A SIMULATION STUDY OF DATA DISCOVERY MECHANISM FOR SCIENTIFIC DATA GRID ENVIRONMENT

* A. Abdullah, * M. Othman, * M. N. Sulaiman,
* H. Ibrahim, ** and A. T. Othman

** Faculty of Computer Science and Information Technology, Universiti Putra Malaysia, 43400 Serdang, Selangor*

*** Faculty of Information Technology and Communication, Universiti Pendidikan Sultan Idris, 35900 Tanjong Malim, Perak*

Email : { azizol, mothman,nasir,hamidah }@fsktm.upm.edu.my
Email : abutalib@upsi.edu.my

ABSTRACT

Research in the area of grid computing has given us various ideas and solutions to address the requirements in a modern scientific computing community that manage massive amounts of a very large data collections in a geographically distributed environment. Data Grids mostly deal with large computational problems and provide geographically distributed resources for large-scale data-intensive applications that generate large data sets. A number of research groups are working on the data distribution problems in Data Grids and they are investigating the data replication approaches on the data distribution. This leads to a new problem in discovery of and access to data in the Data Grid environment. To address this problem we have developed a model to study various discovery mechanisms and investigate these mechanisms for Dynamic Scientific Data Grid Environments using our Grid Simulator. In this paper, we illustrate our model and our Grid Simulator.

Key words: replication, data grids, data discovery

1. 0 INTRODUCTION

The community of researchers in modern scientific computing is involved in managing massive amounts of very large data collections through a geographically distributed environment. Recently, research in the area of grid computing has given us various ideas and solutions to address these requirements. There are two aspects in grid technology: sharing of data, which is referred to as Data Grids, and sharing of resources such as processors, which is referred to as Computational Grid. In the scientific community, these two aspects of grid technology are required to form a high-performance computing environment. However, this paper focuses only on the Data Grid system.

A Data Grid system connects a collection of geographically distributed computer and storage resources that may be located in different parts of a country or even in different countries, and enables users to share data and other resources (Chervenak *et al.*, 2000). Within the Grid community, much work has been done on providing the basic infrastructure for a typical grid environment. Recently, there are a few research projects directed towards the development of data grids such as Particle Physics Data Grid (PPDG, n.d.), Grid Physics Network (GriPhyN, n.d.), The China Clipper Project (Johnston *et al.*, 1998), and Storage Request Broker (SRB, n.d.). All these projects aim to build a scientific data grid that enables scientists sitting at various universities and research labs to collaborate with one another and share data sets and computational power. The size of the data that needs to be accessed through the Data Grid is in the order of terabytes today and is soon expected to reach petabytes. As an example, The Large Hadron Collider experiment (LHC, n.d.) is producing terabytes of raw data a year, and by 2005, this experiment at CERN, will produce petabytes of raw data a year that needs to be pre-processed, stored, and analysed by teams comprising of thousands of physicists around the world. Through this process, more derived data will be produced and hundreds of millions of files will need to be managed and stored at more than hundreds of participating institutions. All data must always be available or guaranteed that they will always be easily located.

In scientific environments, even data sets which are created once and then remain read only, their usage often leads to the creation of new files, inserting a new dimension of dynamics into a system (Loebel-Carpenter *et al.*, 2001; Lueking *et al.*, 2001). Ensuring efficient access to such huge and widely distributed data is a serious challenge to network and Data Grid designers. The major barrier of supporting fast data access in a Data Grid system is high

latencies of Wide Area Networks (WANs) and the Internet, which affect the scalability and fault tolerance of total Data Grid systems. There are a number of research groups working on the data distribution problems in Data Grid (Ranganathan and Foster, 2001a; Ranganathan and Foster, 2001b; Stockinger *et al.*, 2001; Vazhkudai *et al.*, 2001). They are still working and investigating various data replication approaches on the data distribution for the Data Grid, in order to improve its ability to access data efficiently. These lead to new problems in discovery of and access to data in the Data Grid environment since data will be replicated, and this replica set changes dynamically as new replicas are being added and old ones are being deleted at runtime in the Data Grid system. To address these problems we have developed a model based on Peer-to-Peer (P2P) architecture to study various discovery mechanisms and investigate these mechanisms for our Dynamic Scientific Data Grid Environment. To evaluate our model and selected discovery mechanisms, we have developed our own Data Grid Simulator using PARSEC (PARallel Simulation Environment for Complex systems) to generate different network topologies and replication strategies to study the impact of discovery mechanisms on the cost of locating data and the data access on the overall Dynamic Scientific Data Grid.

The paper is organised as follows. Section 2 gives an overview of previous work on discovery mechanism. In Section 3, we describe our model. Section 4 describes the simulation framework and our testing results. Finally, we present a brief conclusion and future directions in Section 5.

2.0 DISCOVERY MECHANISMS AND SCIENTIFIC DATA GRIDS

The discovery mechanism is a basic functionality that enhances the accessibility of data in the distributed system context. Support for handling and discovering resource capabilities already exist in some metacomputing system such as Globus (Foster and Kesselman, 1997) and Legion (Grimshaw *et al.*, 2000). In Globus, Resource Broker is responsible for resource discovery within each administration domain, which works with an Information Service and a Co-allocator for monitoring current state of resources and managing an ensemble of resources respectively. Hence, application requirements in Resource Specification Language (RSL) are refined by Brokers into more specific requirements, until a set of resources can be identified. The Legion system describes resources and tasks as a collection of interacting objects. A set of core

objects enable arbitrary naming of resources based on Legion Object Identitifers (LOIDs) and Legion Object Addresses (LOA). Specialised services such as Binding agents and Context objects are provided to translate between its arbitrary resource name and its physical location – enabling the resource discovery to be abstracted as a translation between LOIDs and physical resource location. In CONDOR (Frey *et al.*, 2001), resource describes their capabilities as an advertisement, which is subsequently matched with an advertisement describing the needs of an application. Most of these mechanisms use a centralised approach, which may not scale well.

In a large and dynamic environment, network services such as discovery should be decentralised in order to avoid potential computation bottlenecks at a point and be more scalable. In the last couple of years, the Peer-to-Peer (P2P) system became fashionable (Gnutella, n.d.; The Free Network, n.d.; Napster, n.d.; Konspire, n.d.; Gong, 2001). They emphasise two specific attributes of resource sharing community: scale and dynamics. Most of existing P2P systems such as Gnutella, Freenet, CAN (Ratnasamy *et al.*, 2001) and Chord (Stoica *et al.*, 2001) provide their own discovery mechanisms and focus on specific data sharing environments and, therefore, on specific requirements. As an example, the Gnutella emphasis is on easy sharing and fast file retrieval, but with no guarantees that files will always be located. In Freenet, the emphasis is on ensuring anonymity. CAN and Chord guarantee that files are always located but accepting increased overhead for file insertion and removal.

According Iamnitchi *et al.* (2002), the typical uses of shared data in scientific collaboration have particular characteristics *Group locality* – users tend to work in a group and use the same set of resources (data sets). The newly produced data sets will be of interest to all users in the group. *Time locality* – the same users may request the same data multiple times within short time intervals. However, file discovery mechanisms that are proposed in existing P2P systems such as Gnutella, CAN, Chord, and etc., do not attempt to exploit this behavior. Iamnitchi *et al.* (2002) have also raised a few questions. One of the questions that is of most interest to us is how do we translate the dynamics of scientific collaborations in self-configuring network protocols such as joining the network, finding the right group of interests, adapting to changes in user's interest, etc.? This is both relevant and challenging in the context of self-configuring P2P networks. We support this idea and try to answer the question by developing a framework model, which is discussed in the next section.

3.0 P2P DATA GRIDS FRAMEWORK MODEL

In this section, we illustrate our framework model that provides a generic infrastructure to deploy P2P services and applications. Figure 1 shows our proposed framework model for a data grid environment that will support scientific collaboration. In our framework model, peers could be any network devices and in our implementation, the peers can include PCs, servers and even supercomputers. Each peer operates independently and asynchronously from all other peers and it can self-organised into a peer group. A peer group contains peers that have agreed upon a common set of services, and through this peer group, peers can discover each other on the network. Once a peer joins a group, it uses all the services provided by the group. Peers can join or leave the group at anytime they choose to. In our implementation, once a peer joins the group, all the data sets that are shared by others peers in the group will be available to him/her. Peers can also share any of their own data sets with others peers within the group. Peer may belong to more than one group simultaneously.

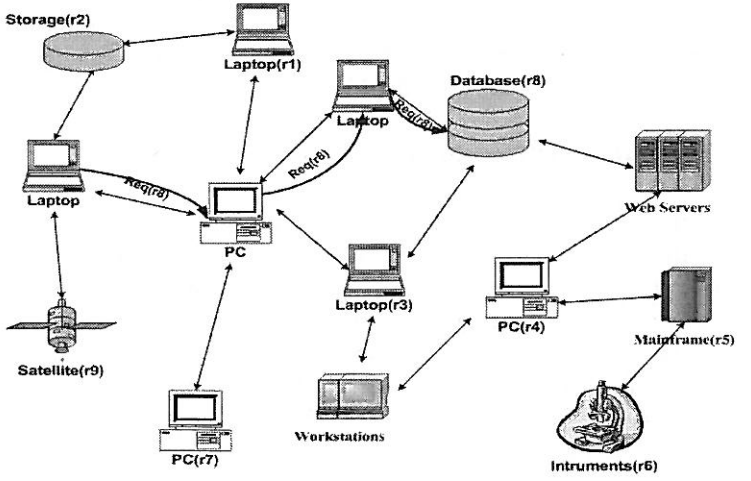

Fig. 1 P2P Framework Model for Data Grid

In designing this collaborative data-sharing model, we are following the ideas of the power grid environment. In the case of power grid supply, the users are not concerned about which power generator delivers the electricity to their home appliances, but they are only concerned that the power is available to drive their appliances correctly. In our scientific community data-sharing model, we try to

provide the same concept as the analogy of an electrical power grid, where the users or scientists, in our case, peers, can access their required data sets without knowing which peer delivers the data sets. In other words, they can execute their applications, get the remote data sets (without concern about the provider) and then wait for the results. This will all be done by the discovery mechanism. Our focus in this research is to propose a dynamic discovery mechanism for the Dynamic Scientific Data Grid Environment.

4.0 SIMULATION FRAMEWORK AND RESULTS

A simulation has many advantages over actual implementation/deployment since it offers a fully customisable and controllable environment, and it gives us another alternative to conduct an experiment without deploying on the actual network infrastructure. With our limited resources, we could not deploy our experiment on the actual network infrastructure that we have. However, by developing and using the simulator we can conduct our experiments in a customisable and controllable network environment. The presented results were obtained with the simulation of only read requests without any background traffic in the network and the stream of requests for the Data Grid. The current simulation is described in the next subsection.

4.1 The Data Grid Simulator

Our simulator is written in PARSEC simulation language. PARSEC is a C-based discrete-event simulation. In PARSEC, an object or set of objects is represented by a logical process. The simulator consists of three parts. The first part is the DRIVER entity that is responsible for creating the rest of entities: the entities that simulate the network nodes and network layer. It also reads all the inputs needed for the simulation. The second part is the network layer, which comprises of two entities: CHANNEL entity and DVMRP entity. The CHANNEL entity simulates a network forwarding protocol and the DVMRP entity simulates the Distance Vector Multicast Routing Protocol. The third part is the NETWORK_NODE entity that simulates the various nodes in the Data Grid.

4.2 Methodology of Simulation Study

The Data Grid Simulator is developed to identify suitable resource discovery strategies and mechanisms for the Dynamic Scientific Data Grid Environment.

The simulator is developed using a modular design approach that divides the simulator into various parts. An important benefit of the modular design is that it is relatively easy to replace any of the modules. Figure 2 shows our simulation architecture framework for the data grid environment. Starting a simulation first involves specifying the topology of the grid, including the number of network nodes, how they are connected to each other and the location of the files across various nodes. The bandwidth of each link is not specified in our current simulation model in order to simplify the model.

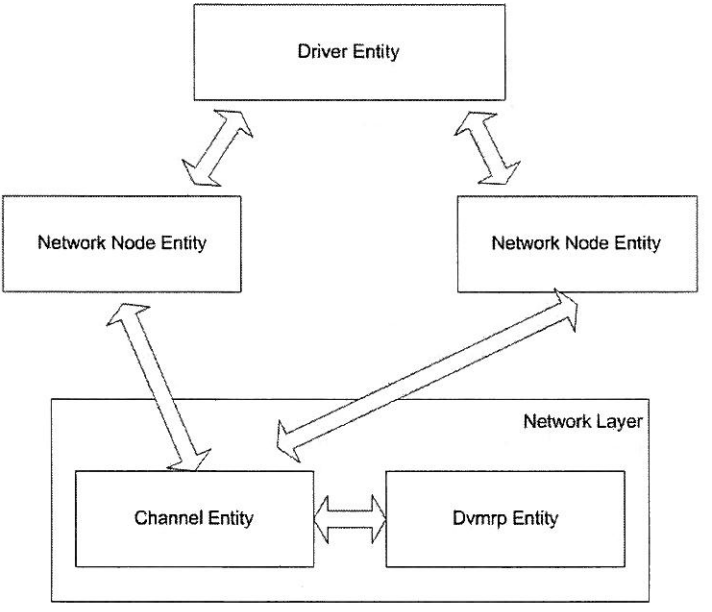

Fig. 2: The Data Grid Simulator Architecture

In this simulation model, the NETWORK_NODE will trigger various file requests originate according to file access patterns to be simulated. The CHANNEL will forward the request based on the information in the routing table provided by DVMRP. Before forwarding the request to several neighbour nodes, it will check the file required and submit the request if the node has the requested file. In this case, the simulator will record the hop count for the successful request. If not, it will forward the request to other nodes until its time-to-live (TTL) for the request has expired.

There are various proposed strategies and mechanism to discover or locate the resources in a distributed system. We studied and compared different discovery strategies using this simulator. Through this simulation, various parameters such as hop count, success rates, response time and bandwidth consumption can be measured. However, in this paper, we will only show the hop count measurement. The simulation was first run on the access patterns that were generated randomly. Table 1 is a sample access pattern file randomly generated during simulation. This being the worse case scenario, more realistic access patterns that contained varying amounts of temporal and geographical locality will be generated in the next version of our simulator.

Table 1: A Sample Access Pattern File

Time	File Name	Requestor Node
20	File1	1
39	File2	5
70	File2	6
75	File3	3
90	File1	2
92	File2	3
98	File3	5

4.3 Simulation Experiments and Results

In a Data Grid system, several factors influence the performance of discovery mechanism;, which are the network topology, resource frequency and resource location. However, in our simulation studies, we will only focus on the last two factors that are related to the replication strategies. Replication strategies are incorporated into the simulation so as to create the dynamic scenario of a Data Grid system. In the real Data Grid system, the replica management system decides when to create a replica and where to place it.

Currently, in our simulation, we have generated random flat network topologies consisting of 10 to 100 nodes. We placed the data file ourselves, and measure the data access cost in average number of hop and successful number of request. In this current simulation, we assume that the overlay network graph topology does not change during the simulation so as to maintain simplicity. In the simulation, we implemented a decentralised discovery mechanism based on flooding and request forwarding technique to test our model and our data grid

simulator. In a flooding protocol, nodes flood an overlay network with queries to discover a data file. In a request forwarding technique, nodes will forward the queries to the selected neighbor. In testing our model and our data grid simulator, we ran a number of simulations. In the first scenario of our simulations, we assume that no replicas are created at the beginning and during the simulation. In the second scenario, we created replicas at the beginning of simulation. We used the average number of hops and the number of "success" requests as our performance metric as it represents the data discovery. Figure 3 shows a graph representing the data access cost in average number of hop for different topologies with different number of nodes without replication using the flooding discovery mechanism.

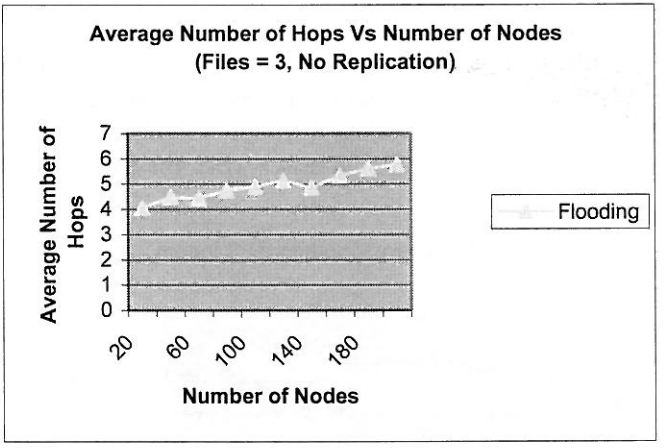

Fig. 3: Average Number of Hop in Different Number of Nodes

The result from figure 3 shows the performance of the discovery mechanism based on flooding becomes worse when the number of nodes increases in the collaboration network. We expect that in a real Dynamic Data Grid Environment, it will become worse as this mechanism may not provide the guarantees required by collaborators and it also does not scale well when the number of nodes increases. The mechanism might give optimal results in a network with a small to average number of nodes.

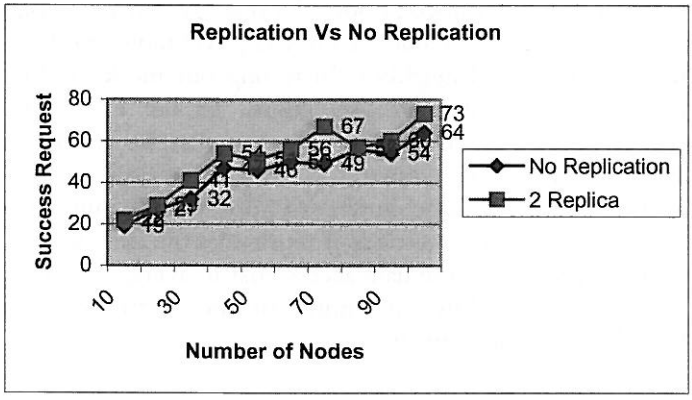

Fig. 4a: Number of Success Queries for Flooding Mechanism

Figure 4a shows a graph representing the data access cost in number of success queries for different topologies with different number of nodes with and without replication using the flooding discovery mechanism. The result shows that, the number of success queries increases when the data file is replicated.

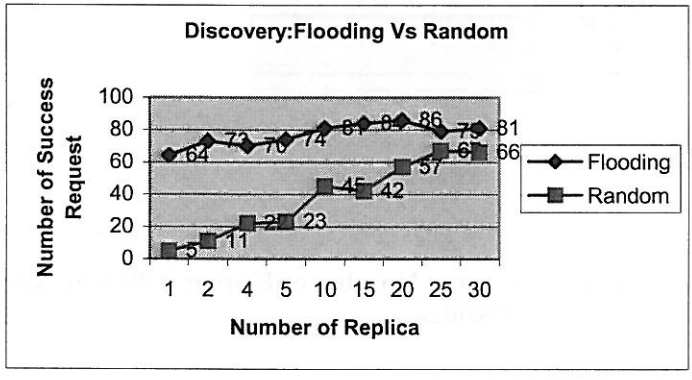

Fig. 4b: The 100 Nodes Network Topology with Various Numbers of Replicas

Figure 4b, shows a graph representing a network topology with 100 nodes and the number of replicas created. In this simulation experiment, we show the impact of replication on two different discovery mechanisms: flooding and random. The result shows that, when the number of replicas increases, the number of success queries for both discovery mechanisms also increases. For

the random discovery mechanism, the rate of success queries is lower than the rate of success queries for flooding mechanism.

5. 0 CONCLUSIONS AND FUTURE WORKS

We have addressed the problem of discovery in the Dynamic Data Grid Environment. We have also described our prototype implementation for Dynamic Data Grid Environment and our Data Grid Simulator for our study. Although the prototype is conceptually simple, it has provided a service that is remarkably useful for the Data Grid Environment research. We believe that, with the very simple simulation experiment that we have done on our model using our Data Grid Simulator, this approach can be used to support good services for data discovery in the Dynamic Data Grid Environment. It also represents an interesting approach to data management in the Data Grid.

Our results show that, in a Dynamic Data Grid Environment with a large population of users, we need a discovery mechanism that could give the shortest time to discover and locate data resources and also provide the guarantees required by collaborators. Although the replication process leads to a new problem for locating and discovering data, it shows that replication has given a good impact on the discovery process, where it gives a better time for discovery. It also gives a higher success rate for queries and improves data availability. In our future work, we will study various proposed strategies and mechanisms to discover or locate the resources in a decentralised environment. Our simulator provides a modular framework within which optimisation strategies for data discovery strategies can be studied under different data grid configurations. Various replication strategies will be incorporated to create more a dynamic data grid scenario. More realistic access patterns that contain varying amounts of temporal and geographical locality will be generated. The ultimate goal of our study is to explore and propose a Decentralised Discovery Mechanism for Dynamic Data Grid Environment.

REFERENCES

Chervenak, A., Foster, I., Kesselman, C., Salisbury, C., & Tuecke, S. (2001). The data grid: Towards an architecture for the distributed management and analysis of large scientific data sets. *Journal of Network and Computer Applications, 23*,187-200.

Foster, I., & Kesselman, C. (1997). Globus: A metacomputing infrastructure toolkit. *International Journal of Supercomputer Applications, 11*(2) 115-128.

Frey, J., Tannenbaum, T., Foster, I., Livny, M., Tuecke, S. (2001). Condor-G: A computation management agent for multi-institutional grids. *Proceedings of the Tenth IEEE Symposium on High Performance Distributed Computing (HPDC10).* August 7-9. (pp. 55). San Francisco, California: IEEE Press.

Gnutella website (n.d.). Retrieved April 20, 2002, from http://www.gnutella.wego.com

Gong, L. (2002). *Project JXTA: A Technology Overview.* Technical Report. Sun Microsystems Inc. Palo Alto, California.

Grimshaw, A., Lewis, M., Ferrari, A., & Karpovich, J. (2000). Architectural support for extensibility and autonomy in wide-area distributed object systems. In *Proceedings of the 2000 Network and Distributed System Security Symposium (NDSS2000).* February 2-4. Reston, Virginia: Internet Society.

GriPhyn: Grid Physics Network website, (n.d.). Retrieved April 23, 2002, from http://www.griphyn.org

Iamnitchi, A., Ripeanu, M., & Foster, I. (2002). Locating data in (Small-World?) Peer-to-Peer scientific collaborations. *1st International Workshop on Peer-to-Peer Systems.* March 7-8. (pp. 232-241). Cambridge, Massachusetts: Springer-Verlag.

Johnston, W., Lee, J., Tierney, B., Tull, C.,& Millsom, D. (1998). The China clipper project: A data intensive grid support for dynamically configured, adaptive, distributed, high-performance data and computing environments. *Proceedings of Computing in High Energy Physics.* August 31 – September 4. Chicago: Argonne National Laboratory.

Konspire website (n.d.). Retrieved April 20, 2002, from http://konspire.sourceforge.net

LHC- The Large Hadron Collider website (n.d.). Retrieved April 23, 2002, from *http://lhc-new-homepage.web.cern.ch/lhc-new-homepage/*

Loebel-Carpenter, L., Lueking, L., Moore, C., Pordes, R.., Trumbo, J., Veseli, S., et al. (2001). SAM and the particle physics data grid. *Proceedings of Computing in High-Energy and Nuclear Physics.* September 3-7. Beijing, China: IHEP Chinese Academy of Sciences.

Lueking, L., Loebel-Carpenter, L., Merritt, W., & Moore, C. (2001). The DO experiment data grid – SAM. In C.A. Lee (Ed.), *GRID 2001, Lecture Notes in Computer Science, 2242,* Springer-Verlag Berlin Heidelberg. 177-184.

Napster website (n.d.). Retrieved April 20, 2002, from http://www.napster.com

PPDG: Particle Physics Data Grid website (n.d.). Retrieved April 23, 2002, from http://ppdg.net

Ranganathan, K., & Foster, I. (2001a). Identifying dynamic replication strategies for a high performance data grid. *Proceedings of the International Grid Computing Workshop.* November 12. (pp. 75). Denver: Springer-Verlag.

Ranganathan, K., & Foster, I. (2001b). Design and evaluation of replication strategies for a high performance data grid. *Proceedings of Computing in High-Energy and Nuclear Physics.* September 3-7. (pp. 161-172). Beijing, China: IHEP Chinese Academy of Sciences.

Ratnasamy, S., Francis, P., Handley, M., Karp, R., & Shenker, S. (2001). A scalable content-addressable network. *Proceedings of Special Interest Group on Data Communication Conference (SIGCOMM 2001).* August 27 – 31. San Diego, USA: ACM Press.

SRB: Storage Request Broker website (n.d.). Retrieved April 23, 2002, from http://www.npaci.edu/DICE/SRB

Stockinger, H., Samar, A., Allcock, B., Foster, I., Holtman, K., &Tierney, B. (2001). File and object replication in data grids. *Proceedings of the Tenth International Symposium on High Performance Distributed Computing.* August 7 – 9. (pp. 76). San Francisco, California: IEEE Press.

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 Special Interest Group on Data Communication Conference*

(SIGCOMM 2001). August 27 – 31. (pp. 149-160). San Diego, USA: ACM Press.

The Free Network Project website (n.d.). Retrieved April 20, 2002, from http://freenetproject.org

Vazhkudai, S., Tuecke, S., & Foster, I. (2001). Replication selection in the globus data grid. *Proceedings of the First IEEE/ACM International Conference on Cluster Computing and the Grid.* May 15 – 18. (pp. 106-113) , Brisbane, Australia: IEEE Computer Society Press.