs map either directly onto one another, or one is fully contained by another (see Figure 3.)

We store the object's coordinates on the surface of the sphere in Cartesian form, i.e. as a triplet of *x,y,z* values per object. The *x,y,z* numbers represent only the position of objects on the sky, corresponding to the normal vector pointing to the object. (We can guess the distance for only a tiny fraction (0.5%) of the 200 million objects in the catalog.) While at first this representation may seem to increase the required storage (three numbers per object vs. two angles,) it makes querying the database for objects within certain areas of the celestial sphere very efficient. This technique was used successfully by the GSC project [7]. The coordinates of other celestial coordinate systems (Equatorial, Galactic, Supergalactic, etc) can be constructed from the Cartesian coordinates on the fly.

Using the three-dimensional Cartesian representation of the angular coordinates makes it particularly simple to find objects within a certain spherical distance from a given point, or combination of constraints in arbitrary spherical coordinate systems. They correspond to testing linear combinations of the three Cartesian coordinates instead of complicated trigonometric expressions.

The two ideas, partitioning and Cartesian coordinates merge into a highly efficient storage, retrieval and indexing scheme.
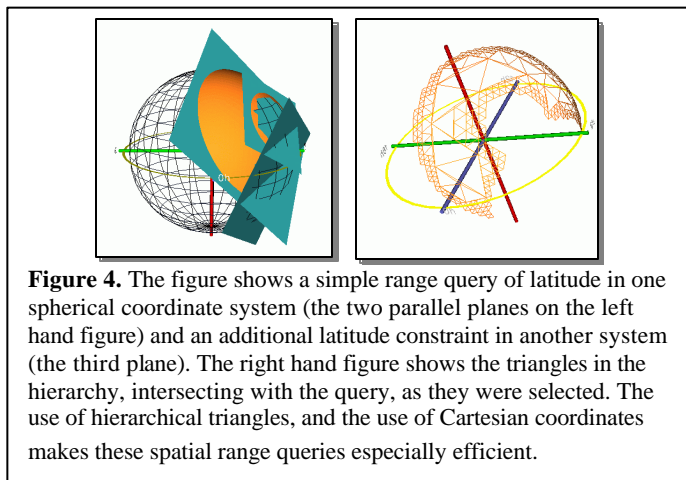

Figure 4. The figure shows a simple range query of latitude in one spherical coordinate system (the two parallel planes on the left hand figure) and an additional latitude constraint in another system (the third plane). The right hand figure shows the triangles in the hierarchy, intersecting with the query, as they were selected. The use of hierarchical triangles, and the use of Cartesian coordinates makes these spatial range queries especially efficient.

We have created a recursive algorithm that can determine which parts of the sky are relevant for a particular query [16]. Each query can be represented as a set of half-space constraints, connected by Boolean operators, all in three-dimensional space.

The task of finding objects that satisfy a given query can be performed recursively as follows. Run a test between the query polyhedron and the spherical triangles corresponding to the tree root nodes. The intersection algorithm is very efficient because it is easy to test spherical triangle intersection. Classify nodes, as fully outside the query, fully inside the query or partially intersecting the query polyhedron. If a node is rejected, that node's children can be ignored. Only the children of bisected triangles need be further investigated. The intersection test is executed recursively on these nodes (see Figure 4.) The SDSS Science Archive implemented this algorithm in its query engine [15]. We are implementing a stored procedure that returns a table containing the IDs of triangles containing a specified area. Queries can then use this table to limit the spatial search by joining answer sets with this table.

Broader Metadata Issues

There are several issues related to metadata for astronomy datasets. First, one must design the data warehouse schema, second is the description of the data extracted from the archive, and the third is a standard representation to allow queries and data to be interchanged among archives.

The SDSS project uses a UML tool to develop and maintain the database schema. The schema is defined in a high level format, and an automated script generator creates the .h files for the C++ classes, and the data definition files for Objectivity/DB, SQL, IDL, XML, and other metadata formats.

About 20 years ago, astronomers agreed on exchanging most of their data in a self-descriptive data format. This format, FITS, standing for the Flexible Image Transport System [17] was primarily designed to handle images. Over the years, various extensions supported more complex data types, both in ASCII and binary form. The FITS format is well supported by all astronomical software systems. The SDSS pipelines exchange most of their data as binary FITS files.

Unfortunately, FITS files do not support streaming data, although data could be blocked into separate FITS packets. The SDSS has implemented an ASCII FITS output stream, using a blocked approach. A binary stream is under development.

We expect large archives to communicate with one another via a standard, easily parseable interchange format. SDSS plans to par-

Figure 5. A typical complex object involving several nearby stars and a galaxy.

ticipate in the definition of interchange formats in XML, XSL, and XQL.

Data Loading

The Operational Archive exports calibrated data to the Science Archive as soon as possible. Datasets are sent in coherent chunks. A chunk consists of several segments of the sky that were scanned in a single night, with all the fields and all objects detected in the fields. Loading data into the Science Archive could take a long time if the data were not clustered properly. Efficiency is important, since about 20 GB arrives after each night of photometric observations.

The incoming data are organized by how the observations were taken. In the Science Archive they are organized into the hierarchy of containers as defined by the multi-dimensional spatial index (Figure 3), according to their colors and positions.

Data loading might bottleneck on creating the clustering units—databases and containers—that hold the objects. Our load design minimizes disk accesses, touching each clustering unit at most once during a load. The chunk data is exported as a binary FITS file from the Operational Archive into the Science Archive. It is first examined to construct an index. This determines where each object will be located and creates a list of databases and containers that are needed. Then data is inserted into the containers in a single pass over the data objects.

The experimental SQL design

SQL Schema

We translated the Objectivity/DB schema into an SQL schema of 25 tables. They were generated from a UML schema by an automated script, and fine-tuned by hand.

The SQL schema has some differences from our Objectivity schema. Arrays cannot be represented in SQL Server, so we broke out the shorter, one-dimensional arrays f[5] as scalar fields f_1,f_2,… The poor indexing in Objectivity/DB forced us to separate star and galaxy objects (for the 2x speedup), while in SQL we were able to merge the two classes and their associations. Object Associations were converted into foreign keys. Other than this, the schema conversion and data extraction and loading was remarkably easy. Detailed information about the data model can be found at (http://www.sdss.jhu.edu/ScienceArchive/doc.html).

The tables can be separated into several broad categories. The first set of tables relate to the photometric observations. A base table, called the *Photo* table, contains the basic photometric attributes of each object. Each record contains about 100 attributes, describing each object detected in the survey, its colors, position in each band, the errors of each quantity, and some classification parameters. The

30% least popular attributes, and a 5x15 radial light profile array are vertically partitioned into a separate *PhotoProfile* table. The array is represented as a BLOB with some user-defined functions to access it.

In the same spirit, the 13 most popular attributes were split off into the Tag objects in the Objectivity/DB design, to make simple queries more efficient. In SQL the *Tag* table object is represented as one or more indexes on the *Photo* table. After this vertical partitioning we also segmented the data horizontally in the Objectivity/DB design. After an object classification, still performed by the pipelines, extended objects, classified as galaxies, and compact objects classified as stars are stored in separate object classes, since they will typically be queried separately most of the time. In SQL the design unifies the two tables, and assumes that clustering and partitioning will be done by the DBMS across the multiple disks and servers. We also created a few attributes from our internal spatial indices – these are the hash codes corresponding to a bit interleaved address in the triangular mesh or on the k-d tree.

There is a particular subtlety in dealing with merged objects. On the sky one often finds a nearby star superimposed over the image of a distant galaxy (see Figure 5). These can often be recognized as such, and deblended. This deblending process creates a tree of object relations, where a parent object may have two or more children, and so on. Also, about 10% of the sky is observed multiple times. All the detections of the same object will have a common object identifier. A unique *primary* selected by its sky position. Each *Photo* object record is marked as a primary or secondary, and all instances of the object have the same object-identifier.

Another set of tables is related to the hierarchy of observations and the data processing steps. The SDSS project observes approximately 130-degree long 2.5-degree wide *stripes* on the sky. Each stripe is actually two *runs* from two different nights of observation. Each run has six 5-color columns, corresponding to the six CCD columns in the camera (see figure 1) separated by 90% of the width of the CCDs. These columns have 10% overlap on each side, and are woven together to form a seamless mosaic of the 130x2.5 degree stripe. Each 5-color column is split into about 800 *fields*. Each field is a 5-color 2048x1489 2-byte per pixel image. There are separate tables with the metadata for stripes, runs, and fields. Each field carries about 60 attributes, consisting of its precise calibration data, and the coefficients of

Figure 6. Each night of observation produces a 130 x 2.5 run of 5 colors. The columns of two adjacent runs have a 10% overlap and can be mosaiced together to form a strip. Stripes are partitioned into fields.

the transformation that maps pixel positions onto absolute sky coordinates. Chunks and Segments carry the observation date and time, the software version used during the data reduction process, and various parameters of the instruments during the observing run.

Another set of tables is related to the spectroscopic observations. They capture the process of target selection and eventual observation. There is a separate *SpectroTarget* table, corresponding to objects selected from the photometry to be observed with the spectrographs. Not all of them will be observed. The observed objects will then be classified as a galaxy, star, quasar, blank sky, and unknown. The observed objects have various attributes, a list of emission and absorption lines detected, estimated redshift, its errors, and quality estimate, stored in a *Spectro* table. Every object has a different number if lines, so the lines are stored in a *Lines* table. Each record has the object identifier, line type, (emission or absorption), lab wavelength, and rest-frame wavelength, line identifier, line width, and strength, and a few fitting parameters, and of course uncertainly estimates. There is also a *Plate* table describing the layout of a spectroscopic observation, since 640 objects are measured simultaneously.

Finally, there is a table to capture cross-reference information about SDSS objects also detected in other catalogs, if a unique identification can be made. This cross-reference table can evolve as the need arises.

SQL Queries

We developed a set of 20 queries that we think characterize the kinds of questions Astronomers are likely to ask the SDSS archives. This is much in the spirit of the Sequoia 2000 benchmark of [14]. We are in the process of translating these queries into SQL statements and evaluating their performance on a relational system.

Here follow the queries and a narrative description of how we believe they will be evaluated.

Q1: Find all galaxies with unsaturated pixels within 1 arcsecond of a given point in the sky (right ascension and declination). This is a classic spatial lookup. We expect to have a quad-tree spherical triangle index with object type (star, galaxy, …) as the first key and then the spatial attributes. So this will be a lookup in that quad-tree. Select those galaxies that are within one arcsecond of the specified point.

Q2: Find all galaxies with blue surface brightness between 23 and 25 mag per square arcseconds, and -10<super galactic latitude (sgb) <10, and declination less than zero. This searches for all galaxies in a certain region of the sky with a specified brightness in the blue spectral band. The query uses a different coordinate system, which ismust first be converted to the hierarchical triangles of Figure 3 and section 3.5. It is then a set of disjoint table scans, each having a compound simple predicate representing the spatial
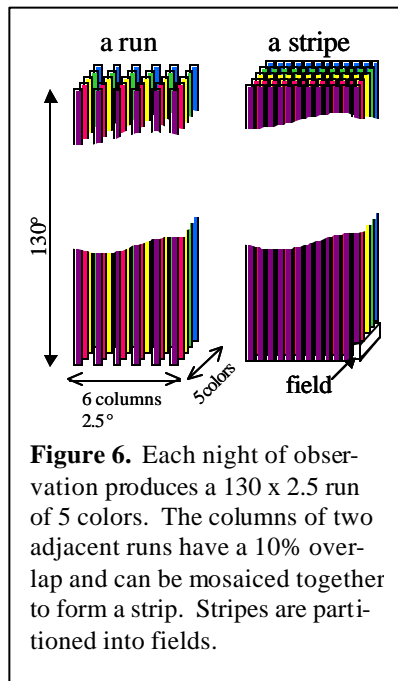
boundary conditions and surface brightness test.

Q3: Find all galaxies brighter than magnitude 22, where the local extinction is >0.75. The local extinction is a map of the sky telling how much dust is in that direction, and hence how much light is absorbed by that dust. The extinction grid is stored as a table with one square arcminute resolution – about half a billion cells. The query is either a spatial join of bright galaxies with the extinction grid table, or the extinction is stored as an attribute of each object so that this is just a scan of the galaxies in the Photo table.

Q4: Find galaxies with a surface brightness greater than 24 with a major axis 30"<d<1', in the red-band, and with an ellipticity>0.5. . Each of the 5 color bands of a galaxy will have been pre-processed into a bitmap image which is broken into 15 concentric rings. The rings are further divided into octants. The intensity of the light in each ring is analyzed and recorded as a 5x15 array. The array is stored as an object (SQL blob in our type impoverished case). The concentric rings are pre-processed to compute surface brightness, ellipticity, major axis, and other attributes. Consequently, this query is a scan of the galaxies with predicates on precomputed properties.

Q5: Find all galaxies with a deVaucouleours profile ($r^{1/4}$ falloff of intensity on disk) and the photometric colors consistent with an elliptical galaxy. The deVaucouleours profile information is precomputed from the concentric rings as discussed in Q4. This query is a scan of galaxies in the Photo table with predicates on the intensity profile and color limits.

Q6: Find galaxies that are blended with a star, output the deblended magnitudes.
Preprocessing separates objects that overlap or are related (a binary star for example). This process is called deblending and produces a tree of objects; each with its own 'deblended' attributes such as color and intensity. The parent child relationships are represented in SQL as foreign keys. The query is a join of the deblended galaxies in the photo table, with their siblings. If one of the siblings is a star, the galaxy's identity and magnitude is added to the answer set.

Q7: Provide a list of star-like objects that are 1% rare for the 5-color attributes. This involves classification of the attribute set and then a scan to find objects with attributes close to that of a star that occur in rare categories.

Q8: Find all objects with spectra unclassified.
This is a sequential scan returning all objects with a certain precomputed flag set.

Q9: Find quasars with a line width >2000 km/s and 2.5<redshift<2.7. This is a sequential scan of quasars in the Spectro table with a predicate on the redshift and line width. The Spectro table has about 1.5 million objects having a known spectrum but there are only 100,00 known quasars.

Q10: Find galaxies with spectra that have an equivalent width in H_a >40Å (H_a is the main hydrogen spectral line.)

This is a join of the galaxies in the Spectra table and their lines in the Lines table.

Q11: Find all elliptical galaxies with spectra that have an anomalous emission line. This is a sequential scan of galaxies (they are indexed) that have ellipticity above .7 (a precomputed value) with emission lines that have been flagged as strange (again a precomputed value).

Q12: Create a grided count of galaxies with u-g>1 and r<21.5 over 60<declination<70, and 200<right ascension<210, on a grid of 2', and create a map of masks over the same grid. Scan the table for galaxies and group them in cells 2 arc-minutes on a side. Provide predicates for the color restrictions on u-g and r and to limit the search to the portion of the sky defined by the right ascension and declination conditions. Return the count of qualifying galaxies in each cell. Run another query with the same grouping, but with a predicate to include only objects such as satellites, planets, and airplanes that obscure the cell. The second query returns a list of cell coordinates that serve as a mask for the first query. The mask may be stored in a temporary table and joined with the first query.

Q13: Create a count of galaxies for each of the HTM triangles (hierarchal triangular mesh) which satisfy a certain color cut, like 0.7u-0.5g-0.2 and i-magr<1.25 and r-mag<21.75, output it in a form adequate for visualization. This query is a sequential scan of galaxies with predicates for the color magnitude. It groups the results by a specified level in the HTM hierarchy (obtained by shifting the HTM key) and returns a count of galaxies in each triangle together with the key of the triangle.

Q14: Provide a list of stars with multiple epoch measurements, which have light variations >0.1 magnitude. Scan for stars that have a secondary object (observed at a different time) with a predicate for the light variations.

Q15: Provide a list of moving objects consistent with an asteroid. Objects are classified as moving and indeed have 5 successive observations from the 5 color bands. So this is a select of the form: select moving object where $\sqrt{((deltax5-deltax1)^2 + (deltay5-deltay1)^2)} < 2$ arc seconds.

Q16: Find all star-like objects within DeltaMagnitde of 0.2 of the colors of a quasar at 5.5<redshift<6.5. Scan all objects with a predicate to identify star-like objects and another predicate to specify a region in color space within 'distance' 0.2 of the colors of the indicated quasar (the quasar colors are known).

Q17: Find binary stars where at least one of them has the colors of a white dwarf. Scan the Photo table for stars with white dwarf colors that are a child of a binary star. Return a list of unique binary star identifiers.

Q18: Find all objects within 1' of one another other that have very similar colors: that is where the color ratios u-g, g-r, r-I are less than 0.05m. (Magnitudes are logarithms so these are ratios.) This is a gravitational lens query. Scan for objects in the Photo table and compare them to all objects within one arcminute of the object. If the color ratios match, this is a candidate object. We may precompute the five nearest neighbors of each object to speed up queries like this.

Q19: Find quasars with a broad absorption line in their spectra and at least one galaxy within 10". Return both the quasars and the galaxies. Scan for quasars with a predicate for a broad absorption line and use them in a spatial join with galaxies that are within 10 arc-seconds. The nearest neighbors may be precomputed which makes this a regular join.

Q20: For a galaxy in the BCG data set (brightest color galaxy), in 160<right ascension<170, 25<declination<35, give a count of galaxies within 30" which have a photoz within 0.05 of the BCG. First form the BCG (brightest galaxy in a cluster) table. Then scan for galaxies in clusters (the cluster is their parent object) with a predicate to limit the region of the sky. For each galaxy, test with a sub-query that no other galaxy in the same cluster is brighter. Then do a spatial join of this table with the galaxies to return the desired counts.

Analysis and Visualization

We do not expect many astronomers to know SQL. Rather we expect to provide graphical data analysis tools. Much of the analysis will be done like a spreadsheet with simple equations. For more complex analysis, the astronomers will want to apply programs written in Java, C++, JavaScript, VB, Perl, or IDL to analyze objects.

Answers to queries will be steamed to a visualization and analysis engine that SDSS is building using VTK and Java3D. Presumably the astronomer will examine the rendered data, and either drill down into it or ask for a different data presentation, thereby steering the data analysis.

In the spirit of the database query statements, here are a dozen data visualization and analysis scenarios.

VA1: Generate a 2D scatter plot of two selected attributes. This is the bread and butter of all data analysis tools. The user wants to point at some data point and then drill down on it. It is also important to subset the data, do regressions, and other statistical tests of the data.

VA2: Generate a 3D scatter-plot of three selected attributes, and at each point use color, shape, size, icon, or animation to display additional dimensions. This is the next step in visualization after VA1, handling higher dimensional data.

VA3: Superpose objects (tick marks or contours) over an image. This allows analysts to combine two or more visualizations in one pane and compare them visually.

VA4: Generate and 2D and 3D plot of a single scalar field over the sky, in various coordinate systems. Astronomers have many different coordinate systems. They are all just affine transformations of one another.   Sometimes they want polar, sometimes Lambert, sometimes they just want to change the reference fame. This is just a requirement that the visualization system supports the popular projections, and allows new ones to be easily added.

VA5: Visualize condensed representations or aggregations. For example compute density of objects in 3D space, phase space, or attribute space and render the density map.

VA6: 2D/3D scatter plots of objects, with Euclidean proximity links in other dimensions represented. Use connecting lines to show objects that are closely linked in attribute space and in 3 space.

VA7: Allow interactive settings of thresholds for volumetric visualizations, showing translucent iso-surfaces of 3D functions. This kind of visualization is common in parts-explosion of CAD systems; it would also be useful in showing the volumetric properties of higher-dimensional data spaces.

VA8: Generate linked multi-pane data displays. In steering the computation, scientists want to see several "windows" into the dataset, each window showing one of the above displays.   As the analyst changes the focus of one window, all the other windows should change focus in unison. This includes subsetting the data, zooming out, or panning across the parameter space. The http://aai.com/ Image Explorer is and example of such a tool.

Scalable Server Architectures

The SDSS data is too large to fit on one disk or even one server. The base-data objects will be spatially partitioned among the servers. As new servers are added, the data will repartition.   Some of the high-traffic data will be replicated among servers. In the near term, designers must specify the partitioning and index schemes, but we hope that in the long term, the DBMS will automate this design task as access patterns and data volumes change.

Accessing large data sets is primarily I/O limited. Even with the best indexing schemes, some queries must scan the entire data set. Acceptable I/O performance can be achieved with expensive, ultra-fast storage systems, or with many servers operating in parallel. We are exploring the use of inexpensive servers and storage to allow inexpensive interactive data analysis.

As reviewers pointed out, we could buy large SMP servers that offer 1GBps IO. Indeed, our colleagues at Fermi Lab

are using SGI equipment for the initial processing steps. The problem with the supercomputer or mini-supercomputer approach is that the processors, memory, and storage seem to be substantially more expensive than commodity servers – we are able to develop on inexpensive systems, and then deploy just as much processing as we can afford. The "slice-price" for processing and memory seems to be 10x lower than for the high-end servers, the storage prices seems to be 3x lower, and the networking prices seem to be about the same (based on www.tpc.org prices).

We are still exploring what constitutes a balanced system design: the appropriate ratio between processor, memory, network bandwidth, and disk bandwidth. It appears that Amdahl's balance law of one instruction per bit of IO applies for our current software (application + SQL+OS).

Using the multi-dimensional indexing techniques described in the previous section, many queries will be able to select exactly the data they need after doing an index lookup. Such simple queries will just pipeline the data and images off of disk as quickly as the network can transport it to the astronomer's system for analysis or visualization.

When the queries are more complex, it will be necessary to scan the entire dataset or to repartition it for categorization, clustering, and cross comparisons. Experience will teach us the necessary ratio between processor power, main memory size, IO bandwidth, and system-area-network bandwidth.

Our simplest approach is to run a **scan machine** that continuously scans the dataset evaluating user-supplied predicates on each object [1]. We are building an array of 20 nodes. Each node has dual Intel 750 MHz processors, 256MB of RAM, and 8x40GB EIDE disks on dual disk controllers (6TB of storage and 20 billion instructions per second in all). Experiments show that one node is capable of reading data at 120 MBps while using almost no processor time – indeed, it appears that each node can apply fairly complex SQL predicates to each record as the data is scanned. If the data is spread among the 20 nodes, they can scan the data at an aggregate rate of 2 GBps. This 100 K$ system could scan the complete (year 2005) SDSS catalog every 2 minutes. By then these machines should be 10x faster. This should give near-interactive response to most complex queries that involve single-object predicates.

Many queries involve comparing, classifying or clustering objects. We expect to provide a second class of machine, called a **hash machine** that performs comparisons within data clusters. Hash machines redistribute a subset of the data among all the nodes of the cluster. Then each node processes each hash bucket at that node. This parallel-clustering approach has worked extremely well for relational databases in joining and aggregating data. We believe it will work equally well for scientific spatial data.

The hash phase scans the entire dataset, selects a subset of the objects based on some predicate, and "hashes" each object to the appropriate buckets – a single object may go to several buckets (to allow objects near the edges of a region to go to all the neighboring regions as well). In a second phase all the objects in a bucket are compared to one another. The output is a stream of objects with corresponding attributes.

These operations are analogous to relational hash-join, hence the name [5]. Like hash joins, the hash machine can be highly parallel, processing the entire database in a few minutes. The application of the hash-machine to tasks like finding gravitational lenses or clustering by spectral type or by redshift-distance vector should be obvious: each bucket represents a neighborhood in these high-dimensional spaces. We envision a non-procedural programming interface to define the bucket partition function and to define the bucket analysis function.

The hash machine is a simple form of the more general dataflow programming model in which data flows from storage through various processing steps. Each step is amenable to partition parallelism. The underlying system manages the creation and processing of the flows. This programming style has evolved both in the database community [4, 5, 9] and in the scientific programming community with PVM and MPI [8]. This has evolved to a general programming model as typified by a river system [2, 3, 4].

We propose to let astronomers construct dataflow graphs where the nodes consume one or more data streams, filter and combine the data, and then produce one or more result streams. The outputs of these rivers either go back to the database or to visualization programs. These dataflow graphs will be executed on a **river-machine** similar to the scan and hash machine. The simplest river systems are sorting networks. Current systems have demonstrated that they can sort at about 100 MBps using commodity hardware and 5 GBps if using thousands of nodes and disks [13].

With time, each astronomy department will be able to afford local copies of these machines and the databases, but to start they will be network services. The scan machine will be interactively scheduled: when an astronomer has a query, it will be added to the query mix immediately. All data that qualifies is sent back to the astronomer, and the query completes within the scan time. The hash and river machines will be batch scheduled.

Desktop Data Analysis

Most astronomers will not be interested in all of the hundreds of attributes of each object. Indeed, most will be interested in only 10% of the entire dataset – but different communities and individuals will be interested in a different 10%.

We plan to isolate the 10 most popular attributes (3 Cartesian positions on the sky, 5 colors, 1 size, 1 classification parameter) into a compact table or index. We will build a spatial index on these attributes that will occupy much less space and thus can be searched more than 10 times faster, if no other attributes are involved in the query. This is the standard technique of *covering indices* in relational query processing.

Large disks are available today, and within a few years 100GB disks will be common. This means that all astronomers can have a vertical partition of the 10% of the SDSS on their desktops. This will be convenient for targeted searches and for developing algorithms. But, full searchers will still be much faster on servers because they have mo re IO bandwidth and processing power.

The scan, hash, and river machines can also apply vertical partitioning to reduce data movement and to allow faster scans of popular subsets.

We also plan to offer a 1% sample (about 10 GB) of the whole database that can be used to quickly test and debug programs. Combining partitioning and sampling converts a 2 TB data set into 2 gigabytes, which can fit comfortably on desktop workstations for program development.

Distributed Analysis Environment

It is obvious, that with multi-terabyte databases, not even the intermediate data sets can be stored locally. The only way this data can be analyzed is for the analysis software to directly communicate with the data warehouse, implemented on a server cluster, as discussed above. An **Analysis Engine** can then process the bulk of the raw data extracted from the archive, and the user needs only to receive a drastically reduced result set.

Given all these efforts to make the server parallel and distributed, it would be stupid to ignore I/O or network bottlenecks at the analysis level. Thus it is obvious that we need to think of the analysis engine as part of the distributed, scalable computing environment, closely integrated with the database server itself. Even the division of functions between the server and the analysis engine will become fuzzy — the analysis is just part of the river-flow described earlier. The pool of available CPU's will be allocated to each task.

The analysis software itself must be able to run in parallel. Since it is expected that scientists with relatively little experience in distributed and parallel programming will work in this environment, we need to create a carefully crafted application development environment, to aid the construction of customized analysis engines. Data extraction needs to be considered also carefully. If our server is distributed and the analysis is on a distributed system, the extracted data should also go directly from one of the servers to one of the many Analysis Engines. Such an approach will also distribute the network load better.

Sky Server

Some of us were involved in building the Microsoft TerraServer (http://www.TerraServer.Microsoft.com/) which is a website giving access to the photographic and topographic maps of the United States Geological Survey. This website has been popular with the public, and is starting to be a portal to other spatial and spatially related data (e.g., encyclopedia articles about a place.)

We are in the process of building the analog for astronomy: SkyServer (http://www.skyserver.org/). Think of it as the TerraServer looking up rather than down. We plan to put online the publicly available photometric surveys and catalogs as a collaboration among astronomical survey projects. We are starting with Digitized Palomar Observatory Sky Survey (POSS-II) and the preliminary SDSS data. POSS-II covers the Northern Sky in three bands with arcsecond pixels at 2 bits per pixel. POSS-II is about 3 TB of raw image data. In addition, there is a catalog of approximately one billion objects extracted from the POSS data. The next step will add the 2 Micron All Sky Survey (2MASS) that covers the full sky in three near-infrared bands at 2-arcsecond resolution. 2MASS is an approximately 10 TB dataset. We are soliciting other datasets that can be added to the SkyServer.

Once these datasets are online, we hope to build a seamless mosaic of the sky from them, to provide catalog overlays, and to build other visualization tools that will allow users to examine and compare the datasets. Scientists will be able to draw a box around a region, and download the source data and other datasets for that area of the sky. Other surveys will be added later to cover other parts of the spectrum.

Summary

Astronomy is about to be revolutionized by having a detailed atlas of the sky available to all astronomers, providing huge databases of detailed and high-quality data available to all. If the archival system of the SDSS is successful, it will be easy for astronomers to pose complex queries to the catalog and get answers within seconds, and within minutes if the query requires a complete search of the database.

The SDSS datasets pose interesting challenges for automatically placing and managing the data, for executing complex queries against a high-dimensional data space, and for supporting complex user-defined distance and classification metrics.

The efficiency of the instruments and detectors used in the observations is approaching 80%. The factor limiting resolution is the Earth atmosphere. There is not a large margin for a further dramatic improvement in ground-based instruments.

On the other hand, the SDSS project is "riding Moore's law": the data set we collect today – at a linear rate – will be much more manageable tomorrow, with the exponential growth of CPU speed and storage capacity. The scalable archive design presented here will be able to adapt to such changes.

Acknowledgements

References

[1] Acharya, S. R. Alonso, M. J. Franklin, S. B. Zdonik, "Broadcast Disks: Data Management for Asymmetric Communications Environments." SIGMOD Conference 1995: 199-210.

[2] Arpaci-Dusseau, R. Arpaci-Dusseau, A., Culler, D. E., Hellerstein, J. M., Patterson, D. A. ,"The Architectural Costs of Streaming I/O: A Comparison of Workstations, Clusters, and SMPs", Proc. Fourth International Symposium On High-Performance Computer Architecture (HPCA), Feb 1998.

[3] Arpaci-Dusseau, R. H. , Anderson, E.,.Treuhaft, N., Culler, D.A., Hellerstein, J.M., Patterson, D.A., Yelick, K., "Cluster I/O with River: Making the Fast Case Common." IOPADS '99 pp 10-22, http://now.cs.berkeley.edu/River/

[4] Barclay, T. Barnes, R., Gray, J., Sundaresan, P., "Loading Databases Using Dataflow Parallelism." SIGMOD Record 23(4): 72-83 (1994)

[5] DeWitt, D.J. Gray, J., "Parallel Database Systems: The Future of High Performance Database Systems." CACM 35(6): 85-98 (1992)

[6] Csabai, I., Szalay, A.S. and Brunner, R., "Multidimensional Index For Highly Clustered Data With Large Density Contrasts," in *Statistical Challenges in Astronomy II*, eds. E. Feigelson and A. Babu, (Wiley), 447 (1997).

[7] Greene, G. et al, "The GSC-I and GCS-II Databases: An Object Oriented Approach", in *New Horizons from Multi-Wavelength Sky Surveys*, eds. B. McLean et al, Kluwer, p.474 (1997)

[8] Gropp, W., Huss-Lederman, H., *MPI the Complete Reference: The MPI-2 Extensions, Vol. 2,* MIT Press, 1998, ISBN: 0262571234

[9] Graefe, G., "Query Evaluation Techniques for Large Databases". ACM Computing Surveys 25(2): 73-170 (1993)

[10] Hartman, A. of Dell Computer, private communication. Intel Corporation has generously provided the SDSS effort at Johns Hopkins with Dell Computers.

[11] Samet, H., *Applications of Spatial Data Structures: Computer Graphics, Image Processing, and GIS*, Addison-Wesley, Reading, MA, 1990. ISBN 0-201-50300-0.

[12] Samet, H., *The Design and Analysis of Spatial Data Structures*, Addison-Wesley, Reading, MA, 1990. ISBN 0-201-50255-0.

[13] http://research.microsoft.com/barc/SortBenchmark/

[14] Stonebraker, M., Frew, J., Gardels, K., Meredith, J., "The Sequoia 2000 Benchmark." Proc. ACM SIGMOD, pp. 2-11, 1993.

[15] Szalay, A.S. and Brunner, R.J.: "Exploring Terabyte Archives in Astronomy", in *New Horizons from Multi- Wavelength Sky Surveys*, IAU Symposium 179, eds. B. McLean and D. Golombek, p.455. (1997).

[16] Szalay, A.S., Kunszt, P. and Brunner, R.J.: "Hierarchical Sky Partitioning," Astronomical Journal, to be submitted, 2000.

[17] Wells, D. C., Greisen, E. W., and Harten, R. H., "FITS: A Flexible Image Transport System," Astronomy and Astrophysics Supplement Series, 44, 363-370, 1981.

[18] Zadonic, S., Maier, D., *Readings in Object Oriented Database Systems*, Morgan Kaufmann, San Francisco, CA, 1990. ISBN 1-55860-000-0