

Distributing the Sloan Digital Sky Survey Using UDT and Sector

Yunhong Gu and Robert L. Grossman

National Center for Data Mining

University of Illinois at Chicago

851 S. Morgan Street, Chicago, IL 60607 USA

yunhong@lac.uic.edu and grossman@uic.edu

Alex Szalay and Ani Thakar

The Johns Hopkins University

3701 San Martin Drive

Baltimore, MD 21218 USA

szalay@pha.jhu.edu and thakar@jhu.edu

Abstract

In this paper, we describe a peer-to-peer storage system called Sector that is designed to access and transport large data sets over wide area high performance networks. We also describe our recent experience using Sector to distribute the Sloan Digital Sky Survey BESTDR4 catalog data.

1 Introduction

Providing access to large scientific data sets is a challenging problem. The problem is especially difficult if the data sets are distributed. A number of e-science applications involve these types of data sets, including high energy physics [19], astronomy [24], and climate simulation. Fortunately, the emergence of wide area high performance networks and new network protocols designed to exploit the bandwidth available on such networks is enabling new types of approaches and solutions [10].

In this paper, we describe one such approach. We show how a peer-to-peer system called Sector [14] can be used to distribute large scientific data sets over wide area high performance networks. We also describe how Sector has been used recently to distribute data from the Sloan Digital Sky Survey (SDSS) [24].

A common approach today, that is used in the distribution of high energy physics data for example, is to use Globus to build a data grid for managing the data [19]. A data grid uses parallel TCP (through GridFTP) for data transport, data replication services, and grid services (and

more recently web services) to provide access to distributed collections of data. Globus provides a full security infrastructure for managing distributed resources, including authentication, authorization, and access controls. On the other hand, some users, especially those interested in accessing and exploring data that is freely available, when faced with the overhead of installing Globus and the complexity of developing applications with it, would prefer a simpler alternative. Sector is one such alternative.

We feel that Sector is innovative for the following reasons:

1. Sector is the first system that we are aware of for distributing large scientific data sets that is based upon a peer-to-peer storage system. Previously, these types of systems have been used mainly for distributing music and video files.
2. Sector is designed to exploit the bandwidth available in wide area, high performance networks, and to do this in a way that is fair to other high volume flows and friendly to traditional TCP flows [11]. Sector employs UDT [28] to achieve this and is the first system that we are aware of that uses alternatives to TCP to exploit the bandwidth available in these new emerging networks.
3. Sector is designed to provide *simple* access to remote and distributed data. No other infrastructure is required than a fast network and a small Sector client application. In contrast, installing and operating the infrastructure for a data grid can sometimes be challenging.

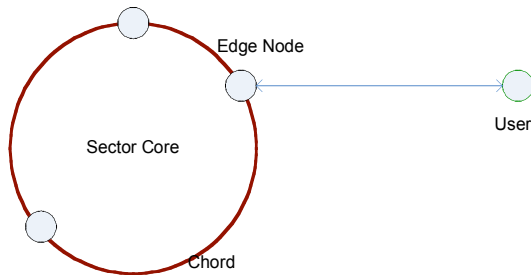


Figure 1. This diagram describes the Sector architecture. Data is stored on Sector Core nodes, which we assume are connected with a wide area, high performance network. A Sector client can access data transparently from any of the nodes that hold it in the Sector Core.

2 Requirements

Our goal was to design a system with the following requirements in order to support the access, integration and analysis of large scientific data sets, especially distributed data sets.

1. The first requirement was that the data should be accessible by name without having to know its location.
2. The second requirement was that some or all of the data should be able to be distributed over multiple nodes connected with a high performance network. With this architecture, the Sector software can then transfer the data from the Sector node or nodes with the smallest round trip times and the highest performance network links.
3. The third requirement was to support the scalability of the storage system by enabling additional Sector nodes to be added easily to the system.
4. The final and fourth requirement was that the system be simple and easy for client applications to use.

We assume that other essential requirements, such as security are managed outside of the Sector system.

3 Architecture

The Sector Core. The Sector architecture assumes that the Sector Core nodes are high-end workstations that are connected by high performance networks to each other. A

second assumption is that the nodes trust each other. If a node requests to join the Sector Core, at least one of the current nodes must add its IP address to the trust list. See Figure 1.

A Sector client can talk to any of the Sector nodes. If there are multiple core nodes that can provide the requested service, Sector selects an optimal node for a given user request. Currently, the node selection is based on the round trip time (RTT) between the user and each candidate node. Sector plans to use more sophisticated algorithms in the future.

A user request is processed as follows:

1. A user sends a request to a Sector edge node E_0 for data or a file X .
2. E_0 starts a search inside the Sector core to locate X , and finds a set of Sector nodes $E_1 - E_n$ that contain X .
3. E_0 chooses an optimal node E' from E_1 to E_n and tells the user the address of E' .
4. The user sends a request to E' for X and sets up a connection to transport the data or file. All access operations for the data or file are transmitted via this connection.

Group Messaging Protocol (GMP). We designed and implemented a protocol called the Group Messaging Protocol or GMP for messaging passing between Sector nodes. We chose not use TCP because as the number of Sector nodes increases, it becomes prohibitive to create a TCP connection between every two Sector nodes. Also, if Sector created a TCP connection on the fly as needed, there would be additional delays when responding to Sector client requests.

The GMP uses both UDP and TCP. Each GMP entity creates a UDP socket and a listening TCP socket. For most messages that are less than a threshold size (maximum UDP payload size), the message will be sent via UDP, and GMP manages the delivery reliability. If the message is so large that it cannot be packed into a single UDP packet, a TCP connection will be set up and the message delivered using TCP.

Both the implementation of the peer-to-peer routing protocol and the implementation of Sector itself use GMP for exchanging messages.

4 Chord

The routing protocol is derived from Chord [26]. The descriptor of a file is put on the node whose ID is the smallest among those nodes whose IDs are larger than or equal to the file ID. If no such node exists, the file descriptor is stored on the node with the smallest ID in the system.

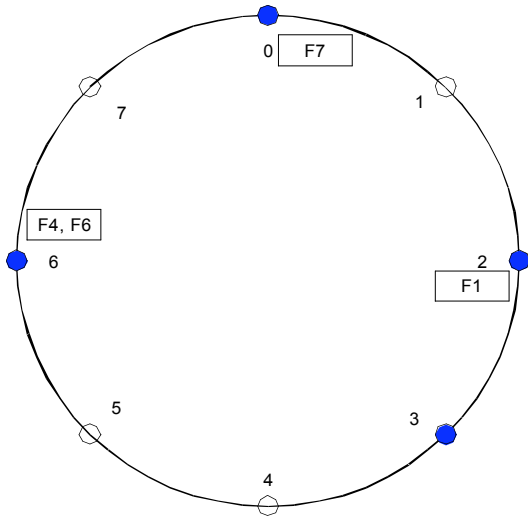


Figure 2. This diagram illustrates the Chord routing protocol for a simple example in which the file ID and node ID each use 3 binary digits.

For example, if both the node ID and file ID use 3 binary digits, there are at most 8 nodes and 8 files that can be managed by the system, as shown in Figure 2. In Figure 2, a dark dot represents a real node. A file with ID 1 is put on node 2, files with IDs 4 and 6 are put on nodes 6, and file with ID 7 is put on node 0.

More details about Chord can be found in [26]. Note that routing is an independent layer in the Sector system and that other protocols may be used in the future. We chose to use Chord in the first version of Sector because of its simplicity and ease of implementation.

5 Sector

Sector provides two levels of data access service. The first level is distributed file access by name. A file may have multiple copies in Sector and the location information can be obtained by the routing protocol (i.e., Chord). Once a connection is set up between the Sector node that contains a copy of the file and the user, the operation is defined by a file access protocol.

In detail, Sector file access service works as follows. Each Sector node maintains two indexes: the local file index and the remote file index. The local file index records all the file copies on the Sector node itself, whereas the remote file index maintains those files whose location information is on the node.

At the end of each interval of constant time, the local file index is scanned for each entry, and Sector checks if the file is correctly maintained in the remote file index of the proper Sector node (defined by Chord, see Figure 1). Meanwhile, the remote file index is scanned, and each entry is checked to guarantee that this is the correct node to store the location of the file and that the remote file does exist.

The periodical checking keeps updating these two indexes. Once a request comes, the remote file index is checked and the file location information is returned. Note that there may be a transient error due to a node joining or leaving the Sector system. Although such an error is inevitable, Sector keeps it transient: the errors will be fixed during the next index checking and the user can submit the request again if the first try fails.

The second level, which is not yet included in the current Sector software release, is to allow locating and accessing data by attributes or keys. Locating data by attributes follows the same process as locating a file by name, as described above. However, new components are required to provide data representation and simple SQL query support.

In addition, we will also introduce global indexes to provide broader searching service.

6 Implementation

Sector is implemented in C++ and includes Sector server software and Sector client software. The Sector Server is an application used for the Sector Core. A Sector node can start, join and leave the Sector Core using the server side software.

The client side software is actually an API that provides Sector access.

The current implementation of Sector uses version 3 of the high performance network protocol UDT [11], [12] for data and file access. The design is modular and it is easy to use other high performance protocols in future implementations. Version 3 of UDT is a high performance protocol *framework* and several different high performance network protocols may be deployed using it, including UDT itself, RBUDP, Scalable TCP, High Speed TCP and others [13].

Sector is open source and will be distributed by Source Forge in the near future.

Some screen shots may be seen in Figures 3 and 4.

7 Experimental Studies

During the past few months, we have used Sector to distribute the BESTDR4 catalog data from the Sloan Digital Sky Survey (SDSS) [24]. This data set is about one TB in size when compressed, or 1.7-1.8 TB when uncompressed.

To distribute this SDSS catalog data, Sector has been installed on several nodes connected via 10GE networks

SDSS Data Distribution using Sector and UDT

search

[Overview](#)

[Download Instructions](#)

[Software](#)

[Nodes Status](#)

[Downloading Records](#)

[Documentation](#)

[Technical Contact](#)

Related Links

[NCDM](#)

[SDSS](#)

[UDT](#)

[Sector](#)

[Teraflow Testbed](#)


Overview

Using this website, you can download the Sloan Digital Sky Survey (SDSS) data if you have access to a high speed wide area network. For example, if your organization is attached to the [National Lambda Rail](#) or Internet2's [Abilene Network](#), then you should be able to download the entire SDSS BESTDR4 catalog data set in less than two hours.

In general, it can be quite challenging to use effectively the available bandwidth over a wide area, high performance network. This project uses the UDP-based Data Transfer Protocol or UDT, which has been developed by the National Center for Data Mining (NCDM) at the University of Illinois at Chicago to make effective use of the bandwidth available from high performance wide area networks.

The project is supported by the National Science Foundation through the grant SCI II: The TeraFlow Project: High Performance Flows for Mining Large Distributed Data Archives, Award SCI-0430781.

Sloan Digital Sky Survey (SDSS)



The SDSS is systematically mapping a quarter of the entire sky, producing a detailed image of it, and determining the positions and absolute brightnesses of more than 100 million celestial objects. It is also measuring the distances to a million of the nearest galaxies, giving us a three-dimensional picture of the universe through a volume one hundred times larger than that explored to date. SDSS is also recording the distances to 100,000 quasars — the most distant objects known — giving us unprecedented knowledge of the distribution of matter to the edge of the visible universe.

Figure 3. This is an image of the web site for distributing Sloan Digital Sky Survey data using Sector.

Nodes Status

ID	IP	Alternative IP	Location	CPU	Memory (GB)	NIC (Gb/s)	Status
1	SL-1	NULL	Chicago	Opteron	2	10	●
2	SL-2	NULL	Chicago	Opteron	2	10	●
3	JGN2-1	NULL	Tokyo	Opteron	2	10	●
4	APAN	202.180.*.*	Tokyo	Operton	2	10	●
5	KISTI-1	203.230.*.*	Daejeon	Opteron	2	10	●
6	NASA-1	NULL	McLean	Opteron	2	10	●
7	NASA-2	NULL	McLean	Opteron	2	10	●
8	NASA-3	NULL	McLean	Opteron	2	10	●
9	NASA-4	NULL	McLean	Opteron	2	10	●

Figure 4. The Sector Code nodes currently consist of nodes in Chicago, Greenbelt, Tokyo, and Korea. All these nodes are connected by 10G networks.

that are optically interconnected at the StarLight Facility in Chicago [25]. In particular, Sector Core nodes have been installed in the following locations: Chicago, Illinois; Greenbelt, Maryland; Tokyo, Japan; and Daejeon, Korea. At each of these locations, either the entire SDSS BESTDR4 catalog data or portions of the catalog have been installed.

This SDSS catalog data is distributed via the web site `sdss.ncdm.uic.edu`. From this web site, a user can download a version of the Sector software that has been customized to distribute the SDSS BESTDR4 catalog data by simply including a list of the names of the files that constitute the BESTDR4 data set. A user simply issues a Sector get with this list of files and Sector will transport the data set to his or her workstation.

Table 2 shows the time required to transport the SDSS BESTDR4 data from Chicago to Greenbelt (18956 seconds), Maryland, Daejeon, Korea (35716 seconds), and Tokyo, Japan (55735 seconds). The time required by Sector clients to download the SDSS data is logged by the Sector nodes serving the SDSS data and additional information, including these logs, is available from the web site `sdss.ncdm.uic.edu`. These timings do not include the times required to verify the data. The files transferred are DBMS server backup files.

We emphasize that although the current Sector Core Nodes are connected with high performance networks, the Sector Core Nodes contain commodity disks. Copying data over local area networks using these disks proceeds achieves a performance of about 400-500 Mbps. For additional information about the equipment used in the testbed, see Table 1.

8 Related Work

Distributed file systems are described in [27], [1], [15]; parallel file systems are described in [6] and [3]; and peer-to-peer file sharing systems are described in [22], [7] and [18].

The Network File System (NFS) is a remote file access protocol for remotely sharing files. The Andrew File System (AFS) [15] and the Distributed File System (DFS) [16] provide improved cache performance and coherence for wide area applications. Frangipani [27] and xFS [1] are serverless cluster file systems. Both are designed to provide good performance and availability.

Parallel file systems, such as Vesta [6] and PVFS [3] provide parallel access and an API suitable for parallel computing, but are not designed for wide area applications.

In contrast to these systems listed above, Sector does not provide a directory service; instead, all files are identified by a globally unique name. Furthermore, Sector relies on high performance networks and specialized high performance network protocols to obtain high performance. Viewed this

way, Sector is not a distributed file system, but rather a specialized peer-to-peer storage system.

In addition to Chord [26], there are also other routing protocols, such as Pastry [23], Tapestry [29], and CAN [20]. Pastry and Tapestry use a prefix based lookup mechanism, and they also consider locality. CAN routes messages in a k -dimension space. On the other hand, Chord uses a distributed hash technique to distribute file IDs uniformly. CAN and Chord make no effort to achieve network locality. Instead, with the necessary information about other nodes, files can be accessed from any node, despite its locality or lack of locality. This provides good scalability, but may also mean longer lookup delays.

There are several storage systems that have been developed based on these lookup mechanisms, including PAST [22], OceanStore [18], and CFS [7]. Both PAST and OceanStore examine the network topology to achieve locality. Replicas in OceanStore can also relocate over time according to usage patterns. CFS uses Chord as a lookup mechanism.

We do not have Globus running over the Sector nodes, so we could not directly compare Sector to gridFTP [5]. On the other hand, from prior experimental studies [11], we expect gridFTP to achieve approximately the same bandwidth. The Globus replication infrastructure [5] provides an alternative mechanism for replicating large scientific data sets. From a pure performance point of view, Sector provides no advantage over Globus and Globus data replication mechanisms. On the other hand, the underlying usage model is fundamentally different. Globus is a secure infrastructure enabling collaborations to share computational and data resources. Sector is a high performance distributed storage and data system designed to run as an application that provides access to distributed files and data using a peer-to-peer architecture and employing high performance network protocols. Sector employs a data web security model not a data grid security model.

9 Summary and Conclusion

We have described a peer-to-peer distributed storage system called Sector that is designed a) to store large data sets over a set of distributed Sector Core nodes connected by high performance networks, and b) to enable Sector clients to access this data simply and easily.

Since Sector uses high performance network protocols, such as UDT [11] designed to exploit the bandwidth available on wide area high performance networks without the necessity of specialized installation or tuning, accessing data using Sector can be over a hundred times faster than using protocols such as TCP, as it is commonly deployed.

We have described some of our experiences distributing the one Terabyte Sloan Digital Sky Survey (SDSS)

node	CPU	Memory	Disk	Disk Read	Disk Write	NIC
Chicago	Dual Opteron250 2.4GHz	4GB	1.5TB 4-Disk RAID 5	56MB/sec	50MB/sec	Intel 10GE LR
Tokyo	Dual Opteron250 2.4GHz	4GB	1.5 TB 4-Disk RAID 0	190MB/sec	140MB/sec	Intel 10GE LR
Daejeon	Dual Opteron250 2.4GHz	4GB	2.0 TB 4-Disk RAID 0	190MB/sec	180MB/sec	Intel 10GE LR
Greenbelt	Dual dual-core Opteron 1.8GHz	4GB	2.0 TB 6-Disk RAID 5	160MB/sec	90MB/sec	Intel 10GE SR

Table 1. This table shows the hardware configuration of path of teraflow testbed nodes used in the experiments.

Source	Destination	Total Time	Bandwidth - Mean	Max	Min	Stdev
Chicago	Greenbelt	18956	344	378	0	54.6
Chicago	Tokyo	55735	163	203	0	52.8
Chicago	Daejeon	35716	229	244	135	13.7
Tokyo	Tokyo	NA	228	304	174	36.2

Table 2. Some experimental results of distributing the SDSS BESTDR4 data set using Sector over nodes connected with 10G networks. The data set was divided into 64 files and the mean is the mean bandwidth obtained averaged over these 64 files. The maximum and minimum are the maximum and minimum over these 64 files, as is the standard deviation. The mean, max, and min are all measured in Mbps. Total time is measured in seconds. Notice that Sector transfers data over the wide area network between Chicago and Tokyo at about 70% of the speed that ftp transfers the data over the local area network connecting the two Sector nodes in Tokyo. Of course, with higher performance disks, these transfer speeds would be higher.

BESTDR4 catalog data using Sector. In particular, we are currently distributing this data using Sector Core nodes in Chicago, Greenbelt, Tokyo, and Daejeon. We have described some measurements transporting data to each of these sites using Sector. In particular, we note that using Sector data can be moved to a node in Tokyo at over 70% of the speed that it can be transferred to that node over a local area network from another node in the same rack.

Since the Sector software can be installed at the application layer and does not require any special security infrastructure, it is substantially easier for many users to install than a typical data grid application or a solution that requires substantial tuning of the TCP stack to achieve high performance.

To summarize, we have demonstrated by the experimental studies in this paper that by using systems such as Sector, we can effectively and practically distribute large scientific data sets using wide area high performance networks.

References

- [1] Thomas E. Anderson, Michael D. Dahlin, Jeanna M. Neefe, David A. Patterson, Drew S. Roselli, and Randolph Y. Wang. Serverless network file systems. In Proceedings of the 15th Symposium on Operating Systems Principles, pages 109–126, Copper Mountain Resort, Colorado, December 1995. ACM.
- [2] Beck, M., Moore, T., and Plank, J.S. An end-to-end approach to globally scalable network storage. Proc. of ACM SIGCOMM '02, Pittsburgh, August 2002.
- [3] P. H. Carns, W. B. Ligon III, R. B. Ross, and R. Thakur, "PVFS: A Parallel File System For Linux Clusters", Proceedings of the 4th Annual Linux Showcase and Conference, Atlanta, GA, October 2000, pp. 317-327.
- [4] A. Chervenak, R. Schuler, C. Kesselman, S. Koranda, B. Moe, Wide Area Data Replication for Scientific Collaborations, Proceedings of 6th IEEE/ACM International Workshop on Grid Computing (Grid2005), November 2005.
- [5] A. Chervenak, I. Foster, C. Kesselman, C. Salisbury, and S. Tuecke, The Data Grid: Towards an architecture for the distributed management and analysis of large scientific datasets, Network Storage Symposium (NetStore '99), October 1999.
- [6] Corbett, P. F. and Feitelson, D. G. 1996. The Vesta parallel file system. ACM Transactions on Computer Systems 14, 3 (August), 225–264.

- [7] Frank Dabek, M. Frans Kaashoek, David Karger, Robert Morris, and Ion Stoica. Wide-area cooperative storage with CFS. In Proceedings of the 18th ACM Symposium on Operating Systems Principles (SOSP '01).
- [8] I. Foster, C. Kesselman, G. Tsudik, S. Tuecke. A Security Architecture for Computational Grids. Proc. 5th ACM Conference on Computer and Communications Security Conference, pp. 83-92, 1998.
- [9] Robert Grossman, and Marco Mazzucco, DataSpace - A Web Infrastructure for the Exploratory Analysis and Mining of Data, IEEE Computing in Science and Engineering, July/August, 2002, pages 44-51.
- [10] Ian Foster and Robert L. Grossman, Data Integration in a Bandwidth Rich World, Communications ACM, Volume 46, Issue 11, November, 2003, pages 50-57.
- [11] Robert L. Grossman, Yunhong Gu, Xinwei Hong, Antony Antony, Johan Blom, Freek Dijkstra, and Cees de Laat, Teraflows over Gigabit WANs with UDT, Journal of Future Computer Systems, Elsevier Press, Volume 21, Number 4, 2005, pages 501-513.
- [12] Yunhong Gu, Xinwei Hong, and Robert Grossman, Experiences in Design and Implementation of a High Performance Transport Protocol, ACM/IEEE International Conference for High Performance Computing and Communications (SC '04), page 22.
- [13] Yunhong Gu and Robert Grossman, Supporting Configurable Congestion Control in Data Transport Services, ACM/IEEE International Conference for High Performance Computing and Communications (SC '05).
- [14] Yunhong Gu and Robert L. Grossman, The Design and Implementation of the Sector Peer-to-Peer Storage System, in preparation.
- [15] J. Howard, M. Kazar, S. Menees, D. Nichols, M. Satyanarayanan, R. Sidebotham, and M. West. Scale and Performance in a Distributed File System. ACM Transactions on Computer Systems, 6(1):51-81, February 1988.
- [16] Kazar, M. L., Leverett, B. W., Anderson, O. T., Apostolides, V., Bottos, B. A., Chutani, S., Everhart, C. F., Mason, W. A., Tu, S. and Zayas, E. R., "DEcorum File System Architectural Overview", Proceedings of the Summer 1990 USENIX Conference, Anaheim, CA, June 11-15 1990, 151- 164.
- [17] C. Kommareddy, N. Shankar, and B. Bhattacharjee. "Finding Close Friends on the Internet", IEEE ICNP, November 2001.
- [18] Kubiawicz J, Bindel D, Chen Y, Czerwinski S, Eaton P, Geels D, Gummadi R, Rhea S, Weatherspoon H, Weimer W, Wells C, Zhao B. 2000. OceanStore: An architecture for globalscale persistent store. In: Proceedings of ASPLOS'2000; 2000; November; Cambridge, MA; Pages 190-201.
- [19] Particle Physics Data Grid, retrieved from www.ppdg.org on June 10, 2006.
- [20] Sylvia Ratnasamy, Paul Francis, Mark Handley, Richard Karp, and Scott Shenker. A scalable content-addressable network. In Proc. ACM SIGCOMM 2001, August 2001.
- [21] P. Rodriguez, A. Kirpal, and E. W. Biersack. Parallel-access for mirror sites in the Internet. In Proc. of IEEE Infocom 2000, volume 2, pages 864-873, Tel Aviv, Israel, Mar. 2000.
- [22] Rowstron A, Druschel P. 2001. Storage management and caching in PAST, a largescale, persistent peer-to-peer storage utility. In: Proceedings of ACM SOSP'01; 2001; October; Banff, Canada; Pages 188-201.
- [23] Rowstron A, Druschel P. 2001. Pastry: Scalable, distributed object location and routing for large-scale peer-to-peer systems. In: Proceedings of IFIP/ACM Middleware; 2001; Heidelberg, Germany.
- [24] Adelman-McCarthy et al. 2006. The Fourth Data Release of the Sloan Digital Sky Survey, In: The Astrophysical Journal Supplement, vol. 162, issue 1, pp. 38-47.
- [25] StarLight, retrieved from www.startap.net on June 4, 2006.
- [26] Stoica I, Morris R, Karger D, Kaashoek M. F, Balakrishnan H. 2001. Chord: A scalable peer-to-peer lookup service for Internet applications. In: Proceedings of ACM SIGCOMM'01; August; San Diego, CA; Pages 149-160.
- [27] C. A. Thekkath, T. Mann, and E. K. Lee. Frangipani; A Scalable Distributed File System. In Proceedings of the 16th ACM Symposium on Operating Systems Principles, Oct. 1997.
- [28] UDP-Based Data Transport Protocol, retrieved from udt.sf.net on June 10, 2006.
- [29] Zhao B. Y., Kubiawicz J. D., Joseph A. D. 2001. Tapestry: An infrastructure for fault-resilient wide-area location and routing. Technical Report UCB//CSD-01-1141, U.C. Berkely; 2001; April.