AstroPortal: A Science Gateway for Large-scale Astronomy Data Analysis

Ioan Raicu*, Ian Foster*+, Alex Szalay#, Gabriela Turcu*

*Computer Science Dept.
The University of Chicago
iraicu@cs.uchicago.edu

+Math and Computer Science Div.
Argonne National Laboratory
foster@mcs.anl.gov

#Dept. of Physics and Astronomy
The Johns Hopkins University
szalay@jhu.edu

Abstract

The creation of large digital sky surveys presents the astronomy community with tremendous scientific opportunities. However, these astronomy datasets are generally terabytes in size and contain hundreds of millions of objects separated into millions of files—factors that make many analyses impractical to perform on small computers. To address this problem, we have developed a Web Services-based system, AstroPortal, that uses grid computing to federate large computing and storage resources for dynamic analysis of large datasets. Building on the Globus Toolkit 4, we have built an AstroPortal prototype and implemented a first analysis, "stacking," that sums multiple regions of the sky, a function that can help both identify variable sources and detect faint objects. We have deployed AstroPortal on the TeraGrid distributed infrastructure and applied the stacking function to the Sloan Digital Sky Survey (SDSS), DR4, which comprises about 300 million objects dispersed over 1.3 million files, a total of 3 terabytes of compressed data, with promising results. AstroPortal gives the astronomy community a new tool to advance their research and to open new doors to opportunities never before possible on such a large scale.

Keywords: *AstroPortal, web portal, grid computing, astronomy, stacking, SDSS*

1 Introduction

The astronomy community is acquiring an abundance of digital imaging data, via sky surveys such as SDSS [2], GSC-II [3], 2MASS [4], and POSS-II [5]. However, these datasets are generally large (multiple terabytes) and contain many objects (100 million +) separated into many files (1 million +). Thus, while it is by now common for astronomers to use Web Services interfaces to retrieve individual objects, analyses that require access to significant fractions of a sky survey have proved difficult to implement efficiently. There are five reasons why such analyses are challenging: (1) *large dataset size*; (2) *large number of users* (1000s); (3) *large number of resources* needed for adequate performance (potentially 1000s of processors and 100s of TB of disk); (4) *dispersed geographic distribution of the users and resources*; and (5) *resource heterogeneity*.

We propose to use grid computing to enable the dynamic analysis of large astronomy datasets. The term "Grid" denotes a distributed computing infrastructure for advanced science and engineering. Grid is distinguished from conventional distributed computing by its focus on large-scale resource sharing, innovative applications, and high-performance orientation [1].

The key question we answer in this paper is: "*How can we leverage Grid resources to make the analysis of large astronomy datasets a reality for the astronomy community?*" Our answer is "AstroPortal," a gateway to grid resources tailored for the astronomy community. We have implemented our prototype as a Web Service using the Globus Toolkit 4 (GT4) [10] and deployed this service on TeraGrid [8]. The astronomy dataset we are using is the Sloan Digital Sky Survey (SDSS), DR4, which comprises about 300 million objects dispersed over 1.3 million files adding up to 3 terabytes of compressed data.

1.1 Stacking

The first analysis that we have implemented in our AstroPortal prototype is "stacking," image cutouts from different parts of the sky. This function can help to statistically detect objects too faint otherwise. Astronomical image collections usually cover an area of sky several times (in different wavebands, different times, etc). On the other hand, there are large differences in the sensitivities of different observations: objects detected in one band are often too faint to be seen in another survey. In such cases we still would like to see whether these objects can be detected, even in a statistical fashion. There has been a growing interest to re-project each image to a common set of pixel planes, then stacking images. The stacking improves the signal to noise, and after coadding a large number of images, there will be a detectable signal to measure the average brightness/shape etc of these objects. While this has been done for years manually for a small number of pointing fields, performing this task on wide areas of sky in a systematic way has not yet been done. It is also expected that the detection of much fainter sources (e.g., unusual objects such as transients) can be obtained from stacked images than can be detected in any individual image. AstroPortal gives the astronomy community a new tool to advance their research and opens doors to new opportunities.

1.2 AstroPortal

AstroPortal provides both a Web Services and a Web portal interface. Figure 1 is a screenshot of the AstroPortal Web Portal, which allows a user to request a "stacking" operation on an arbitrary set of objects from the SDSS DR4 dataset. The AstroPortal Web Portal is implemented using Java Servlets and Java Server Pages technologies; we used Tomcat 4.1.31 as the container for our web portal.

User input comprises (1) user ID and password, (2) a stacking description, and (3) the AstroPortal Service location. The user ID and password are currently created out-of-band; in the future, we will investigate alternatives to making this a relatively automated process [17]. The stacking description is a list of objects identified by the tuple {ra dec band}. The AstroPortal Web Service location is currently statically defined in the web portal interface, but in the future we envision a more dynamic discovery mechanism.

Following submission (see Figure 1), the user gets a status screen showing the progress of the stacking, including percentage completed and an estimated completion time. Once the stacking is complete, the results are returned along with additional information about performance and any errors encountered. Figure 2 shows an example result from the stacking of 20 objects. The results include (1) summary, (2) results, and (3) statistics and errors.

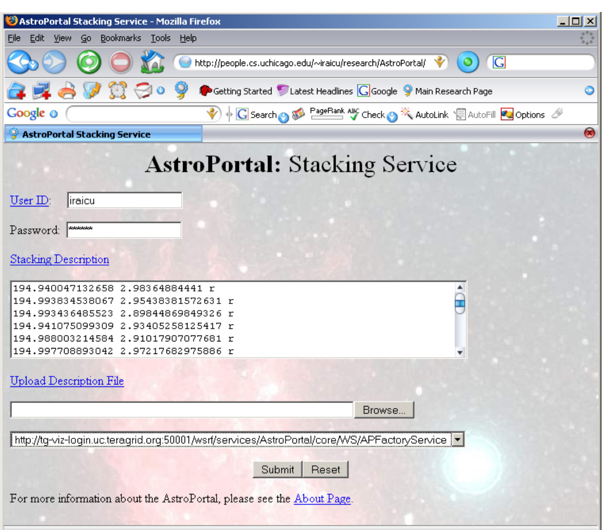

Figure 1: AstroPortal Web Portal Stacking Service

The summary includes all information entered on the submission page: (1) user ID and password, (2) a stacking description (along with the size of the stacking), and (3) the AstroPortal service location.

The results displays a JPEG equivalent of the result for quick interpretation, along with the size of the result (in KB), the physical dimensions of the result (in pixels x pixels), and a link to the result in FIT format [18].

The final section specifies the completion time, number of computers used, number of objects found, the number (and address) of star objects not found in the SDSS dataset, and the number (and address) of data objects not found in the data cache. Some star objects might not be found in SDSS since the SDSS dataset does not cover the entire sky; other objects might not be found in the data cache due to inconsistencies (e.g., read permission denied, corrupt data, data cache inaccessible) between the original data archive and the live data cache actually used in the stacking.

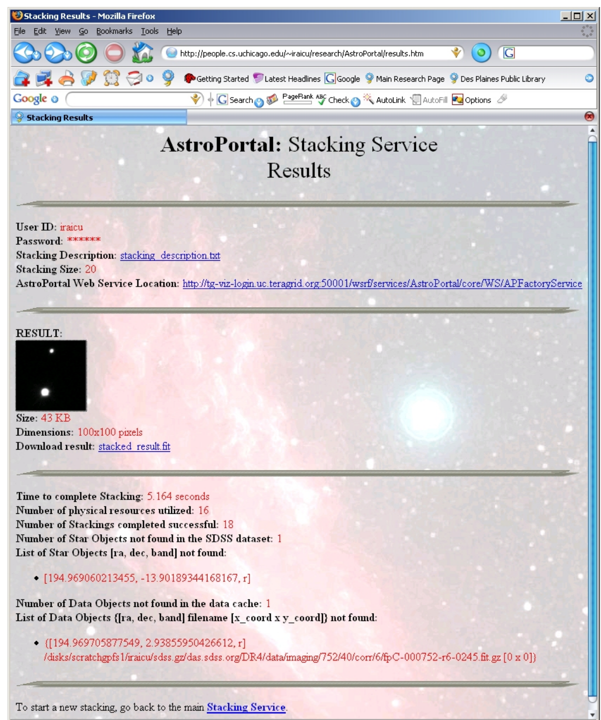

Figure 2: Stacking Service Result

2 Background Information

We describe related work similar to AstroPortal and cover background material (existing astronomy datasets, the TeraGrid testbed, and science gateways) necessary to make this paper self contained.

2.1 Astronomy Datasets

Astronomy faces a data avalanche, with breakthroughs in telescope, detector, and computer technology allowing astronomical surveys to produce terabytes

(TB) of images. Well-known large astronomy datasets that could potentially be used by AstroPortal include the Sloan Digital Sky Survey (SDSS) [2], the Guide Star Catalog II (GSC-II) [3], the Two Micron All Sky Survey (2MASS) [4], and the Palomar Observatory Sky Survey (POSS-II) [5]. Such astronomy datasets are generally large (TB+) and contain many objects (>200M). For example, SDSS has 300M objects in 10TB of data; GSC-II has 1000M objects with 8TB of data; 2MASS has 500M objects in 10TB of data; and POSS-II has 1000M objects in 3TB of data.

2.2 Related Work

Other work is underway to apply TeraGrid resources to explore and analyze the large datasets being federated within the NSF National Virtual Observatory (NVO) [6, 15].. For example, Montage [7] is a portable, compute-intensive, custom astronomical image mosaic service that, like AstroPortal, makes extensive use of grid computing. Our work is distinguished by its focus on the "stacking" problem and by its architecture that allows for flexible and dynamic provisioning of data and computing resources.

2.3 TeraGrid & Science Gateways

TeraGrid [8] is an open scientific discovery infrastructure combining leadership class resources at eight partner sites to create an integrated, persistent computational resource. TeraGrid provides over 40 teraflops of computing power and nearly 2 petabytes of rotating storage, interconnected at 10-30 gigabits/second via a dedicated national network. The initial prototype is deployed at the University of Chicago (UC) site in the TeraGrid. We will deploy future implementation iterations over the entire TeraGrid.

AstroPortal is an example of what the TeraGrid community calls a Science Gateway [9]: an user-oriented problem solving system that makes use of TeraGrid resources to deliver the value of high-performance computing to a large community. Science Gateways signal a paradigm shift from traditional high performance computing use.

3 AstroPortal Architecture

The AstroPortal implementation uses components of the Globus Toolkit version 4 (GT4) [10], and has been deployed in TeraGrid [8]. GT4 components used include WS GRAM [10], GridFTP [11], and WS Core [10]. Our current implementation's focus has been on the AstroPortal functionality as a science gateway, along with the basic resource management functions required for astronomy analysis codes to be run efficiently on large datasets. In the rest of this section, we focus on our use of the SDSS DR4 [12] dataset as the first supported dataset in our prototype deployed on the ANL/UC TeraGrid system.

3.1 Architecture

As shown in Figure 3, AstroPortal (AP) includes (1) the AstroPortal Web Service (APWS), (3) the Astro Workers (AW) running on the compute nodes, and (4) the Astro Users (AU). The communication between all these components is done using Web Services (WS). Furthermore, we have leveraged GT4 functionality which offers persistent state storage for Web Services; the persistent state makes the APWS more robust to failures as it allows us to continue execution of unfinished jobs after a system restart, bringing the AstroPortal implementation a step closer to being a production ready service.

The APWS is the main component of the system where the resource management innovation needs to occur to extract the best performance possible from the TG resources; an additional component could be a data manager which would manage the data placement, replication, synchronization, and expose an interface to locate the data with the least expensive access method. Both of these components are rather generic, and with minor tuning, could be used in the analysis of other large non-astronomy related datasets. The AW and AU are specific to the astronomy community, and will offer the analysis and visualization functionality needed make the AstroPortal system useful to astronomers.

APWS is the centralized gateway to which all AUs submit their analysis requests. Once the APWS has started, it could register itself with a well-known MDS4 Index, so that the AU can dynamically find the location of the APWS; however, in the current implementation, the AU must discover the APWS via an out-of-band mechanism. A AU can use any of many existing tools offered by the SDSS/SkyServer [13] to find the location (i.e., sky coordinates – {ra dec band}) of the objects of interest. The AU then sends the list of locations along with the analysis to be performed to the APWS for processing as a job.

The APWS is responsible for dispatching each incoming request to one or more worker resources, which were created via the GRAM API. The number of workers can be varied over time, increasing under heavy loads and decreasing when load reduces. Astro Workers (AW) register with the APWS, and it is the APWS responsibility to notify AW of new work that needs to be completed. Upon the APWS receiving the work from the AU, it finds the necessary data that will need to be accessed to perform the analysis and it notifies the AW that work is available. When the APWS receives the results from an entire job (it could have been fragmented into smaller pieces, with each small piece done independently by multiple concurrent AW), it packages or aggregates them (depending on the analysis) and sends the results back to the AU. Stacking operations normally produce relatively small results, but for larger results, we can return just their location, leaving the actual results to be retrieved via GridFTP, thus avoiding XML processing costs.

AstroPortal's internal design is depicted in Figure 4. The AstroPortal Factory maintains a list of workers that have registered as available for work. (Workers keep their registrations valid by periodic updates.)

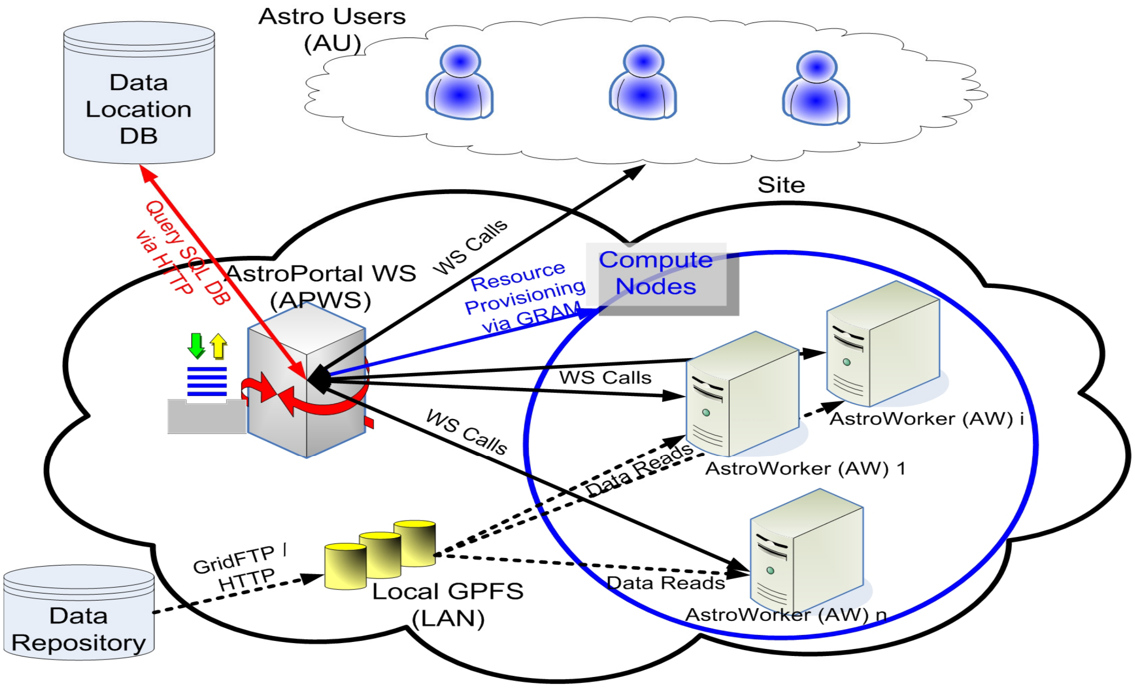

Figure 3: AstroPortal architecture

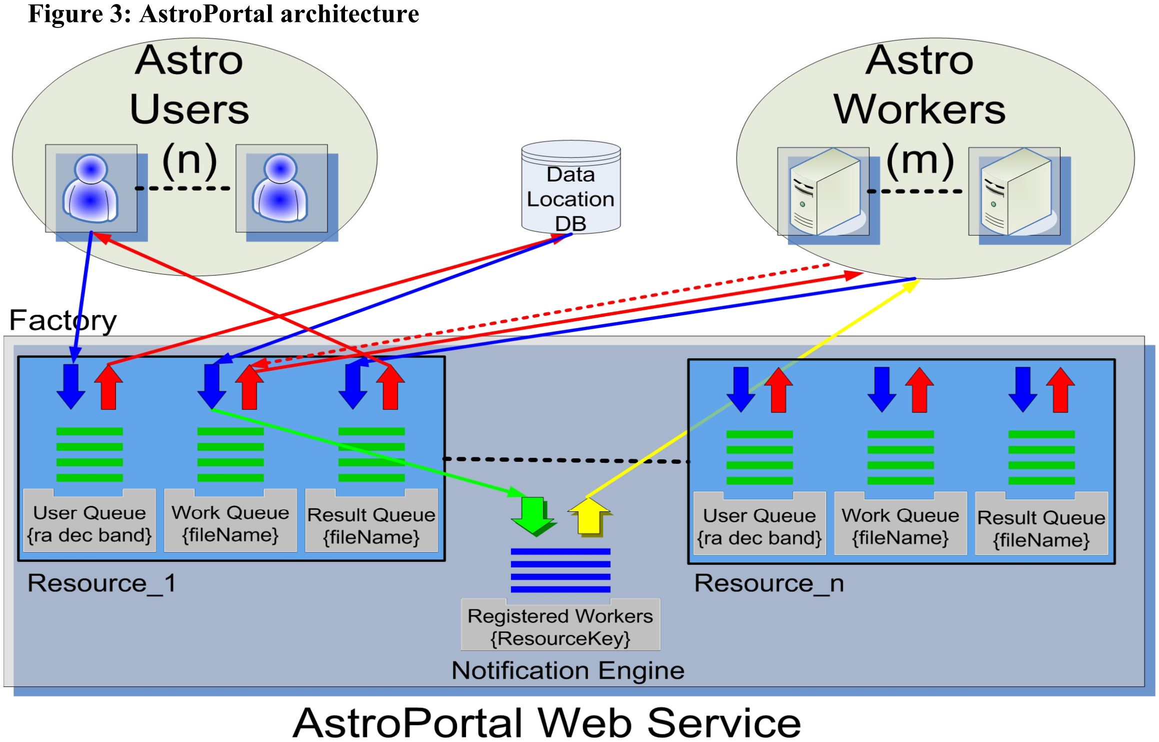

Figure 4: Internal AstroPortal Web Service Overview

Every user creates a WS-Resource before it submits the stacking operation; this design allows for easy client monitoring and management of the submitted work, as each task is represented by a unique WS-Addressing Endpoint Reference (EPR).

Once a user has created a resource to use for the "stacking" jobs, it submits the stacking description to the resource. The APWS places the list of tuples {ra dec band} into the User Queue. A translation thread takes elements from the User Queue and contacts a database to get the data location where the data needed to access the object in question can be found, and the result is placed into the Work Queue. When there is work available in the Work Queue, the Notification Engine sends a notification to some workers specifying which specific Resource has work to get done. Once a worker receives a notification, the interaction is all between the worker and the particular resource that had the work. Once the worker completed an assigned stacking, it sends the result back to the corresponding resource, and the result is inserted into the Result Queue. Meanwhile, the user polls the corresponding resource for the progress of the stacking; when the stacking is complete, the user invokes a WS call to retrieve the result.

3.2 Preliminary Performance Evaluation

We conducted a brief study of AstroPortal performance. The experiment consisted of a single user using the AstroPortal and all its resources. We performed experiments while varying (1) the number of images from 1 to 4096, and (2) the number of workers (Dual Xeon 2.4 GHz CPUs with 4GB RAM) from 1 to 48. Data was accessed on the GPFS parallel file system at Argonne National Laboratory over a LAN (Gbit/s connectivity) in GZIP compressed format. Each stacking was done over a 100x100 sub-image from a 2048x1489 16-bit image that was approximately 2.3 MB in compressed format.

The workers were implemented in Java and rely on a JAVA FITS library [16] that provides efficient I/O for FITS images and binary tables. This library supports all basic FITS formats and gzip compressed files.

Figure 5 shows our performance results. AstroPortal and the Argonne GPFS scale well up to 32 workers, but performance decreases slightly as we increase the number of workers to 48. We tentatively blame the GPFS for this performance limit. Each worker runs multiple parallel stacking threads that perform parallel reads from the SDSS dataset; in our experiments, we had each worker run 10 threads in parallel in order to utilize the local computational resource fully. With 32 concurrent workers and each worker running 10 parallel reads, we have 320 parallel reads from the ANL GPFS, which only has 8 servers running in the back-end. Further tests need to be done before small differences (such as those observed between 32 workers and 48 workers) become statistically significant, especially as the ANL GPFS is a shared file system and could see transient loads generated by other users and services.

We see that AstroPortal is most efficient with large number of stackings, when it can take the most advantage of the resulting massive parallelization. For example, we can perform 1024 stackings in under 30 seconds with 32 workers, considering the fact that the 1024 objects needed to be accessed were contained in 1024 files summing to almost 2.4 GB of compressed data. Based on the results shown in Figure 5, using 32 workers, we could perform a stacking of 1.3 million objects that would touch (reading of 100x100 sub-image from every compressed image) the entire SDSS dataset (1.3 million data files and about 3TB of compressed data) in just over 10 hours. Unfortunately, stacking(s) of 300 million objects (all objects from SDSS) would take about 100 days, a relatively large time period. We expect to see significant improvements in performance on the decompressed dataset. It is worthwhile to mention that the stacking operations are most likely to be I/O bound as many small reads must occur for a single stacking to be completed successfully. Having data that is locally

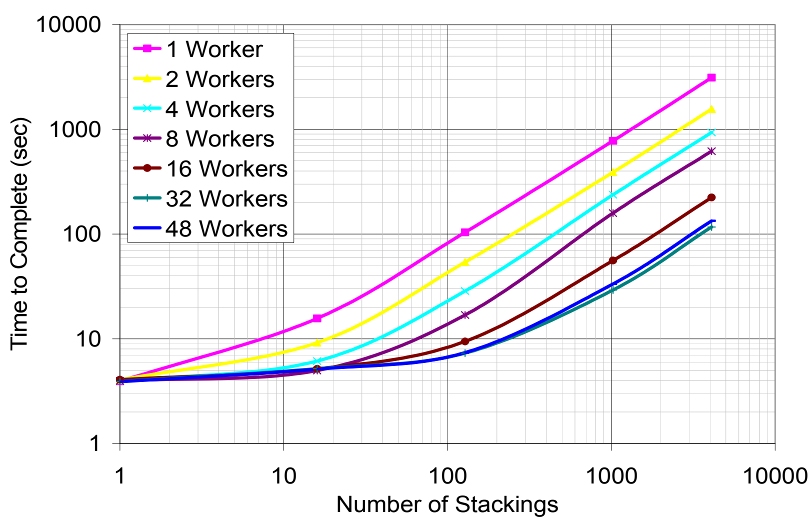

Figure 5: Stacking Performance on compressed images on the ANL

cached at the workers could alleviate the GPFS back-end servers from the large number of concurrent I/O calls, and better utilizing the available disk read performance. We intend to pursue better data access and management techniques in order to get better utilization of the raw available hardware, and at the same time, better performance with faster response times for the users.

Performance with fewer stackings is relatively poorer, presumably due to (inter and intra process) communication among the various AstroPortal components, including the initialization of needed resources (e.g., initializing local state and queue creation, thread creation). One way to improve the performance of small number of stackings is to perform small stackings locally within AstroPortal Web Service, without dispatching it to other resources.

Further analysis is needed to evaluate AstroPortal's ability to handle concurrent users, memory consumption (per user), and robustness in the face of concurrency and failure. We will conduct our extended performance evaluation via DiPerF [14], a DIstributed PERformance testing Framework, that simplifies and automates service performance evaluation. DiPerF coordinates a pool of machines that test a single or distributed target service, collects and aggregates performance metrics from the client point of view, and generates performance statistics (we will collect similar performance metrics at AstroPortal for validation of the obtained results). The aggregate data collected provides information on service throughput, service response time, on service 'fairness' when serving multiple clients concurrently, and on the impact of network latency on service performance.

4 Conclusions and Future Work

The key question we have addressed by the implementation of AstroPortal is: "*How can we leverage Grid resources to make the analysis of large astronomy datasets a reality for the astronomy community?*" AstroPortal is a science gateway to grid resources, tailored for the astronomy community. It gives the astronomy community a new tool to advance their research and opens doors to previously inaccessible opportunities. As the astronomy community uses AstroPortal, we will evolve its design and implementation to optimize it for the particular workloads and access patterns observed.

We see three areas with open research problems that the AstroPortal architecture exposes, and could offer contributions in. These areas, all in the broad context of resource management, include: (1) resource provisioning (advanced reservations, resource allocation, de-allocation, and migration); (2) data management (data location, data caching, and data replication); and (3) distributed resource management. In addition, we hope that our work will identify new resource and data management approaches for the efficient and successful dynamic analysis of large scientific datasets in other fields of astronomy and science.

References

[1] I Foster, C Kesselman, S Tuecke, "*The Anatomy of the Grid*", Intl. Supercomputing Applications, 2001.

[2] SDSS: Sloan Digital Sky Survey, http://www.sdss.org/

[3] GSC-II: Guide Star Catalog II, http://www-gsss.stsci.edu/gsc/GSChome.htm

[4] 2MASS: Two Micron All Sky Survey, http://irsa.ipac.caltech.edu/Missions/2mass.html

[5] POSS-II: Palomar Observatory Sky Survey, http://taltos.pha.jhu.edu/~rrg/science/dposs/dposs.html

[6] R Williams, A Connolly, J Gardner. "*The NSF National Virtual Observatory, TeraGrid Utilization Proposal to NRAC*", http://www.us-vo.org/pubs/files/teragrid-nvo-final.pdf

[7] JC Jacob, DS Katz, T Prince, GB Berriman, JC Good, AC Laity, E Deelman, G Singh, MH Su. "*The Montage Architecture for Grid-Enabled Science Processing of Large, Distributed Datasets*", Earth Science Technology Conference, 2004.

[8] TeraGrid, http://www.teragrid.org/

[9] TeraGrid: Science Gateways, http://www.teragrid.org/programs/sci_gateways/

[10] I Foster. "*A Globus Toolkit Primer*," 02/24/2005. http://www-unix.globus.org/toolkit/docs/development/3.9.5/key/GT4_Primer_0.6.pdf

[11] B Allcock, J Bresnahan, R Kettimuthu, M Link, C Dumitrescu, I Raicu, I Foster. "*The Globus Striped GridFTP Framework and Server*", IEEE/ACM SC 2005.

[12] SDSS Data Release 4 (DR4), http://www.sdss.org/dr4/

[13] Sloan Digital Sky Survey / SkyServer, http://cas.sdss.org/astro/en/

[14] I Raicu, C Dumitrescu, M Ripeanu, I Foster. "*The Design, Performance, and Use of DiPerF: An automated DIstributed PERformance testing Framework*", to appear in the Journal of Grid Computing, Special Issue on Global and Peer-to-Peer Computing 2006.

[15] US National Virtual Observatory (NVO), http://www.us-vo.org/index.cfm

[16] JAVA FITS library: http://heasarc.gsfc.nasa.gov/docs/heasarc/fits/java/v0.9/

[17] I. Foster, V. Nefedova, L. Liming, R. Ananthakrishnan, R. Madduri, L. Pearlman, O. Mulmo, M. Ahsant. "Streamlining Grid Operations: Definition and Deployment of a Portal-based User Registration Service." Workshop on Grid Applications: from Early Adopters to Mainstream Users 2005.

[18] Hanisch, R. J.; Farris, A.; Greisen, E. W.; Pence, W. D.; Schlesinger, B. M.; Teuben, P. J.; Thompson, R. W.; Warnock, A., III. Definition of the Flexible Image Transport System (FITS) Astronomy and Astrophysics, v.376, p.359-380, September 2001.