IJITKM Volume 7 • Number 1 • December 2013 pp. 121-124 (ISSN 0973-4414)

Data Management in Grid Computing

Ritu Baniwal
Department of Computer Sc. & Engg, GJUS&T Hisar
baniwalritu@gmail.com

Abstract- Grid computing is being used in various applications. Many grid applications such as weather forecasting, astrophysics deal with large amounts of data. Data intensive applications involve a high degree of data access locality. In Grid computing environments data storage management on large scale is one of the most challenging issues. Dedicated servers are used for data storage in traditional approaches.
Keywords- Grid, data management, MSS.

1. INTRODUCTION

A Grid is a collection of machines, sometimes referred to as nodes, resources, members, clients, hosts, engines, and many other terms [1]. Grid computing is being used in various applications. Many grid applications such as weather forecasting, astrophysics deal with large amounts of data. Data intensive applications involve a high degree of data access locality. This type of work is increasingly being performed in collaborative efforts between geographically distributed scientists and organizations, utilizing shared resources that are also widely distributed [1]. Network based storage systems such as Network Attached Storage (NAS) [7] and Storage Area Network (SAN) [8] offer an easy method to handle large amounts of storage.

2. DATA MANAGEMENT MODEL

In Grids, dedicated Metadata Services needed because of usability and ease of schema discovery. The metadata attributes could be represented in some database technology, such as relational or XML, and that the discovery of data objects be mapped into queries [2]. Databases provide a very general and flexible infrastructure for data management.
Metadata model supports a variety of interactions which allow users to describe attributes of data items and organize them in ways that are needed by collaboration and by individual users [2]. The model supports a set of attributes. The basic object is defined as a data item within a metadata service. The role of each entity is described as:

Users

There are three types of users: the collaboration, the members of the collaboration and the general public. The collaboration publishes the data sets. Individuals or groups within collaboration may provide additional attributes and annotation and may structure the metadata in a personalized way [2]. General public may make query about the metadata services.

Structuring Data Items

Refering a set of data items with a single name is very convenient e.g. authorization can be imposed on to that single object and there is no need to impose it on every individual item. Metadata model includes a concept of collections. Collections allow a name and attributes to be associated with an arbitrary set of data objects and these can aggregate a set of data items or other collections [2].

Support for Authorization

Database Access and Integration (DAI) supports authorization by mapping users to database roles [2]. The Distinguished Name (DN) from the certificate that the user presents for authentication identifies a user in Metadata Catalog Services (MCS) [2]. The user is mapped to an anonymous DN if the user does not use any authentication.

Support for a Attribute Set

To represent the metadata we need to define domain-independent attributes as well as additional attributes that can represent domain specific metadata. This can be achieved by creating a basic table that contains the common attributes as table columns and then create additional tables for a fixed set of predefined attribute types [2].

3. DATA MANAGEMENT IN GRID COMPUTING

3.1. TRADITIONAL DATA MANAGEMENT

Many data management systems developed to handle data e.g. hierarchical, network. Relational data management technology was the most successful data management system.

3.2. DYNAMIC AND SCALABLE STORAGE MANAGEMENT (DSSM)

The DSSM divides Grid storage devices into multiple geographically distributed to domains to facilitate the data access locality [9]. Data locality is a measure of how well data can be selected, retrieved, compactly stored, and reused for subsequent accesses [9]. There are two types of data locality: temporal and spatial. Temporal locality denotes that the data accessed at one point in time will be accessed in the near future. Spatial locality means the data which is near to the data which is recently accessed will be accessed in the near future.

The architecture consists of two levels: The bottom level adopts multicast to achieve dynamic, scalable, and self-organized physical domains [1]. This method is used to simplify the intra-domain storage management. The upper level selects geographically distributed and dynamic agents from each physical domain.

3.3. GRIDFTP

The applications to satisfy specific needs and requirements accessing large datasets include the Distributed Parallel Storage System (DPSS) and the High Performance Storage System (HPSS) [3]. These storage systems use incompatible protocols for accessing data. Each storage system requires the use of its own client. These protocols partition the datasets available on the Grid.

A common data transfer protocol GridFTP developed for all the customized storage systems. Establishing a common data transfer protocol would eliminate the current duplication of effort in developing unique data transfer capabilities for different storage systems [3]. GridFTP extends the standard FTP protocol. It includes a superset of the features offered by the various Grid storage systems. FTP is a well understood protocol used for data transfer on the internet and provides a well-defined architecture for protocol.

3.4. FILE AND OBJECT REPLICATION IN DATA GRIDS

In a Data Grid data replication is a key. A data replication tool, the Grid Data Management Pilot (GDMP) [4] was proposed. GDMP is a file replication software system. It was designed to replicate database files from one storage location to one or more other remote sites. A storage location can be a disk space on a single machine or it can be the space on several machines which are connected via a network and a network file system. Data replication is an optimization technique to achieve better access times to data. GDMP provides some preliminary storage management functionality and file replication services. GDMP uses: Replica Catalog Service, Storage Management Service for the storage management.

Replica Catalog Service

The GDMP replication service uses a Replica Catalog to maintain a global file name space of replicas. The Globus replica catalog keeps track of multiple physical copies of a single logical file by maintaining a mapping from logical file names to physical locations [4]. An end-user uses GDMP to publish information into the replica catalog.

Storage Management Service

The GDMP service uses external tools for staging files to make an interface with the Mass Storage Systems (MSS). For each type of Mass Storage System, tools for staging files to and from a local disk pool have to be provided [4] and each site has a disk pool that can be regarded as a data transfer cache for the Grid and that.A file staging facility is necessary when the disk space is limited and many users make request for the files concurrently. GDMP initializes the staging process if a remote site requests for a replica from another remote site and the file is not available in the disk pool. In the replica catalog, physical file locations are stored and contain file locations on disk. A file is first looked for on its disk location. If the file is not is not present in the disk location, it is searched in the Mass Storage System.

The GDMP replication service has been enhanced with more advanced data management features, including namespace and file catalog management, efficient file transfer, and preliminary storage management [4].

4. GASS: A DATA ACCESS SERVICE FOR WIDE AREA COMPUTING SYSTEMS

In wide area computing, programs are executed frequently at remote sites. Due to the data access mechanisms functionality demands on an application are limited which in turn lead to high-performance implementations. To address these requirements, Global Access to Secondary Storage GASS) was proposed. GASS is a data movement and remote access service. This service defines a global name space via Uniform Resource Locators and allows applications to access remote files via standard I/O interfaces [5]. High performance is achieved by incorporating default data movement strategies that are specialized for I/O patterns common in wide area applications and by providing support for programmer management of data movement [5].

a) Default Data Movement Strategies

For the efficient execution of grid applications we need to manage the available limited network bandwidth. In distributed file systems data movement operations are from the application programmer. The control is provided to focus on latency management rather than bandwidth management. GASS addresses bandwidth management issues by providing a file cache: a local secondary storage area in which copies of remote files can be stored [5].

b) Specialized Data Movement Strategies

These mechanisms fall into two general classes: relatively high-level mechanisms concerned with prestaging data into the cache prior to program access and with poststaging of data subsequent to program access; and low-level mechanisms that can be used to implement alternative data movement strategies [5].

Presaging and post staging commands provide the extension to the cache management mechanisms. These commands are used when the data-files are very large, or the files are used in multiple runs of a program.

Low-level cache mechanisms provide more control over cache behavior e.g. a user can specify when and where a file is cached. The first reason to use these mechanisms is that in distributed file system it may be impractical to place a large cached data in a common pre-allocated file area. A second reason is that GASS operations can exploit local distributed file systems e.g. cached files can be placed in a file system that can be accessed from more then one computational resource [5].

5. REMOTE I/O: FAST ACCESS TO DISTANT STORAGE [6]

Remote I/O can improve performance by overlapping computation and data transfer or by reducing communication requirements. Remote I/O introduces new challenges like portability, performance, and integration with distributed computing systems. RIO addresses issues of portability by adopting the quasi-standard Message Passing Interface (MPI)-IO interface and by defining a RIO device and RIO server I/O device architecture [6]. It addresses performance issues by providing asynchronous operations and by implementing buffering and message forwarding techniques to offload communication overheads. Remote I/O libraries allow access to data contained in remote filesystems.

CONCLUSION

Many grid applications such as weather forecasting, astrophysics deal with large amounts of data. In Grid computing environments data storage management on large scale is one of the most challenging issues. Various approaches have been followed for handling data in grid environments.

Dedicated servers are used for data storage in traditional approaches. The DSSM divides Grid storage devices into multiple geographically distributed to domains to facilitate the data access locality [9]. A common data transfer protocol GridFTP was developed for all the customized storage systems. GridFTP is secure and reliable transport mechanism. the Grid Data Management Pilot (GDMP) [4] was proposed as a data replication tool. The GDMP service uses external tools for staging files. A file staging facility is necessary when the disk space is limited and many users make request for the files concurrently. GASS a data movement and remote access service was proposed to handle remote accesses. GASS addresses bandwidth management issues by providing a file cache: a local secondary storage area in which copies of remote files can be stored [5]. Remote I/O can improve performance by overlapping computation and data transfer or by reducing communication requirements.

REFERENCES

[1] Ajay Kumar and Seema Bawa, "Distributed and big data storage management in grid computing"
[2] Ewa Deelman1, Gurmeet Singh, Malcolm P. Atkinson, Ann Chervenak, Neil P Chue Hong, Carl Kesselman, Sonal Patil, Laura Pearlman, Mei-Hui Su "Grid-Based Metadata Services",.
[3] Bill Allcock, Joe Bester, John Bresnahan, Ann L. Chervenak, Ian Foster, Carl Kesselman, Sam Meder, Veronika Nefedova, Darcy Quesnel, Steven Tuecke, "Data Management and Transfer in High-Performance Computational Grid Environments".
[4] Heinz Stockinger, Asad Samar, Bill Allcock, Ian Foster, Koen Holtman, Brian Tierney , "File and Object Replication in Data Grids".
[5] Joseph Bester, Ian Foster, Carl Kesselman, Jean Tedesco, Steven Tuecke, "GASS: A Data Movement and Access Service for Wide Area Computing Systems".
[6] Ian Foster, David Kohr, Rakesh Krishnaiyer, Jace Mogilly, "Remote I/O: Fast Access to Distant Storage".
[7] Yuhui Deng, "Deconstructing Network Attached Storage systems", aEMC Research China, Beijing 100084, PR China -2008.
[8] Arie Shoshani, Alex Sim, Junmin "Storage Resource Managers: Middleware Components for Grid Storage", GU Lawrence Berkeley National Laboratory Berkeley, California 94720.
[9] Yuhui Deng , Frank Wang, Na Helian, Sining Wu, Chenhan Liao (2008) "Dynamic and scalable storage management architecture for Grid Oriented Storage device" Parallel Computing 34 (2008) 17-31.