FERMILAB-CONF-06-423-CD

Astronomical Data Analysis Software and Systems XVI **P1-07**
ASP Conference Series, Vol. XXX, 2006
R. Shaw, F. Hill and D. Bell, eds.

Running the Sloan Digital Sky Survey Data Archive Server

Eric H. Neilsen, Jr., Chris Stoughton

Fermi National Accelerator Laboratory

Abstract. The Sloan Digital Sky Survey (SDSS) Data Archive Server (DAS) provides public access to over 12Tb of data in 17 million files produced by the SDSS data reduction pipeline. Many tasks which seem trivial when serving smaller, less complex data sets present challenges when serving data of this volume and technical complexity. The included output files should be chosen to support as much science as possible from publicly released data, and only publicly released data. Users must have the resources needed to read and interpret the data correctly. Server administrators must generate new data releases at regular intervals, monitor usage, quickly recover from hardware failures, and monitor the data served by the DAS both for contents and corruption. We discuss these challenges, describe tools we use to administer and support the DAS, and discuss future development plans.

1. Introduction

The Sloan Digital Sky Survey, described in Stoughton et al. 2002, consists of images, spectra, catalogs of objects and object parameters derived from these images and spectra, and a significant body of metadata describing the collection and processing of this data. Most users are primarily interested in the object catalogs and corresponding science parameters. The catalog archive server (CAS), described in more detail in Szalay et al. 2002, provides online access to a database containing the catalogs of objects and object parameters and much of the metadata, allowing users to search the survey using complex queries and even perform some limited analysis. While the remote processing of queries relieves users of the need to download significant volumes of data, the limited capacity of the server necessarily limits the computing resources (such as processing time and memory) available to the user. Furthermore, the database does not include all of the lower level data generated by the pipeline. Some users therefore require an additional data access mechanism.

The Data Archive Server (DAS) provides access to the raw output of the SDSS data reduction pipelines. The DAS access mechanism is very basic; a standard HTTP server (apache) serves the data files produced by the pipeline, and a few CGI scripts help the user generate lists of desired files based on observational metadata. The DAS server mounts the filesystems containing the files with the data using NFS, and serves the data from directory structures (or "link tree") that contain links to the files. Each data release has a corresponding link tree that contains links only to those files that are part of that data release. Note that links to individual files must be used, because the directories that contain the files themselves often contain other files that are not part of a data release. We do not simply create copies of the data for each data release because of the disk space required; data release 5 holds over 12 Tb of data in 17 million files. Because most files in one data release are also present in the next, using links instead of copies saves significant disk space.

In some cases, a user will copy an entire data release, in which case the use of the DAS is relatively straightforward, if time consuming: the user can recursively download everything. In most cases, though, users are only interested in a subset of the files served, and use other mechanisms (such as the CAS) to help determine which files are needed. For example, a user might search the CAS and find a list of objects meeting some science criteria, and wish to download the raw spectra of these objects. A query to the CAS will provide observation dates and plate numbers for the spectra, and these values can be fed in to the DAS CGI scripts to produce a list of files containing the desired spectra. A non-interactive HTTP client such as wget can then retrieve the files in the list.

Despite the simplicity and limited functionality of the DAS server, administration of the server itself presents challenges.

2. The SDSS Pipeline and Interpretation of the Data

The first challenge is the nature of the data itself. The imaging camera collects data in "runs," continuous reads of pixel values collected as the telescope drifts along "stripes." The data acquisition software organizes these pixels into "frames," fits images of uniform height. Each run, therefore, consists of a collection of frames for each CCD in the camera. The camera has 30 data CCDs, arranged into five rows and six columns. Each row has a different filter, so an individual fits image can be specified using the run, frame, filter, and camera column.

The photometric pipeline produces a number of output files for each frame, such as corrected frames, color jpeg images, and object catalogs. Because the photometric pipeline generates these files by frame, it names and organizes the output files by frame, and users wishing to download these files must determine which run, camera column, filter, and frames are of interest. The pipeline produces other output that describes larger sets of the data. For example, the pipeline produces quality control plots that apply to many frames. Even in these cases, users must know the run, filter, camera column, and frame of interest to know which parts of the plots, for example, are of interest.

After the imaging data has been produced and calibrated, target selection software can be used to plan spectroscopy. Spectroscopic data collection and processing is similarly observation oriented; the spectrographs produce spectra taken using different plates, and each object corresponds to a fiber in a plate. A plate may be observed on several nights, so to completely specify a spectrum a user must know the plate, night of observation, and fiber of interest.

Users must not only be able to obtain the files, they must also be able to read and interpret them. Although the pipeline produces files either in self describing ASCII files or in standard formats such as fits, reading these files is not always straightforward. For example, fits files that hold mask, PSF, and atlas image information do so in fits binary tables (including "heap" extensions) rather than fits images, and direct interpretation of the table contents is not straightforward; we supply programs to generate fits images from these files. In many cases, the proper interpretation of the values stored in these files requires a detailed understanding of the algorithms used by the pipelines. While most of the information needed can be found in online documentation, and the scope and complexity of the pipelines can make locating and understanding the documentation a significant obstacle. Users often require support from SDSS personnel with expertise in the relevant parts of the pipeline to properly interpret the files provided; a helpdesk supported by the experts is essential.

3. Verification and Validation of a Data Release

To ensure that we are serving the correct data, we have constructed tools that:

- verify that all files that should be served by the DAS are present,
- verify that no files that should not be served are, and
- check each file being served for corruption.

Administration tools read the definition of the data release from a collection of parameter files, and generate a list of directories and their contents based on that definition. In each directory, the tools generate and maintain a "file list file:" a list of files that should be present in that directory and their expected checksums, and store a list of the file list files that define the contents of all directories in the data release. Automated tasks then check the contents and checksums of files served by the DAS against the file lists.

This approach has several drawbacks, some of which may be addressed in future releases. First, corrupt files do not always get replaced by files identical to the originals. Some files are regenerated rather than restored from back up, in which case the data content will be the same but dates in headers may differ. The file list cannot be used to verify that the data content has not changed, and must be updated whenever this occurs.

The calculation of the checksum in a separate step from file generation introduces a second flaw: it is possible for the file to become corrupt before the checksum is calculated.

Finally, the storage of validation information in a custom format reduces their usefulness to users.

Several alternate approaches address some or all of these problems. Our fits writer could be modified to include checksum information in headers. Alternately, it could be modified to compress output files using GNU gzip, which includes a checksum. Unfortunately, neither of these approaches are practical for existing data. The custom checksum file format could be replaced by md5sum output, which would result in a standard format with minimal modification of our tools, but would only address the last of the known problems listed above.

4. File Systems and Large Directory Structures

To publish a new data release on the DAS, we must generate a directory structure containing symbolic links to each of the files that constitute the data release. (Links to directories are not acceptable, because the directories hosting the data may contain data that is not part of the data release.)

This apparently simple task is surprisingly time consuming; generating the populated directory structure for the fifth data release, containing 17 million files, took more than 24 hours. The file systems we tried (ext3 and xfs) are better optimized for file creation than deletion, so removing an existing data structure was even more time consuming. Restoration of these directory structures from backup or regeneration to correct errors was therefore a problem.

We find keeping each data release on its own partition, and using a raw disk dump of the partition as a backup, to be a more practical solution. If we write null bytes over the partition before formatting it, and use it only for the data release, a backup copy of the partition can be compressed to a modest size. (Recall that the partition contains only links, not real data.) Saving and restoring such partitions is much faster than rebuilding the links.

5. Future Directions

While most users of the DAS retrieve only a handful of files, a handful of users retrieve large fractions of the available data. These users are typically generating local full or partial mirrors. The receiving sites are often overseas, which can slow the already long transfers. We are exploring the use of P2P software, particularly bitTorrent, to take advantage of existing mirrors to improve the download time by allowing clients to retrieve different parts of the data release from different mirrors.

While bitTorrent appears promising, several challenges must be met. The data release must be divided into torrents, which in turn must be divided into pieces. In the clients we have studied so far, memory availability limits the number of pieces a client can manage, the size of each piece, and the number of torrents managed. For bitTorrent to be used, we will need either to be very careful in the construction of our torrents, or construct a client that manages memory differently.

References

http://www.bittorrent.org/protocol.html

Stoughton, C. et al. 2002, AJ, 123, 485

Szalay, A. et al. 2002, Proceedings of the 2002 ACM SIGMOD International Conference of Management of Data, 570-581