An Open Job Scheduling Service for Large-Scale Data Processing

Zhenghua Xue, Jianhui Li, Yuanchun Zhou, Yang Zhang, Geng Shen
Computer Network Information Center
Chinese Academy of Sciences
Beijing, China, 100190
{zhxue, lijh, zyc, zhangyang, shengeng}@cnic.cn

Abstract—**Scientific Data Grid (SDG) of Chinese Academy of Sciences aims at integrating distributed scientific data, providing transparent data access mechanism and efficient data analysis，processing and visualization services. SDG Job Scheduler (SDGJS) adopts an open and service-oriented framework. The scheduling policy of SDGJS considers both performance of computing nodes and distribution of large data so that it can achieve an efficient job processing. SDGJS has successfully served for some data grid applications.**

Keywords-Large-Scale Data; Grid; Job Schedule;

I. INTRODUCTION

With continuous expansion of scientific data and the enhancement of collaborative research activities, the traditional service architecture providing data sharing with stand-alone mode can't meet the needs of researchers. Applications for scientific data are facing new challenges. These challenges are highlighted in the following areas: large and valuable scientific data need stable, reliable storage management mechanism; geographically distributed, heterogeneous scientific data need facilitated and unified access mechanism; computing and analysis for data-intensive applications require support of powerful computing resources; data, computing, science-oriented knowledge and visualized analysis tools require appropriate integration. To address these challenges, Chinese Academy of Sciences (CAS) launched the Scientific Data Grid project, which established a data grid group on the basis of scientific database of CAS. In order to provide transparent data access, SDG integrated the distributed scientific data and formed a unified logic view of data resources. Based on the scientific data and advanced computing and storage facilities, SDG can provide convenient application services for analysis, processing and visualization of scientific data and forms the framework for the development of scientific data applications. As the first phase of establishment for data grid, SDG has formed a grid operation center and four sub-center for four disciplines, namely, Chemical Data Grid, Space Data Grid, Microbiological and Viruological Data Grid and Geographical Data Grid. Currently, these projects have been initially completed and are able to provide data, middleware, grid application and other services.

Grid concept was originally put forward by Ian Foster [1-3] and other scholars. The goal of grid contains two aspects: firstly, grid aggregates distributed resources to form a supercomputing capacity to meet the needs of large-scale computing applications; secondly, grid shares the heterogeneous network resources to make full use of various resources. According to the focused technical points and different application scenarios, grid can be divided into computing grid and data grid. Computing grid focus on the management and deployment of computing resources and jobs, but data grid focus on data and storage resource management and aims at efficient data storage solution, backup, transfer and sharing issues. There are no strict distinctions between computing grid and data grid. From the application perspective, data analysis and processing relies on computation, while the computation tends to produce large amounts of data. U.S. Open Science Grid (OSG), Tera Grid, and the EU's European grid (Euro Grid) focus on computing. American physical grid (GriPhyN) [4] and the Euro Data Grid are typical data grid [5-7]. Grid technology has also been in a rapid development in China. Some representative grid projects include: China National Grid (CNGrid) supported by Chinese Ministry of Science and the ChinaGrid supported by Chinese Ministry of Education. These grid projects promoted the development of grid technology, and their achievements have also been widely used in education, research, national defense and other fields.

During establishing scientific data grid, a key challenge is how to aggregate distributed computing resources to support for computation related to data processing and analysis. For different computing tasks, scientific data grid requires an efficient job management system to perform data analysis and processing on the distributed computing nodes. To meet these challenges, we developed scientific data management system (SDGJS) to provide support for the job scheduling of SDG.

II. SYSTEM ARCHITECTURE

A. Architecture of Scientific Data Grid

Figure 1 shows the overall architecture of scientific data grid. The infrastructure services mainly include storage and computing services. Through scientific data grid middleware, these infrastructures provide authorized users with resource calling service. In addition, large amount of scientific data is also one of the important services provided by SDG. Data services provide scientific data management service through

This work was supported by National Natural Science Foundation of China (No. 61003138) and Knowledge Innovation Program of Chinese Academy of Sciences (No. CNIC_QN_11005, CNIC_ZR_201004, CNIC_QN_10006).

data management middleware, such as metadata management, data transmission, data search and data monitoring middleware. SDG also provides public services related to infrastructure environment and data services, such as resource registration services, user management services, grid organization management, operation and maintenance management services. These infrastructures, scientific data, middleware, and other hardware and software together constitute the scientific data grid system.

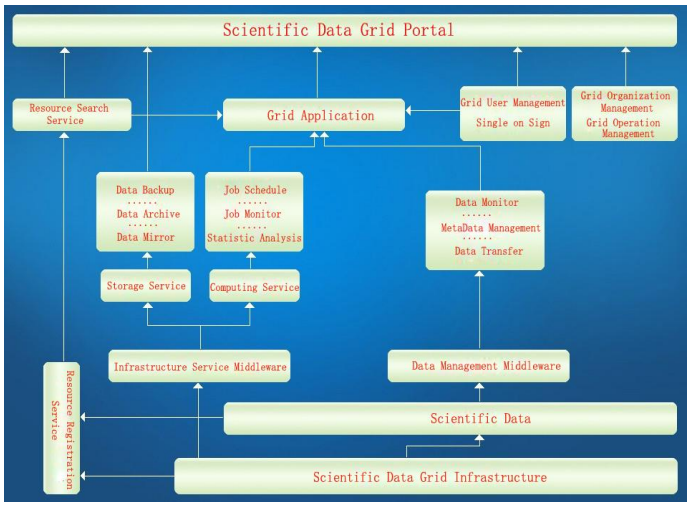

Figure 1. Overall Architecture of Scientific Data Grid in CAS

B. *Computing Service Process*

The ultimate goal of scientific data grid is to provide convenient, efficient and reliable data analysis and processing services for scientific researchers. For scientific researchers in subjects, scientific data grid provides infrastructure services for quickly building application service platform, which facilitates the users to develop Science Gateway based on SDG's infrastructure. Based on SDGJS, Fig. 2 shows the process of infrastructure services for discipline-oriented application.

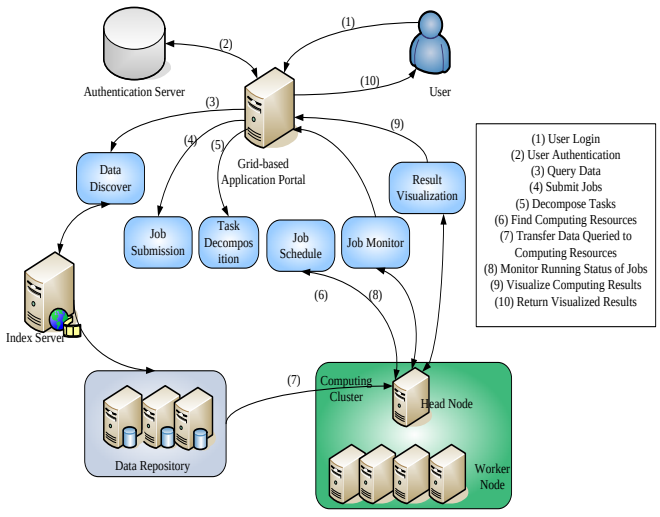

Figure 2. Infrastructure Service Process of Scientific Data Grid in CAS

C. *SDGJS Architecture*

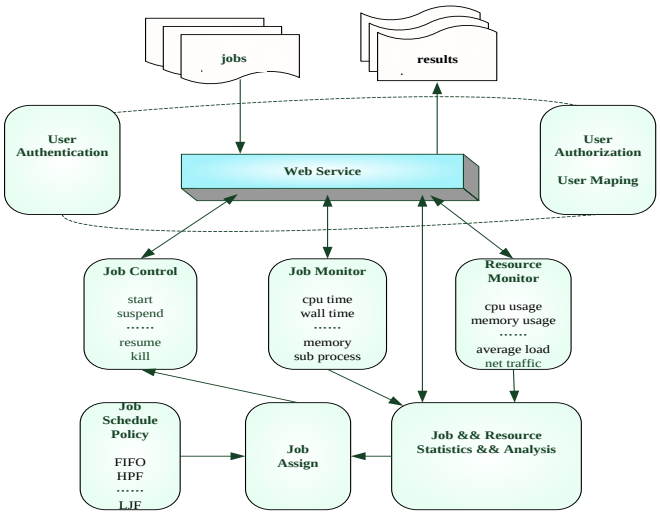

Figure 3. Architecture of SDGJS

Fig. 3 shows the architecture of the job scheduling of SDG. Job controller module is used for the jobs to start, suspend, resume, delete and other operations; job monitoring module provides real-time monitoring for jobs, including job run time, CPU time, memory and the data used; resource monitoring module is responsible for monitoring the CPU utilization, memory usage, network bandwidth, load balance and other information of the data processing system; the module of statistical analysis for job and resource is used for statistical analysis of historical jobs and resource usage. The analysis results are called by job allocation module to determine job allocation. In addition, this module also provides web services for grid user and third-party grid component, which is used to decide whether to submit jobs to a grid site, or to provide unified monitoring and statistical analysis services for the nodes of the entire scientific data grid. In order to conveniently control the executive priority of jobs, it is permissible for administrator of a grid site to modify or add job scheduling strategy.

III. KEY TECHNOLOGY

A. *User Management*

In order to facilitate user management, SDG of CAS adopts unified user access authentication system. The system is single sign on, which allows users to log on the scientific data grid system only once and then to be exempt from the second login in all of the mutual trusted system. To protect the owners' permission for controlling their data resources, storage or computing resources in SDG, SDG adopts the implementation with a separation of authentication and authorization. The unified authentication server only contains the user information, and the owner of system resources or applications determine the users' specific permission. SDGJS's user management module leverages the user mapping method to map the authorized user from authentication server to legitimate users of system resources. For system resources on each grid site, there is a unified authorization node which acts as the

authorization server. In order to achieve centralized control of the authorization behavior, each of the computing and storage nodes acts as a client of the authorization system shared by all computing and storage nodes. This mode facilitates the user management of system and only authorizes users on the authentication server. Therefore, other nodes do not need to focus on specific authentication operation.

B. Computing Resource Monitoring

To achieve load balancing and high availability, SDGJS need to choose the appropriate computing nodes for the assigned tasks according to resource utilization of computing system. In addition, some computing tasks have special needs for computing nodes on processing ability, available memory and disk space, etc. These tasks need to monitor performance and status information of computing system to determine whether computing nodes meet the job's needs. In SDGJS, there is a head node which is responsible to respond to external requests and perform management, performance monitor and task assignment of all other nodes. Most computing nodes in SDG use Linux system, "proc" file system of which is resolved by resource monitoring module to obtain the performance information. As a web service interface, the method accessing performance information is called by external procedures.

C. Job Monitoring

Job monitoring items of SDGJS include: wall time, CPU time, memory required during job running and other information. Once a job is finished, the information will be stored into database to facilitate subsequent statistical analysis, accounting and other operation processes. Since any job can be considered as a process of operating system, SDGJS monitors jobs by using the process monitoring technology. Process is monitored when started, and its sub-processes and the system resources consumed by sub-processes are also monitored. In addition, when exceptions occur in processes or sub-processes, the system resources consumed during running and other information can be stored before the processes exit. Job monitoring module achieves the process monitoring through calling various API provided by operating system.

Unlike the existing job scheduling systems, SDGJS monitors not only job run time, consumed system resources, but also data resources used by jobs. This monitoring information is preserved for statistical analysis to find out what data is hot and what data should be retained with multiple copies in multiple grid sites. In addition, these statistics also help to analyze what data users have preferences, and they can assist job scheduling systems to make better schedule. As a result of the Peer to Peer mode for each grid site, the file system of each site can only find the hot data within the site but fail to perform statistical analysis for the usage of global sites. In addition, the file system can't find the association between users and data, which limits job scheduling system to make a better schedule. During each job submission stage, SDGJS can easily get this information to achieve inter-association among a series of factors, such as users, data and computing needs. This association knowledge helps job scheduling system to make better scheduling.

D. Job Scheduling

In SDG, grid application developers and service providers need firstly to install and deploy the software required by application into the grid nodes. After that, they maintain a list of available grid nodes in the Web server. Job scheduling system will firstly look for nodes which can provide grid application service in this list, and then select the appropriate grid node according to relevant constraints and resource selection algorithm.

SDG mainly focuses on grid application with data-intensive feature. In addition to have a greater demand for computing resources, these applications also have high requirements for data distribution and access methods. Compared to traditional high performance computing, the scheduling mechanism of these applications is more complex. In order to improve the performance of data-intensive computing applications, SDGJS need to consider not only the performance status of computing resources and also data distribution and fast data loading, etc. Currently, most applications deployed in scientific data grid require diverse, multi-source data. According to the amount of data required, SDGJS provides two ways of data loading. SDGJS provides efficient transportation mechanism to achieve real-time access for small data and pre-deployment into grid sites to ensure application performance for large data.

Data-intensive computing applications tend to have high requirements for I/O, while large amount of data transmitted on network will incur performance degradation and system instability. To solve this issue, large data processing system of the operation center of scientific data grid adopts the system architecture with computation and storage tightly coupled, i.e., each computing node acts as storage node. This system architecture greatly improves the "locality" of data. Computing on the data storage, without moving the data, effectively avoid moving a large amount of data on the network. For the grid application based on large data processing, the core idea of SDGJS scheduling is that application moves to large data. Based on this idea, SDGJS designs the monitoring and statistical analysis module which performs statistical analysis for status of the requested data and classification for use frequency of data. Different levels of data corresponding to different amounts of copies. To improve application performance and achieve system load balancing, when access frequency of some data is high, this module will call the relevant interfaces provided by file system to generate new data copies in other nodes. After that, the jobs which access data will be allocated to different processing nodes. When multiple nodes simultaneously have the data required by a job, the resource monitoring module of SDGJS will firstly analyze the load status of those effective nodes, and then assign the job to the nodes with the lower load to improve the response time. For the priority selection of jobs, SDGJS supports custom strategy and some common job scheduling strategies, such as first come first service, and priority to small jobs, etc.

E. Data Transport

Though SDGJS take great efforts to avoid transport data from one site to another, it's inevitable to do that in some occasions. To improve data transport performance, SDGJS

adopt two data transport middleware for massive data transport and real-time data transport which usually are not massive.

SDG is an open system based on SOA, grid applications can be developed by any development languages. To facilitate real-time transportation, we develop a middleware based on Hession2 protocol to support invocation by various languages, including Java, Flash, Flex, Python, C++, C#, Erlang, PHP and Ruby. Hessian2 is upgrade of Hessian, and it is a dynamically-typed, binary serialization and Web Services protocol designed for object-oriented transmission. It supports encryption, compression, signature, and transaction context envelopes. We performed performance measurement on transportation list which encapsulates objects, and compared with other protocols. The results are shown in Fig. 4 and Fig. 5. It prevails over other protocols.

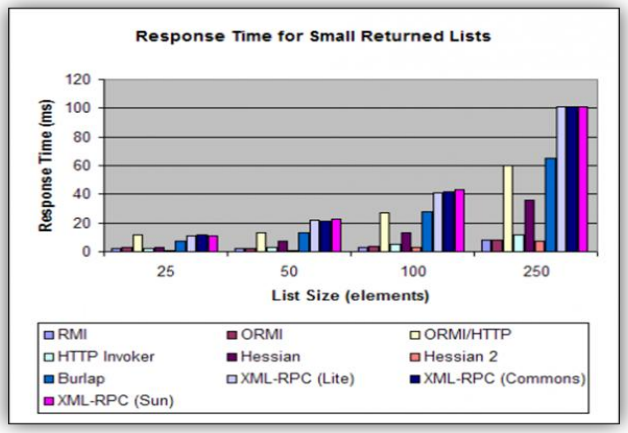

Figure 4. Performance Comparison for Small Lists

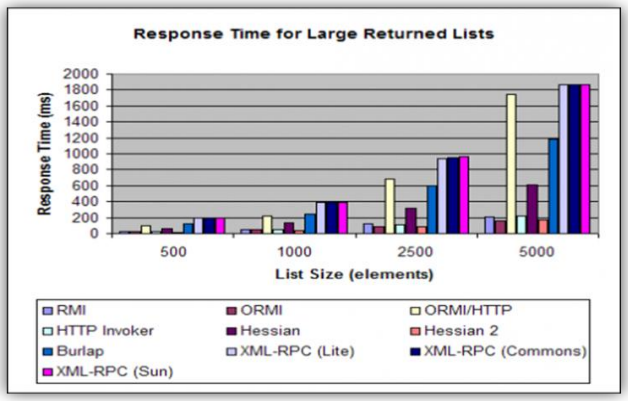

Figure 5. Performance Comparison for Large Lists

For large files which usually are over 1GB, we adopted Globus GridFTP as transport middleware. It is a robust suite designed to move large amounts of data faster, more securely, and more reliably than standard FTP. It has strong authentication, encryption via Globus GSI. It supports parallel transport streams, reliable and resuming transfers. It provides two parameters for performance improvements: *p* (parallel streams) and *tcp-bs* (TCP buffer size). We conducted measurements in different SDG sites which are bridged with 1Gbps bandwidth. The results show that GridFTP is very efficient for transferring large volume of data.

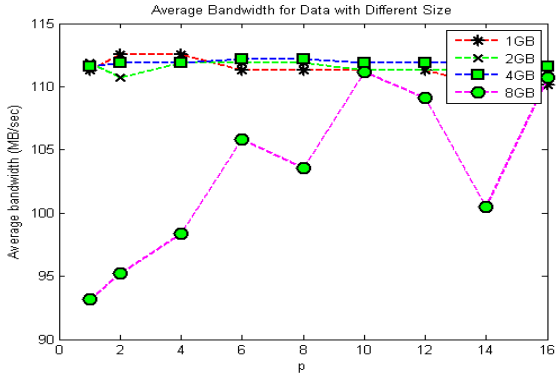

Figure 6. Transport Performance with Various Parallel Streams

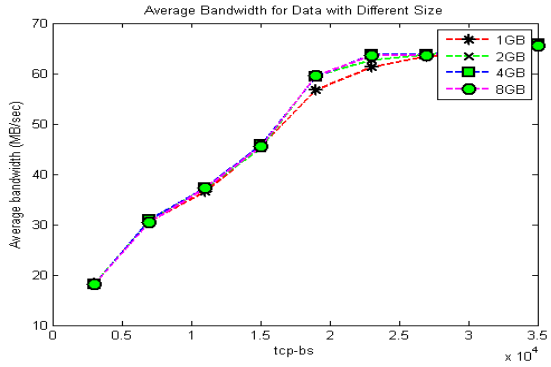

Figure 7. Transport Performance with Various TCP Buffer Sizes

IV. APPLICATION CASES

A. *High preision surface presentation grid application*

As a grid application, this application was developed by virtual organization Geographical Scientific Data Grid. Part of the vector data required by this application comes from Institute of Geography in Beijing, Institute of Water Conservation in Xi'an, Institute of Geography in Harbin, and Institute of Mountain-Land in Chengdu. These data are not large, and can be accessed real-time from database of grid sites by way of calling the data transportation middleware of SDG. Another part of the data, Digital Elevation Model (DEM), is larger and has been pre-deployed to the data processing environment of the operation center and Institute of Geography. By analyzing the use frequency of DEM data, SDGJS generates multiple copies of hot area data of DEM. By reading metadata information from file system and considering the load of computing nodes, SDGJS is able to choose the best suitable computing nodes to deal with tasks. The application provides on-line service at Geographical SDG site [8].

B. *Computing platform of SDG for Biological Sciences*

As a biology-oriented computing platform, this platform was developed by virtual organization Microbiological and

Viruological Data Grid which is one of the subject grid of scientific data grid. Data required by this application comes from Institute of Microbiology, Institute of Virus in Wuhan, and mirror of biological data from other international organizations. We have deployed hundreds of biological computation and analysis programs to the data processing environment of operation center and Microbiology grid site. SDGJS provides some web services, such as job submission, job monitoring, job files uploading and downloading, etc. Biological computing platform integrates these services to their platform, and then calls these services to perform computing. The computing platform presents a unified entrance for user job submission, monitoring, accessing result and displaying. This grid application provides researchers in Biology with a unified biological computing and analysis platform which integrates data, computing model and computing resources. It provides service at Microbiological and Viruological SDG site [9].

V. CONCLUSION AND FUTURE WORK

SDG of CAS aims at being the fundamental framework of the scientific data application environment. To facilitate use of scientific data services, SDG will integrate and organize data, computing models, computing resources to develop scientific data grid applications, and provide efficient, easy-using research platform for researchers. Different from existing job schedulers, SDGJS, as one of the important components of SDG, will provide open and simple interfaces for job scheduling. SDGJS comprehensively considers the performance load of computing resources and large data distribution. It adopts the "computing moves around data" idea and avoids large data transmitting over network, which not only helps to improve application performance, and also helps to enhance the stability of computing system.

REFERENCES

[1] Foster I, Kesselman C, Tuecke S. The Anatomy of the grid: Enabling Scalable Virtual Organizations [J]. International Journal of High Performance Computing Applications, 2001, 15(3):200-222.

[2] Foster I, Kesselman C, Nick J, Tuecke S. Grid Services for Distributed System Integration[J]. Computer, 2002, 35(6):37-46.

[3] Foster I. Service-Oriented Science [J]. Science, 2005, 308(5723): 814-817.

[4] Ewa Deelman, Carl Kesselman, Gaurang Mehta, etc. GriPhyN and LIGO, Building a Virtual Data Grid for Gravitational Wave Scientists [C]. Proceedings of the 11th IEEE International Symposium on High Performance Distributed Computing, 2002.

[5] Graeme A Stewart, David Cameron, Greig A Cowan, etc. Storage and data management in EGEE [C]. Proceedings of the fifth Australasian symposium on ACSW frontiers, 2007.

[6] Miguel Branco, Ed Zaluska, David Roure, Pedro Salgado, etc. Managing Very-Large Distributed Datasets [C]. Proceedings of the OTM 2008 Confederated International Conferences, 2008.

[7] Wääänänen A, Ellert M, Konstantinov A, etc. An Overview of an Architecture Proposal for a High Energy Physics Grid [C]. Proceedings of the 6th International Conference on Applied Parallel Computing Advanced Scientific Computing, 2002.

[8] Geographic SDG site, http://www.973geodata.cn:58080/modelapp/?ticket=ST-175-Aj5uJYA0XX6R002LAntd-cas.

[9] Biological SDG site, http://www.biogrid.cn/submit.