私立東海大學

資訊工程與科學研究所

碩士論文

指導教授：楊 朝 棟 博士

資料網格上使用貝氏網路的副本維護策略

A File Replication Maintenance Strategy Using Bayesian Network in Date Grids

研 究 生：黃健榮

中華民國九十七年七月

摘要

資料網格使得分散在不同區域上的計算及儲存資源(同質或異質)，可以達到分享、選擇、以及相互的溝通的應用。尤其是需要分析大量且密集資料的科學實驗，諸如高能物理、生物資訊的運用、以及氣象的模擬等，透過應用資料網格的都獲得良好的問題解決方式。近幾年來，許多研究學者持續地研究資料網格中副本機制的應用。副本機制可以讓一份資料產生多份的副本並且分散儲存於不同的地方，使得世界各地的研究學者可以容易取得所需的資料，縮短資料取得時間，提高資料分析的效率。由於網路的頻寬狀態屬於變動，使用者所需的資料會隨著時間變動，資料存取點亦需隨之變更。在本論文中，提出了一個基於動態管理服務與貝式網路下的一個副本維護策略。資料副本會動態的調整至適當的位置以供使用者使用，並避免存取次數與資料量隨著節點的增多而造成過多負擔。

關鍵字： 網格計算、資料網格、副本管理、動態管理服務、貝氏網路

Abstract

Data grids are a very important and useful technique for solving problems created by the large amounts of data scientific experiments and simulations produce. Data replication, a technique much discussed by data grid researchers in past years, creates multiple copies of files and stores them in convenient locations to shorten file access times. In this thesis, we propose a dynamic replication maintenance service for maintaining data in grid environments. Replicas are adjusted to appropriate locations using Bayesian Networks (BN) due to the continuous change of the network. The maintenance strategy we propose is called Implicit Dynamic Maintenance Service with Bayesian Network (IDMSBN). If the probability value exceeds the user–defined threshold, the IDMSBN can duplicate the same file to the next best site without replication, in order to reduce the network bandwidth when the user requests a file.

Keywords: Grid Computing, Data Grid, Replica Management, Dynamic Maintenance Service, Bayesian Networks

Acknowledgements

It is time to say thank you to all of those people who have helped me through the completion of this thesis. In particular, I would like to thank my advisor, Dr. Chao-Tung Yang, who introduced me to this topic and gave me a broad support and guidance. Professor Yang gave me a deep influence and inspiration. I would also like to thank Professor Fang-Yie Leu, Professor Chao-Chin Wu, Professor Ching-Hsien Hsu and Professor Kuan-Chou Lai for their help for their valuable comments and advice while serving on my reading committee.

The growth and the success are not merely from one's efforts. There are many people whom I would like to thank; especially my group-mate Shih-Yen Chen who help me construct experimental environment. My research would not have been completed without the help from them. And enormously thank for headshot of TT, James, Pica, Be to let me have some thesis sense with "GoGoGo" last tree months.

Finally, I am grateful to my family and my friends whose unconditional support made this thesis possible.

Tables of Contents

List of Tables

List of Figures

Chapter 1
Introduction

1.1 Motivations

In recent years, many high-energy physics projects have been executed in Europe [3], as well as climate predictions, earthquake simulations, and bioinformatics research, all over the world. These all deal with large-scale files in the terabyte and even petabyte range and require increasing amounts of computing power, network bandwidth, and storage capacity for optimum performance. But computer centers with supercomputers cannot support enough of these device needs. Grid technology is the best approach to this kind of problem. The main purpose of the Grid Project [9, 10, 23] is to enable sharing of computing and storage resources such as CPUs, memory, and hard disks, among others, by users geographically distributed around the world, thus forming a massive virtual computer [1, 2]. Grids need some kind of middleware to integrate equipment and communication for experiments and simulations. The Globus Toolkit [11, 23] is an open-source middleware package for building grid environments. It includes components for security, information infrastructure, resource management, data management, communication, fault detection, and portability.

Data replication [5, 22], which operates based on the Globus Toolkit, can distribute data to many scattered sites. Superior replication strategies can reduce file access times and latency, and accelerate file download speeds to reduce execution times. And if one storage element crashes, client nodes can still fetch desired data

from other storage elements, thus improving fault tolerance and making the entire grid environment more stable and reliable. Another advantage of data grids is that users can download data in parallel from chosen sites or applications chosen automatically after estimation by evaluation models in grid environments. Bandwidth utilization is the most important factor affecting download speeds and execution times. Since network environments are unstable, no replica site is always the best choice for downloading data in minimum time. Thus, replicas should be distributed among appropriate grid storage elements to reduce the time required for users to get data, and to improve execution performance.

1.2 The Goal and Contributions

This thesis presents a replica maintenance strategy called the Implicit Dynamic Maintenance Service with Bayesian Network (IDMSBN), which differs from the Dynamic Maintenance Service (DMS) [31] and others. The DMS framework automatically maintains data using relative statuses and information metadata such as data-access frequency, free space on storage elements to which data will be replicated or migrated, and network conditions between the file site and other sites. Both methods collect server metadata. DMS needs more storage space to record site-to-site metadata, and when the number of sites and files increases, this can expand system IO and reduce performance.

The contributions of this thesis provide another aspect of replication in Bayesian Networks, and reduce metadata read and write frequency times. IDMSBN also adds some effective factors for describing condition replica prediction probabilities.

1.3 Thesis Organization

The rest parts of this thesis are organized as follows. In chapter 2, we describe the background knowledge of grid computing, data grid, some grid middleware, Bayesian Network and some related works about data replication strategies have been proposed before. In chapter 3, we describe the details of design and implement of data grid framework, system component, and the algorithm of IDMSBN, also we will introduce our parameters and evaluation model to determine when should data be adjusted to the appropriate locations. In chapter 4, we simulate some scenarios by using the IDMSBN algorithm compare with other replication strategies and show the experimental results. In the last part, chapter 5, we describe the conclusion and the future works of this thesis.

Chapter 2
Background

2.1 Grid Computing

Grid computing involves sharing heterogeneous resources, based on different platforms, hardware/software, computer architecture, and computer languages, which located in different places belonging to different administrative domains over a network using open standards to solve large-scale computation problems. As more Grids are deployed worldwide, the number of multi-institutional collaborations is rapidly growing. However, for Grid computing to realize its full potential, it is expected that Grid participants are able to use one another resource [9, 10].

Functionally, Grids can be classified as computational Grid and data Grid. The computational Grid is the beacon to scientists for solving large-scale problems like gene comparison, high-energy physics, earthquake simulation, and weather prediction, etc. The subject of this work is the resource management and allocation for a Grid system that is primarily intended to support computationally expensive tasks like simulations and optimizations on a Grid [17, 25-30].

2.1.1 Data Grid

Data Grid is another important topic for grid computing. Data Grid aggregated the storage devices distributed in different area around the world by the Internet into a

virtual storage device to store the data which it capacity is much larger than a physical storage device can store. There are many advantages that Data Grid can offer:

- Make jobs can find suitable file quickly: If jobs needed some data to compute, it can check the information stored in database about which sites stored the desired data, and get those desired file stored in the local region or in the nearest region to reduce the time spend on getting those files.
- Improve the reliability of data grid: If the data is popular, it will have many copies stored in different file sites in the grid environment. When the file site is broken, the jobs can get the same files from other data sites to prevent the jobs stay in idle for getting those files.
- Reduce time spends on accessing and downloading files: When jobs wanted some files to compute, they can divide each of those file into several parts and parallel downloading those several parts from many data sites which stored those desired files to reduce the time spend on getting those files.
- Make the load balanced of file servers: The file sites will replicate many copies of data to other sites. If other jobs needed the files, they can get the desired files in the appropriate file sites to avoid every job getting the data from the same file site. It can make the load of all file sites be balanced.
- Improve performance of all system: By the advantages that we had mentioned above, the Data Grid can improve the performance of all grid system and make the grid system be more stable.

In [25], S. Venugopal et al. proposed the grid topology can be categorized into four models. On the following are descriptions of those four models.

- Monadic: As figure 2-1 shows, there is only one central storage device of all grid environment in this model. It store all data which gathered by scientific detectors. If users want to get data, there is only a single point to query and

provide the desired data that users needed. In this model, only institutions which get the permission can replicate data to the local storage devices, or replication is used to handle the fault tolerance not the usage ratio of data in locality.

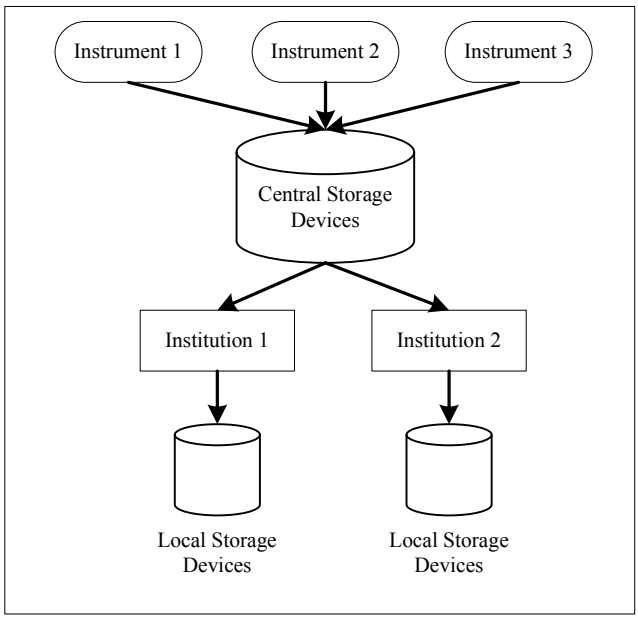


Figure 2-1. Monadic

- Hierarchy: As figure 2-2 shows, in the hierarchy model all the data comes from a single point, Tier 0. It gathers all the data which gathered by scientific detectors and stores the data in its storage devices. If researchers needed the related data in distributed area around the world, it will first transmit the related data to Region Centers (RCs). Then Region Centers transmit the related data downward to the institutions, researchers get their desired data for experiments from local storage devices of institutions. The bandwidth between Tier 0, Tier 1, Tier 2, and Tier 3 is decrease progressively. One advantage of this model is to keep the data in consistency can be easier than other model due to the source comes from a single point, Tier 0.

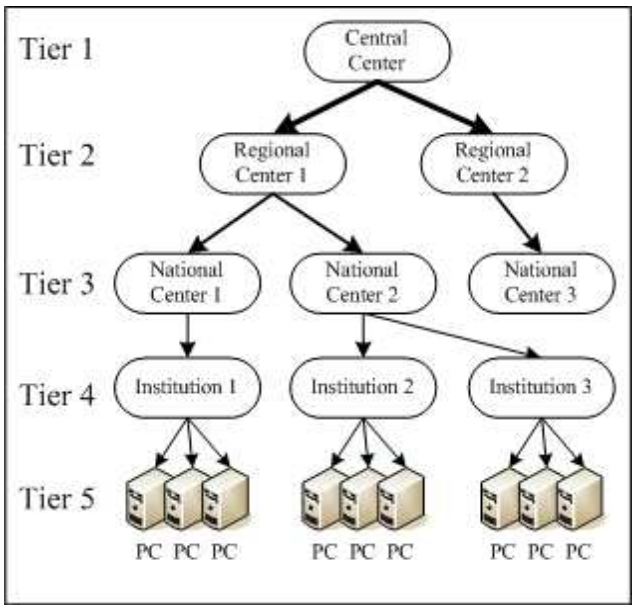


Figure 2-2. Hierarchy

- Federation: As figure 2-3 shows, the federation model exists in institutions or universities which want to share data or resource to each other. Researcher can query their desired data even if the data does not exist in the local storage device, and get the data from other institutions or universities by authentication. Each institutions and universities can control data only in its local storage device. For data replication and data consistency is more complex.

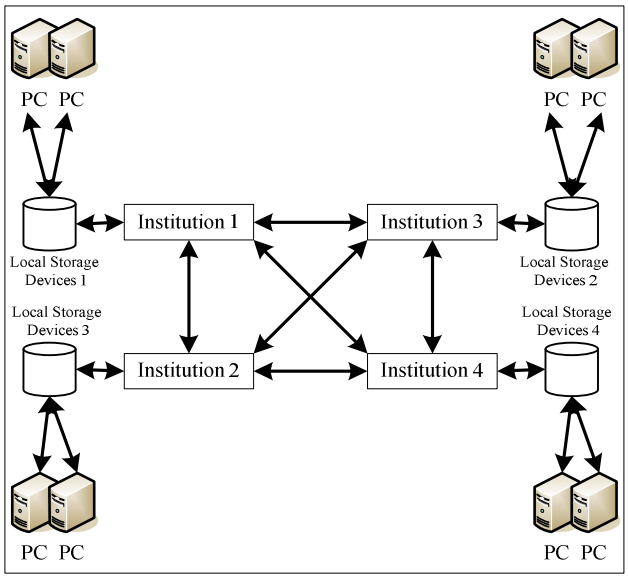


Figure 2-3. Federation

- Hybrid: As figure 2-4 shows, the hybrid model consists of previous grid models, monadic, hierarchy, and federation. The central storage device allocates data downward to the distributors, and the distributors also allocate data downward to institutions. The institutions connected to each other and also query/provide data to each other. This comes out researchers who want to share their products of their analysis to others.

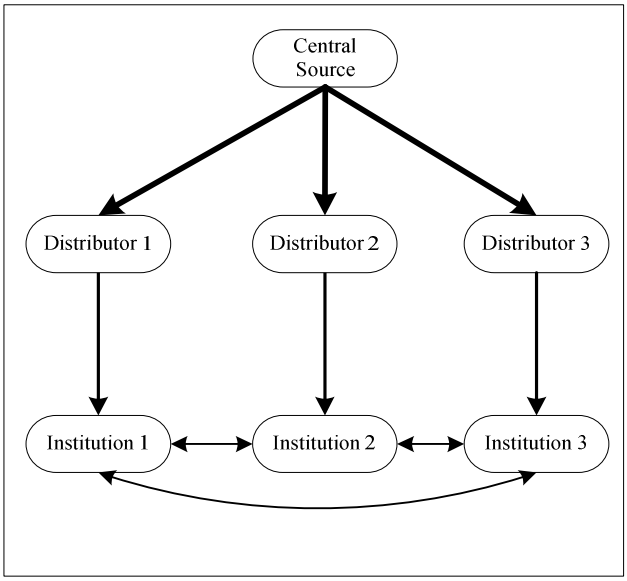


Figure 2-4. Hybrid

2.1.2 Replica Management

Replica management involves creating or removing replicas at a data grid site [24, 25]. In other word, the role of a replica manager is to crate or delete replicas, within specified storage systems. Most often, these replicas are exact copies of the original files, created only to harness certain performance benefits. A replica manager

typically maintains a replica catalog containing replica site addresses and the file instances. The replica management service is responsible for managing the replication of complete and partial copies of datasets, defined as collections of files.

The replica management service is just one component in a Data Grid environment that provides support for high-performance, data-intensive applications. A replica or location is a subset of a collection that is stored on a particular physical storage system. There may be multiple possibly overlapping subsets of a collection stored on multiple storage systems in a Data Grid. These Grid storage systems may use a variety of underlying storage technologies and data movement protocols, which are independent of replica management.

2.1.3 Replica Catalog

The purpose of the replica catalog is to provide mappings between logical names for files or collections and one or more copies of the objects on physical storage systems. The catalog registers three types of entries: logical collections, locations and logical files. A logical collection is a user-defined group of files. We would expect that users will find it convenient and intuitive to register and manipulate groups of files as a collection, rather than require that every file is registered and manipulated individually. Aggregating files should reduce both the number of entries in the catalog and the number of catalog manipulation operations required to manage replicas. Location entries in the replica catalog contain all the information required for mapping a logical collection to a particular physical instance of that collection. The location entry may register information about the physical storage system, such as the hostname, port and protocol. In addition, it contains all information needed to

construct a URL that can be used to access particular files in the collection on the corresponding storage system.

Each location entry represents a complete or partial copy of a logical collection on a storage system. One location entry corresponds to exactly one physical storage system location. The location entry explicitly lists all files from the logical collection that are stored on the specified physical storage system. Each logical collection may have an arbitrary number of associated location entries, each of which contains a (possibly overlapping) subset of the files in the collection. Using multiple location entries, users can easily register logical collections that span multiple physical storage systems.

Despite the benefits of registering and manipulating collections of files using logical collection and location objects, users and applications may also want to characterize individual files. For this purpose, the replica catalog includes optional entries that describe individual logical files. Logical files are entities with globally unique names that may have one or more physical instances. The catalog may optionally contain one logical file entry in the replica catalog for each logical file in a collection.

A Data Grid may contain multiple replica catalogs. For example, a community of researchers interested in a particular research topic might maintain a replica catalog for a collection of data sets of mutual interest. It is possible to create hierarchies of replica catalogs to impose a directory-like structure on related logical collections. In addition, the replica manager can perform access control on entire catalogs as will as on individual logical files.

2.2 Globus Toolkit and GridFTP

The Globus Project [11, 23] provides software tools collectively called The Globus Toolkit that makes it easier to build computational Grids and Grid-based applications. Many organizations use the Globus Toolkit to build computational Grids to support their applications. The composition of the Globus Toolkit can be divided as three major components: Resource Management, Information Services, and Data Management. Each of them makes use of a common foundation of security. Grid Resource Allocation and Management Protocol (GRAM) implement a resource management protocol, Monitoring and Discovery Service (MDS) implements an information services protocol, and GridFTP implements a data transfer protocol. They all use the Grid Security Interface (GSI) security protocol at the connection layer [11, 23].

In Data Grid environments, access to distributed data is typically as important as access to distributed computational resources. Distributed scientific and engineering applications require transfers of large amounts of data between storage systems, and access to large amounts of data generated by many geographically distributed applications and users for analysis and visualization, among others. Unfortunately, the lack of standard protocols for transferring and accessing data in Grids has led to a fragmented Grid storage community. Users wishing to access various storage systems are forced to use multiple protocols, and it is difficult to transfer data efficiently between these various storage systems.

The Globus Project surveyed available protocols and technologies, implemented some prototypes, settled on using FTP and its existing extensions as a base, and then extending it again to add missing required functionality. The Globus alliance proposed a common data transfer and access protocol called GridFTP that provides secure,

efficient data movement in Grid environments. This protocol, which extends the standard FTP protocol, provides a superset of the features offered by the various Grid storage systems currently in use.

2.3 Network Weather Service (NWS)

The Network Weather Service (NWS) [14] is a distributed system that detects network status by periodically monitoring and dynamically forecasting over a given time interval. The service operates a distributed set of performance sensors (network monitors, CPU monitors, etc.) from which it gathers system condition information. It then uses numerical models to generate forecasts of what the conditions will be for a given time period. The system includes sensors for end-to-end TCP/IP performance (bandwidth and latency), available CPU percentage, and available non-paged memory. The sensor interface, however, allows new internal sensors to be configured into the system.

2.4 Bayesian Networks

Bayesian Networks (BN) can represent joint probability distributions among variables and clearly show conditional independence. A BN has two parts: a set of parameters, and a structure containing nodes that can represent any kind of variable and arcs connecting probabilistically related nodes. BN graphs are directed acyclic graphs (DAGs) that cannot start and end at the same node. The NB in Figure 2-5 represents probability relationships between diseases and symptoms. If there is an arc from Node A to Node B, A is called a parent of B, and B a child of A. “parents (Xi)” denotes the

set of parent nodes of node Xi. A directed acyclic graph is a BN relative of a set of variables if the joint distribution of the node values can be written as the product of the local distributions of each node and its parents.

$$P(X_1, X_2, ..., X_n) = \prod_{i=1}^{n} P(X_i \mid parents(X_i)) \quad (1)$$

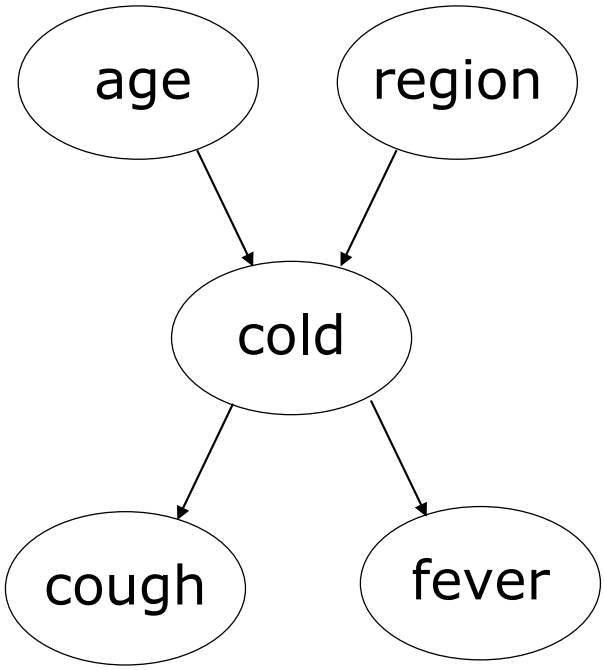


Figure 2-5. An example of a Bayesian network

2.5 Related Works

Many studies have been done on data maintenance in data grids.

In [20, 25], Ranganathan and Foster introduced six dynamic replication strategies and compared them using a simulator called PARSEC to measure the average response time and total bandwidth consumed by each strategy. The lower the response

time and bandwidth consumption, the better the replication strategy. But response time vs. bandwidth consumption is a tradeoff. The authors concluded that if a user considers response time to be the most important issue, then the Cascading strategy is the better choice. On the other hand, if a user considers bandwidth consumption to be the most important issue, then Fast Spread is the better choice among the six strategies evaluated. These two strategies do not check whether there is enough free space to store temporary job data and results. Data no longer popular will still occupy a lot of space in storage elements. This wasted storage space may affect performance.

The authors of [7, 12, 18, 19] all mentioned the p-median problem, defined as: “For a set of n client points, find a set of p server points that minimizes the total distance from each client point to its nearest source point.” Minimizing total distance in grid environments minimizes total response time. The authors in [7, 12] and many previous works indicated that the p-median problem complexity class is NP-hard in two or more dimensions. The connection statuses between nodes in grid environments can be displayed as a matrix, so a grid environment can be defined as a two-dimensional matrix, which means finding p nodes to serve all nodes in grid environments is also NP-hard. In [18, 19], Rashedur M. Rahman et al., proposed a static replica placement algorithm for putting replicas in p-best candidate nodes to minimize total node response times via Lagrangean Relaxation, a heuristic approach [8] to measuring the response times of client nodes to their nearest server nodes. The algorithm is most likely the p-median problem. They also take user requests and network latency as parameters in deciding when to maintain replicas dynamically. They use OptorSim [15] , a simulator developed by the EU Data Grid project to compare their method, called Dynamic p-median, with Static p-median and Best Client. Static p-median replicates no replicas to other data grid environment nodes when user requests and/or network latency change. Best Client replicates desired data

to client nodes when request ratios for certain node files are very high. Simulation results show that for various network loads and user requests, the average response time of the authors' methods are the lowest. Although Dynamic p-median is good for maintaining replicas dynamically, it can't be used in real grid environments because it is NP-hard and would waste a lot of computing power determining new locations for replicas.

In [16], Sang-Min Park et al., proposed a dynamic replica maintenance algorithm called Bandwidth Hierarchy based Replication (BHR). They divide sites into many regions such that sites close to one another are in the same regions according to the bandwidth hierarchy. If a new replica already exists in another site in the same region, the BHR optimizer terminates. In [4], Ruay-Shiung Chang and Jih-Sheng Chang indicated that the BHR algorithm performs better than other strategies only when the storage element capacity is small. We found another problem in BHR. Consider the following scenario: If a file must be replicated in a region, BHR replicates a copy to another site in the region. If that same file must be replicated to a third site in the same region, BHR will see that there is already a duplicate file in the region and terminate the algorithm. Thus, files can have, at most, two copies in each region for parallel downloading of computing data [1-2, 26-30]. The BHR time cost is limited by having only these two download links, and high performance cannot be achieved. Furthermore, this limitation will cause load imbalances on the two sites that have the copies stored in them.

The issue discussed above and the problems pointed out in [16] have been correct by DMS. The Dynamic Maintenance Service (DMS), which use the request frequency and free space of a storage element as parameters to determine when files should be adjusted. In DMS algorithm, the same data in one site will have more than two replicas. And the low access frequency of file will automatically be deleted, and that

will save more free space of storage elements to increase the usage ratio of storage elements to store temporal data or results produced by the jobs.

Some issues concerning predictive techniques for replica selection in grid environments have been explored. R. M. Rahman, K. Barker, R. Alhajj [32] used a neural network (NN) to predict transfer times for various replica sites. NNs mimic neurons in the human brain and can be taught behaviors. They used predictor trace data with a back-propagation algorithm to train the NN to predict current data-transfer performance in grid throughput.

S.Vazhkudai, J. Schopf, [33] used a multi-regression replica selection model to deal with prediction and found that selecting the best site for replication may not yield the best site for the current network state because grid environments are unstable.

Bayesian Methods often are developed for use in diagnostic systems [34] and can detect outbreaks of disease, whether natural or bioterrorist-induced. Several algorithms provide *a priori* and *a posteriori* probabilities in Bayesian Networks. Calculating joint probabilities in multi-variable network structures is difficult with normal personal computing power today, since this was long ago shown be an NP-hard problem. So we find references [35, 36, 37, and 38] to attempts to simplify structures, combine variables, and/or eliminate over-fitting.

Chapter 3
System Design and Implementation

3.1 Software Stack Diagram

Software stack diagrams of our data grid system are shown in Figure 3-1 and Figure 3-2. There are three layers in the data grid system, bottom, middle, and top. The functions of each layer are described below.

- **Bottom Layer:** contains the software installed in each node in the grid environment. The two major components in the Bottom Layer are the Information Provider and Grid Middleware. The Information Provider consists of two blocks, Ganglia and Network Weather Service (NWS). Ganglia gathers machine information such as numbers of processors and processor cores, processor loading, total memory and free memory sizes, and disk usage. NWS is used to gather network bandwidths between nodes and link latencies. The Grid Middleware consists of the Globus Toolkit, which is used to join nodes to the grid environment.
- **Middle Layer:** consists of sites. Each site is made up of several nodes, usually located in the same place or connected by the same switch or hub; all site nodes are connected to one another via the Internet. Moreover, sites are usually built up as clusters with each node having a real IP. The first node in each site is called the Head Node.
- **Top Layer:** components include Applications, Services, the Monitoring Service, and Records. Users can easily control the grid environment with Services, which consist of the Anticipative Recursively-Adjusting

Mechanism, Replica Selection Service, One-way Replica Consistency Service, and Dynamic Maintenance Service provided by the portal. Services operate according to information gathered by the Monitoring Service and Records. Records provide machine and file information before downloading or adjusting file locations. The Monitoring Service provides a web front-end page for users to observe variations during job processing.

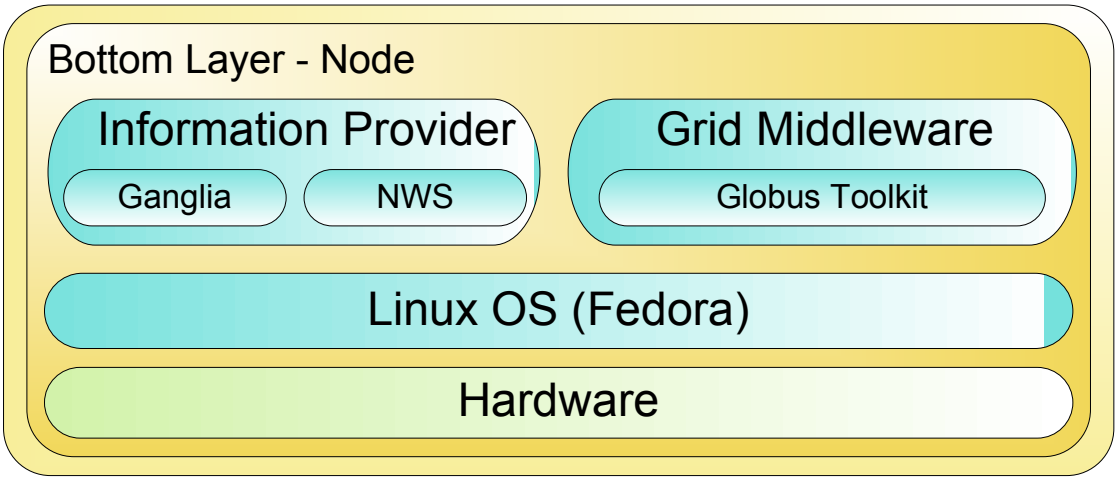


Figure 3-1. Sites, services, and portal software stack

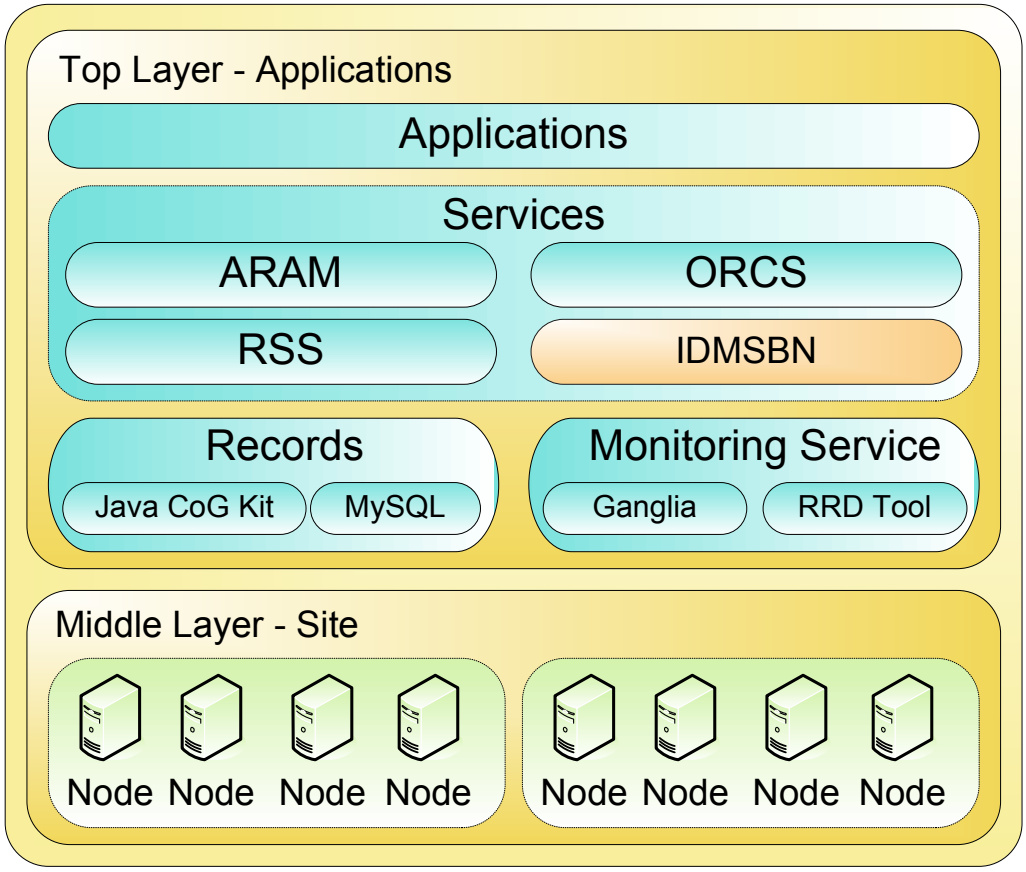


Figure 3-2. Node software stack

The relations between the components described above are shown in Figure 3-3. The four services mentioned above can be divided into two kinds, User-side, and System-side. The User side, consisting of the Anticipative-Recursively-Adjusting Mechanism (ARAM) and Replica Selection Service (RSS), enables users to monitor application operations.

The System-side, consisting of the Implicit Dynamic Maintenance Service with Bayesian Network (IDMSBN) and One-way Replica Consistency Service (ORCS), automatically direct files to appropriate locations and keep them in consistency. Functional details of these four services are described below.

- **Replica Selection Service:** gathers information from the RLS and Information Service to determine sites from which the ARAM can best download files.
- **Anticipative Recursively Adjusting Mechanism:** consists of co-allocation architecture that enables users to download desired data in parallel. It dynamically adjusts download speeds for all file sites according to network bandwidths between server nodes and client nodes, thus balancing file site loadings.
- **One-way Replica Consistency Service:** keeps files consistent with duplicates stored in distributed nodes. When a file is updated, it notifies the other nodes that have the same file to update to the newest version.
- **Implicit Dynamic Maintenance Service with Bayesian Network (IDMSBN):** is the major service in this thesis. It dynamically duplicates grid environment files by measuring variable parameters. This reduces execution times and helps stabilize the system. It can also increase storage device usage-ratio efficiency.

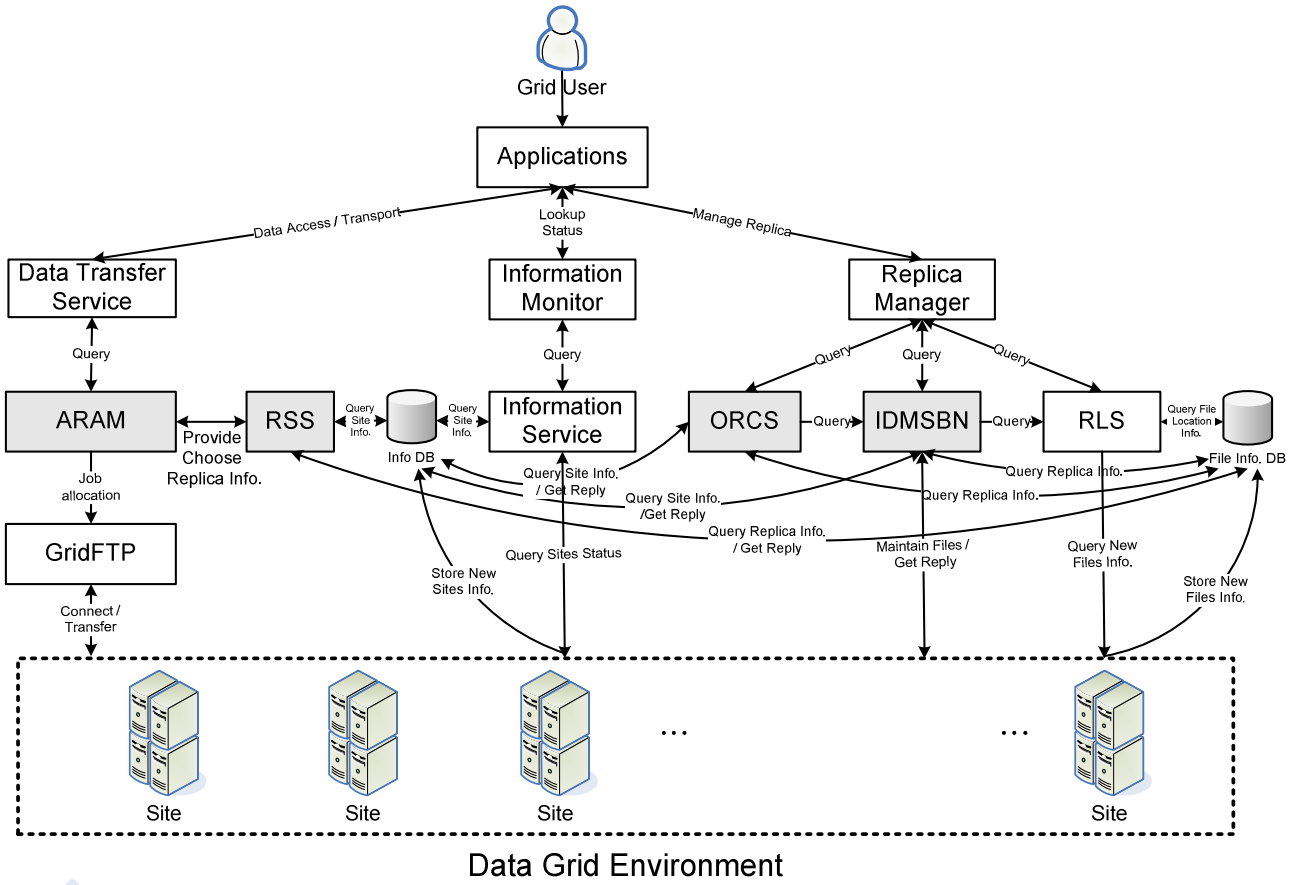


Figure 3-3. Data Grid System Architecture

3.2 The Operation of IDMSBN

The major contribution of this thesis is reporting on construction of a dynamic replica maintenance service for data grid systems: IDMSBN; its operation is shown in Figure 3-4. Prior to IDMSN Dynamic File Maintenance (DMS), the Information Service and Replica Location Service record relevant information in the database for IDMSBN to measure using the cost model described below in Subsection 3.3. The Information Service and Replica Location Service functions are described below:

- **Information Service:** periodically gathers variable idle ratios, such as CPU, Memory, storage device free space, and network bandwidth It records real-time information on these dynamic factors in the Information Database (Info. DB) for IDMSBN to use.
- **Replica Location Service:** records information on each file, such as logical file name, file size, physical location of the file, time of file creation or last update, and file access frequency in the File Information Database (File Info. DB). The Replica Location Service may then be used to find files for downloading in locations nearest the user in the grid environment.

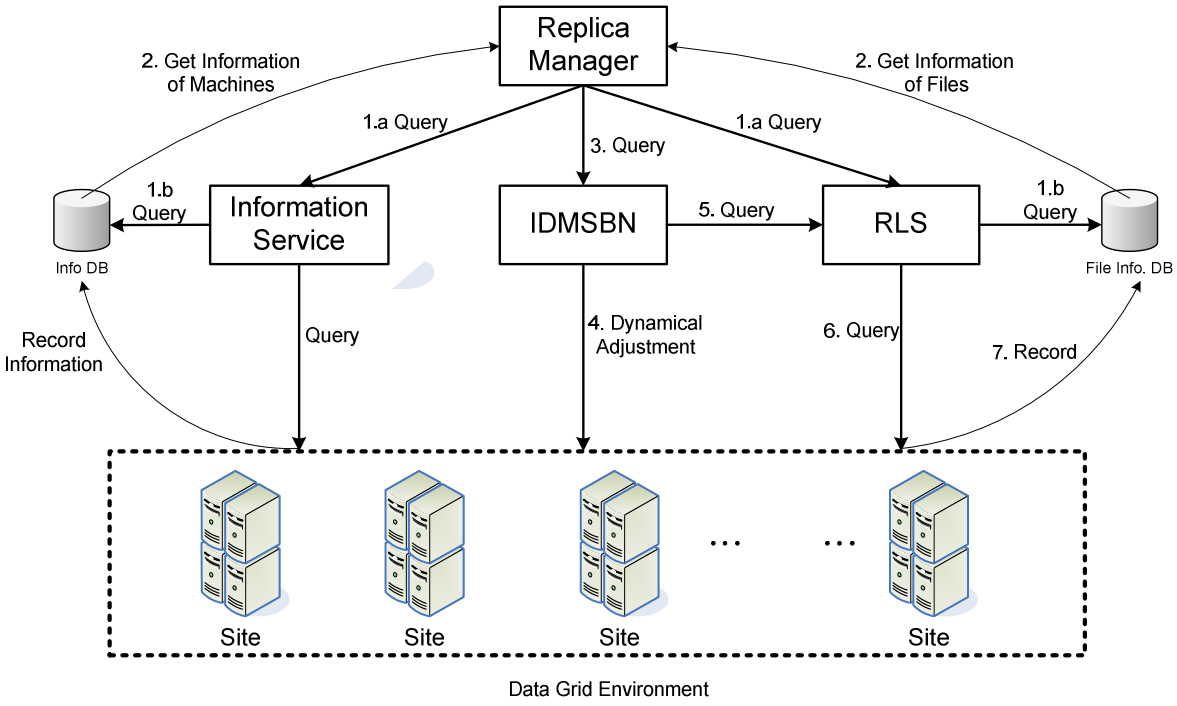


Figure 3-4. IDMSBN operation

Before triggering the IDBMS, the Replica Manager first queries the Information Service and Replica Location Service, which each individually query the Information Database and File Information Database to get all information on the system status

and files. The Replica Manager then evaluates to determine whether any files need adjustment. If there are, the Replica Manager sends a query to the Implicit Dynamic Maintenance Service with Bayesian Network to adjust those files, after which the Dynamic Maintenance Service sends a query to the Replica Location Service to check the new statuses of all files in the grid environment. The Replica Location Service responds by checking file statuses, and then recording the new information in the File Information Database.

3.3 Parameters and Evaluation Model

In this part, we will describe the some used parameters, the definitions of measurable parameters, and the evaluated models which we used to measure the performance of IDMSBN and others replication strategies.

3.3.1 Affect Parameters

Many factors in the grid environment affect replica transfer times and execution performance. For this reason, many parameters, static and dynamic, as well as other factors that affect overall performance must be calculated. The few factors we chose for the model are shown below.

- **Static Parameters:** These factors do not change when the grid environment changes. As Xuanhua Shi et al., mentioned in [21], these factors are system attributes of each site, such as CPU type and frequency, hard disk storage capacity, memory capacity, and network card transfer rate. Generally, the

faster the CPU frequency, the larger the memory and hard disk capacities, and the faster the network card transfer rate, the better the choice for grid users to execute jobs. These cannot be major factors in measuring grid performance due to the changeable nature of grid environments, thus, we focus on the dynamic factors.

- **Dynamic Parameters:** can change when the grid environment changes. Executing jobs consumes computing power, uses memory space, downloads or uploads data, and stores computational results in storage elements. This changes the CPU usage rate, memory space, bandwidth, and node free space. Network bandwidth is the most important factor affecting performance. To achieve high performance, the real-time requirement must be achieved. The NWS was created to periodically monitor and dynamically forecast the performance of various network and computational elements. We use NWS [14] to measure network bandwidth, and the Linux commands, “sar” and “df” to measure CPU and memory usage, and hard disk free space.
- **Other Parameters:** The parameters mentioned above can easily be used to quantify, but other parameters also influence data replica operations. For example, sites being located in the same region, or the date of replica creation. The best parameters may be those found to be suitable in the past.

3.3.2 IMDSBN Model

We often are interested in knowing the probability that P(X|Y). Let X denote hypopaper and Let Y denote evidence. The Bayesian method must then be used to

know P(Y|X) in order to find P(X|Y), because it is easier to model hypopaper leading to evidence.

$$P(X \mid Y) = \frac{P(Y \mid X) \cdot P(X)}{P(Y)} \tag{2}$$

$$P(X \mid Y) = \frac{P(Y \mid X) \cdot P(X)}{\sum P(Y \mid X') \cdot P(X')} \tag{3}$$

The rules in Bayesian terms are as follows: P(X) is the prior probability of X, P(X|Y) is the posterior probability of X given E, P(Y|X') is the sample probability, and P(Y|X) is the likelihood of Y given H.

The selected parameters used to form the BN factor are defined below.

- **AF:** File access frequency, more or less?
- **Region**: Sites in the same location?
- **CP:** Computing performance high or low?
- **Bandwidth:** WAN bandwidth high or low?
- **Replica:** The replica will be moved to another site?

The important task for the IDMSBN is determining parameter probabilities. The initial assumption is that parameters are independent of one another. *AF* is defined as the file being requested frequently in a specific time interval. *Region* is the probability of the requesting and source sites being in the same region. *CP* is the probability of the site from which a file is acquired having high computing performance.

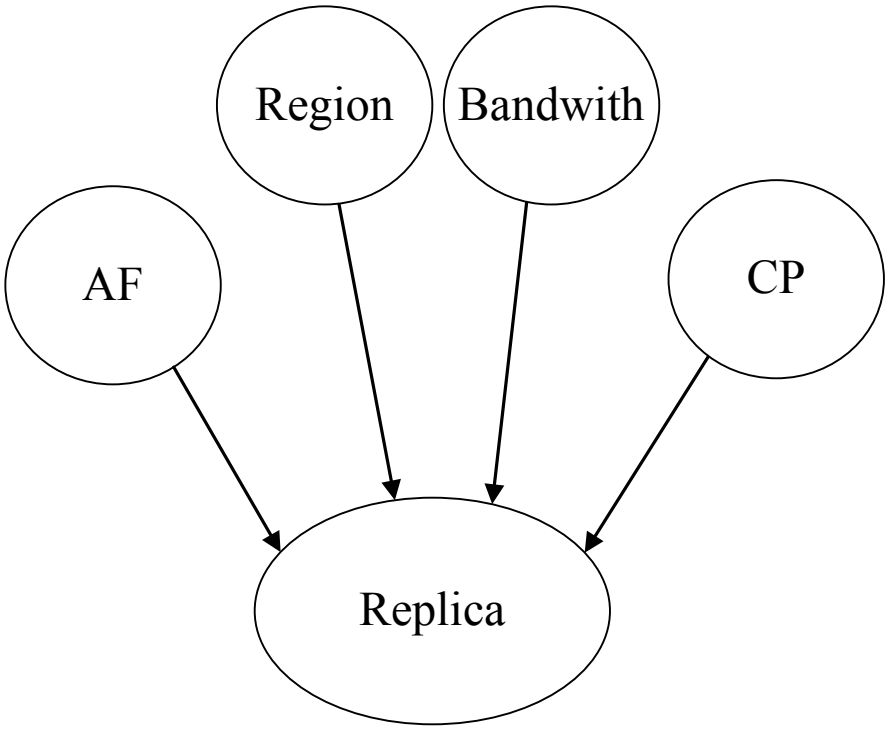


Figure 3-5. The Bayesian network structure

We assume domain experience will simplify the many parameters shown in Figure 3-5. Initially, some probabilities may be assigned to parameters, or some sample training used to build proficiency. User-set threshold values can be used to determine whether or not replicas are created.

The independent parameters have different states that may be shown in conditional probability tables (CPT) describing relations between parameter states at some time in the past. Table 3-1, Table 3-2, Table 3-3 and Table 3-4 below show parameter probability tables. Table 3-5 is a CPT for replication given AF, Region, Local, and CP. Probabilities are usually assigned according to expert experience, but in this thesis we assume the values. Finally, replica parameters, meaning the chances of files being replicated, may obtained by comparing replication values from prior parameters with threshold values set by users.

Table 3-1. Probability of AF

AF	AF=High	AF=Low
p(AF)	0.1	0.9

Table 3-2. Probability of Region

Region	Region=Local	Region=Non_local
p(Region)	0..25	0.75

Table 3-3. Probability of Bandwidth

Bandwidth	Bandwidth=High	Bandwidth=Low
p(AF)	0.25	0.75

Table 3-4. Probability of CP

CP	CP=High	CP=Low
p(CP)	0.2	0.8

Table 3-5. Conditional Probability of replication given AF, Region Bandwidth and CP

AF	Region	Bandwidth	CP	Replica=Yes	Replica=No
High	Local	High	High	0.8	0.2
High	Local	High	Low	0.7	0.3
High	Local	Low	High	0.7	0.3
High	Local	Low	Low	0.5	0.5
High	Remote	High	High	0.9	0.1
High	Remote	High	Low	0.8	0.2
High	Remote	Low	High	0.7	0.3
High	Remote	Low	Low	0.6	0.4
Low	Local	High	High	0.4	0.6
Low	Local	High	Low	0.3	0.7
Low	Local	Low	High	0.2	0.8
Low	Local	Low	Low	0.45	0.55
Low	Remote	High	High	0.25	0.75
Low	Remote	High	Low	0.2	0.8
Low	Remote	Low	High	0..2	0.8
Low	Remote	Low	Low	0..1	0.9

3.4 The Algorithm

Below, we describe our IDMSBN algorithm. First, we get weight and probability for the parameters used from domain experience or training samples, and use them to infer the probability of replication happening during file operations. After sufficient time has passed for the algorithm to get the latest metadata (the parameter probabilities and CPT), it calculates a file replication probability value. DB metadata changes over time, so the file replication threshold value is different when calculating the probability value using BN historical learning. As figure 3-6 shows, if the probability value exceeds the user–defined threshold, the IDMSBN can duplicate the same file to the next best site without replication. And every storage element maintains free space to save larger files by deleting other files. When a site has no free space, files on the site that haven't been used for a long time are deleted to liberate hard disk space. But no matter how many files are deleted, at least one replica always exists on another site.

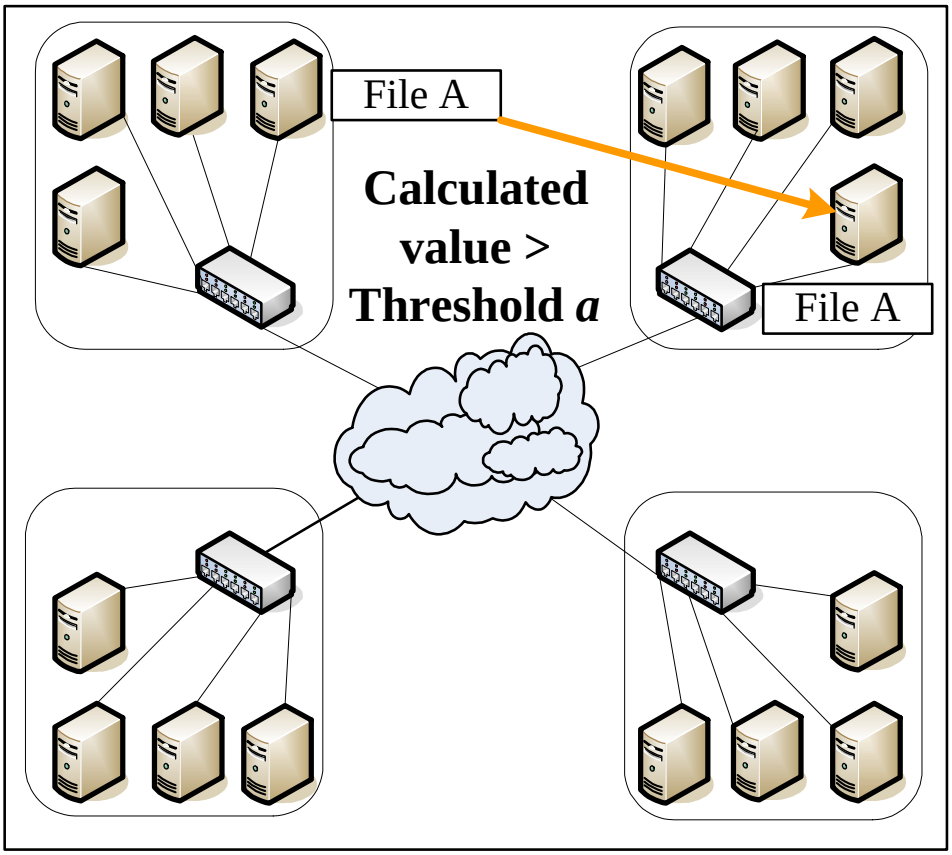


Figure 3-6. Example of replication

Chapter 4
Experimental Results

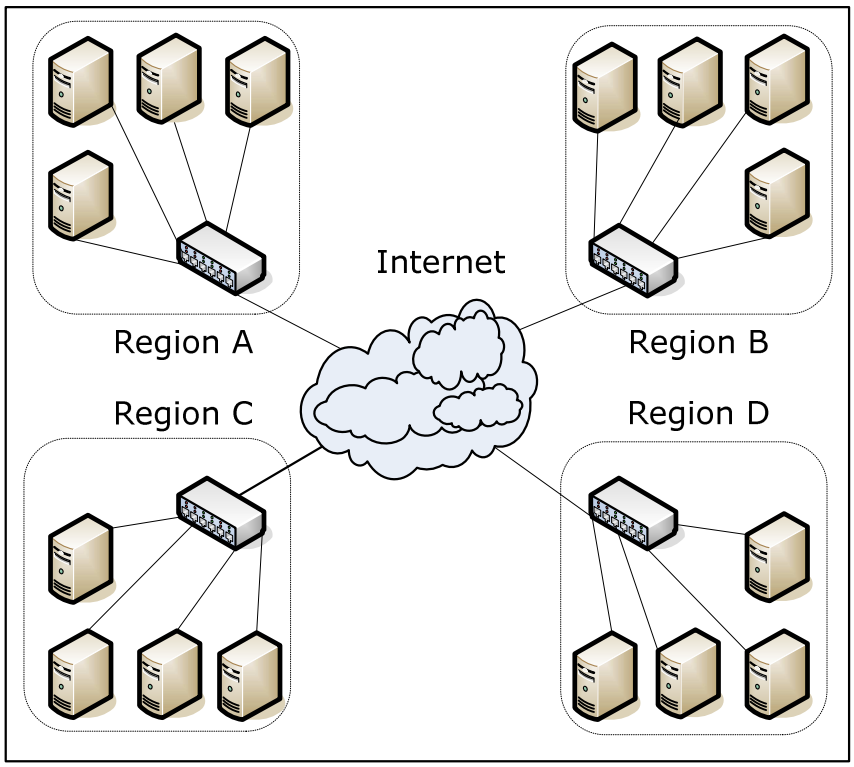
4.1 Experimental Environment

In this chapter, we compare the performance of the IDMSBN algorithm with that of four other replication strategies, Least Frequency Used (LFU), Least Recently Used (LRU), and the simple Accessed Remotely (AR). Strategy. “AR” mean sites have no replicas and all files are accessed remotely. The LFU and LRU strategies always replicate when requests occur, but differ in which files are chosen for deletion when there is not enough storage element free space for replication. LRU chooses the oldest files and LFU chooses the least frequently requested files.

Optorsim simulation was used to measure the performance of the replication strategies. The NETICA API, written in Java, was used in the BN Algorithm.

4.2 Parameter Setting

Figure 4-1 shows the experimental environment containing 16 nodes. The computing and storage elements were in same node.



\

Figure 4-1. Experimental environment

The important parameters used with Optorsim are shown in Table 4-1. The storage disk capacity was set to 200GB initially in the nodes with storage elements. Intranet bandwidth in the regions was 100Mbps and internet bandwidth was 1000 Mbps. Other parameters related to job scheduling including number of jobs, number of job types, number of files in each job, size of each file, and job delay. Number of jobs means the number of jobs submitted during the simulation run. There were 500 jobs in all, with 30 different job types containing fifteen 250MB-4000MB files. Job delay is the interval for submitting jobs in simulated execution.

Table 4-1. Parameters are used in the simulation

Parameters	Values
Node storage disk	**200GB**
Intranet bandwidth	**1000Mbps**
Internet bandwidth	**100Mbps**
Number of jobs	**500**
Number of job types	**30**
File number of each job	**15**
Size of each file	**250-4000MB**
Job delay	**2500ms**

The configuration file contains information about simulated jobs with unique logical file names (LFN) selected for simulation input. Jobs need to access files during execution, and certain file access patterns can be used for job submission: sequential (files are requested in the order stated in the job configuration file), random (files are requested using a uniform random distribution), random walk unitary (files are request in one direction away from the previous file request) and random walk Gaussian (files are accessed in a Gaussian distribution) These file access patterns are shown in Figures Figure .4-2, Figure 4-3, Figure 4-4, and Figure 4-5.

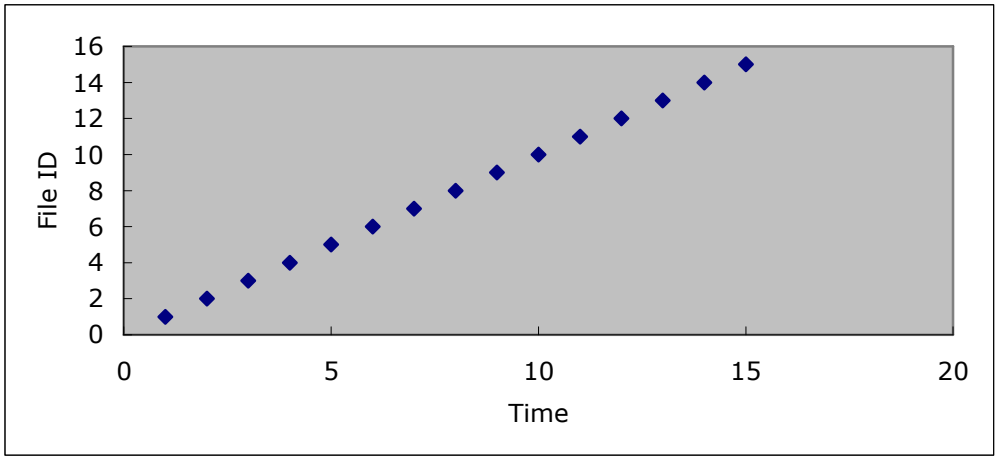


Figure 4-2. Sequential

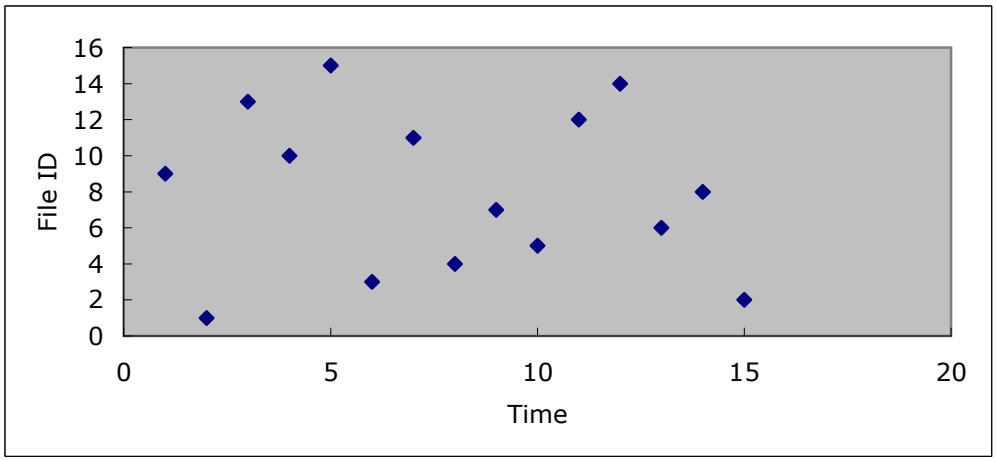


Figure 4-3. Random

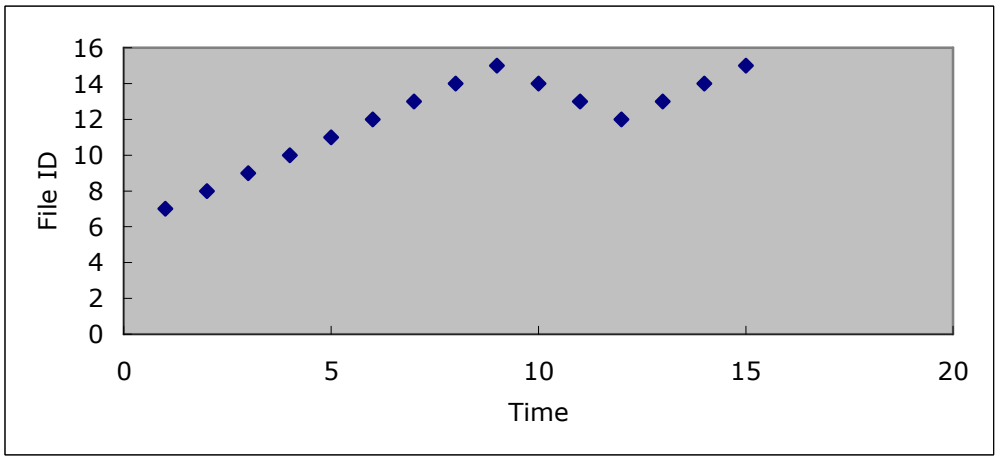


Figure 4-4. Random Walk Unitary

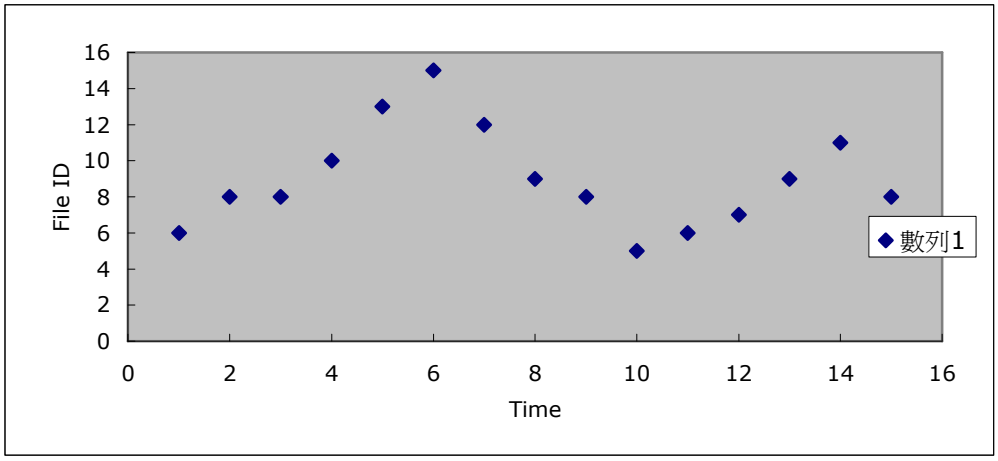


Figure 4-5. Random Walk Gaussian

4.3 Evaluation Metrics

The three evaluation metrics used for the simulation are shown below.

- Total job time
- Effective network usage (ENU)
- Computing elements active (CE)

Total job time is the simulation time from beginning to completion. It is used to measure the performance of various strategies in seconds. ENU is the mean ratio between file requests that use network resources and total file requests, effectively, the ratio of files transferred to files requested. Generally speaking, a low ENU value indicates that the strategy used is good at putting files in the right places. In the equation below, $N_{remote\_file\_accesses}$ is how many times a local site read a file from another site, $N_{file\_replications}$ is how many file replications occurred, and $N_{all\_file\_accesses}$ is the total number of files requested by a local site.

$$\mathrm{ENU} = \frac{N_{\text{remote\_file\_accesses}} + N_{\text{file\_replications}}}{N_{\text{all\_file\_accesses}}} \tag{4}$$

CE usage means the percentage of time the computing elements ran jobs or were otherwise active (the average CE usage for each site).

4.4 Results

As mentioned above, we set the parameters to get the simulation results shown in Figure 4-6, Figure 4-7 and Figure 4-8. The IDMSBN threshold value was set at 0.3.

Total IMDSBN job times may be better for certain file access patterns because some files are popular with users on other sites. If more replicas exist, it can reduce total job wait times. But when IDMSBF adds file replicas, the ENU value increases a little with each replication. High effective network usage is not good in busy network environments, because network bandwidth must be shared with other sites.

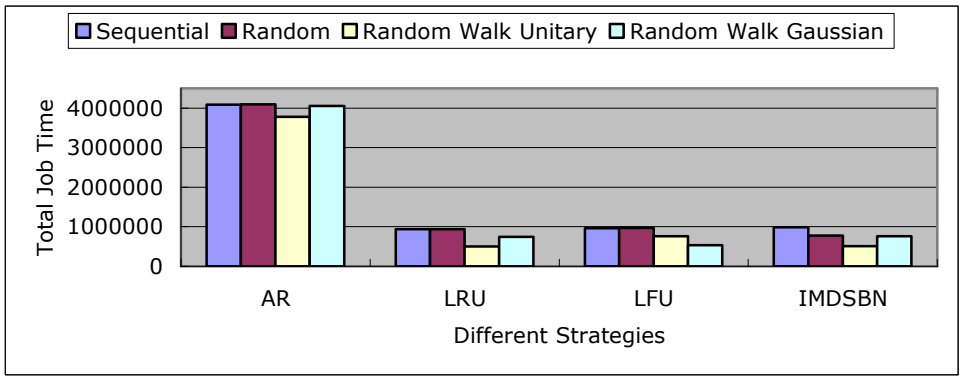


Figure 4-6. Performance comparison for four strategies

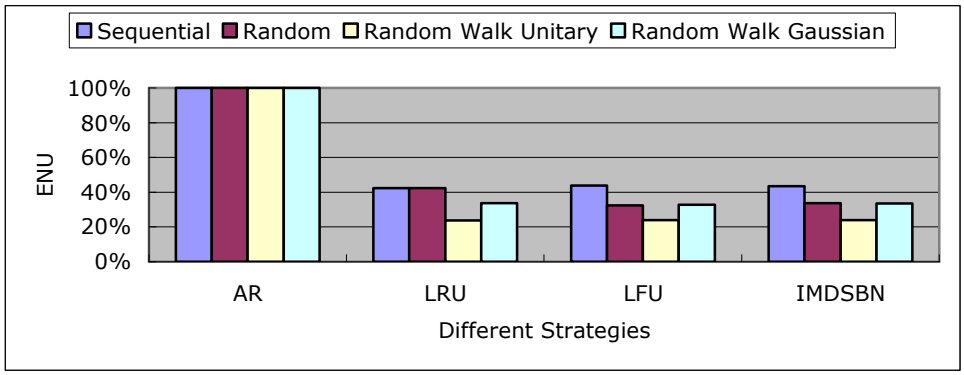


Figure 4-7. ENU comparison for four strategies

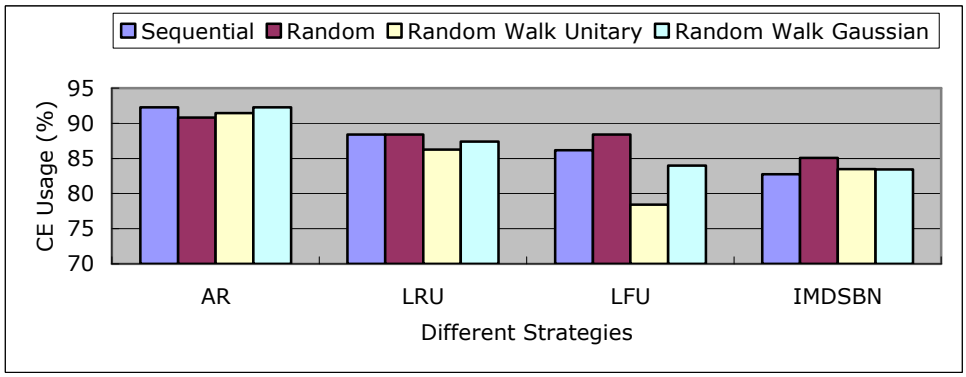


Figure 4-8. CE Usage comparison for four strategies

Total job time and replication number is tradeoff in four kind of threshold shown in Figure 4-9 and Figure 4-10.

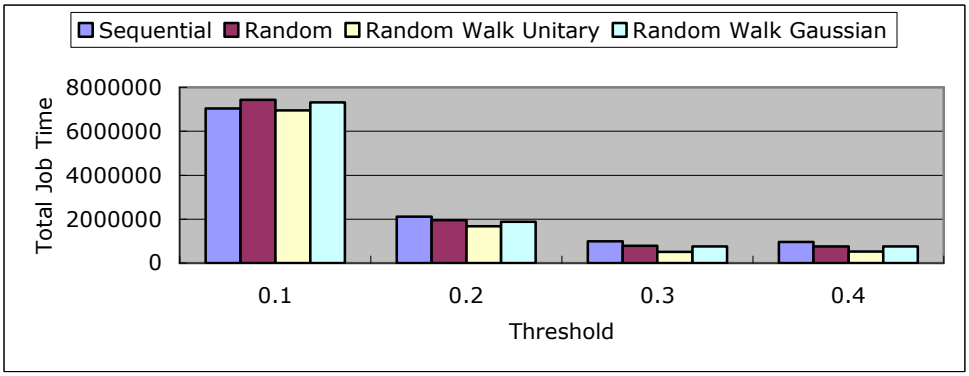


Figure 4-9. Performance comparison for four Threshold

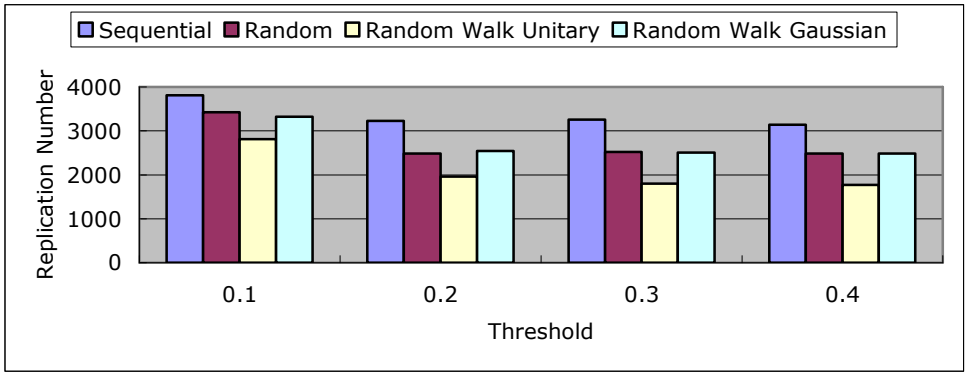


Figure 4-10. Replication Number comparison for four Threshold

Chapter 5
Conclusion and Future Work

This thesis presents a dynamic maintenance service called IDMSBN for improving grid environment performance in some cases. It is aimed at simplifying the problem of the DMS incurring metadata overloads. It also improves grid system performance in certain circumstances. In general, IDMSBN provided a different view of replication maintenance parameters.

In the future, we will try to provide IDMSBN parameters a hierarchical structure and try to find other approaches to parameters. We will continue enhance the reliability for various data, and we will train for different data styles to find appropriate conditional probability tables.

Bibliography

[1] B. Allcock, J. Bester, J. Bresnahan, A. Chervenak, I. Foster, C. Kesselman, S. Meder, V. Nefedova, D. Quesnel, and S. Tuecke, "Data Management and Transfer in High-Performance Computational Grid Environments," *Parallel Computing*, 28(5):749-771, May 2002.

[2] B. Allcock, J. Bester, J. Bresnahan, A. Chervenak, I. Foster, C. Kesselman, S. Meder, V. Nefedova, D. Quesnel, and S. Tuecke, "Secure, efficient Data Transport and Replica Management for High-Performance Data-Intensive Computing," *Proceedings of the Eighteenth IEEE Symposium on Mass Storage Systems and Technologies*, pp. 13-28, 2001.

[3] CERN - http://public.web.cern.ch/Public/Welcome.html

[4] R. S. Chang and J. S. Chang, "Adaptable Replica Consistency Service for Data Grids", *Proceeding of The third International conference of Information Technology (ITNG'06)*, page 646-651, 2006.

[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," *Journal of Network and Computer Applications*, 23:187-200, 2001.

[6] K. Czajkowski, S. Fitzgerald, I. Foster, and C. Kesselman, "Grid Information Services for Distributed Resource Sharing," *Proceedings of the Tenth IEEE International Symposium on High-Performance Distributed Computing (HPDC-10'01)*, 181-194, August 2001.

[7] J. Fathali, "A genetic Algorithm for the p-median problem with pos/neg weights", A*pplied Mathematics and Computation*, 8 August 2006 (to appear)

[8] M.L. Fisher, "The Lagrangian relaxation method for solving integer programming problems", *Management Science*, 27, pp. 1-18.

[9] I. Foster, "The Grid: A New Infrastructure for 21st Century Science", *Physics Today*, 55(2):42-47, 2002.

[10] I. Foster and C. Kesselman, "*The Grid 2: Blueprint for a New Computing Infrastructure (Elsevier Series in Grid Computing), Morgan Kaufmann, 2nd edition*," 1999.

[11] I. Foster and C. Kesselman, "Globus: A Metacomputing Infrastructure Toolkit," *International Journal of Supercomputer Applications and High Performance Computing*, 11(2), pp. 115-128, 1997.

[12] L. E. Jackson , G. N. Rouskas, Matthias F.M. Stallmann, "The directional p-median problem: Definition, complexity, and algorithms", *European Journal of Operational Research*, 2006 (to appear)

[13] Java CoG - http://www-unix.globus.org/cog/

[14] NWS - http://nws.cs.ucsb.edu/

[15] OptorSim – A Replica Optimizer Simulation: http://edg-wp2.web.cern.ch/edg-wp2/optimization/optorsim.html

[16] S. M. Park, J. H. Kim and Y. B. Ko," Dynamic Data Grid Replication Strategy based on Internet Hierarchy", *The second International Workshop on Grid and cooperative Computing (GCC2003)*, page 838-846, 2003.

[17] S. M. Park and J. H. Kim, "Chameleon: A Resource Scheduler in A Data Grid Environment," *Proceedings of Third Internatinal Symposium on Cluster Computing and the Grid*, 2003.

[18] R. M. Rahman, K. Barker, and R. Alhajj, "Replica Placement Design with Static Optimality and Dynamic Maintainability", *Proceedings of the Sixth IEEE International Symposium on Cluster Computing and the Grid (CCGRID'06)*, pp. 434-437, 2006.

[19] R. M. Rahman, K. Barker, and R. Alhajj, "Effective Dynamic Replica Maintenance Algorithm for the Grid Environment", *Proceeding of Advances in Grid and Pervasive Computing Vol. 3947, on Grid and Pervasive Computing 2006 (GPC2006)*, pp. 336-345, 2006.

[20] K. Ranganathan and I. Foster, "Design and evaluation of dynamic replication strategies for a high performance data Grid," *Proceeding of International Conference on Computing in High Energy and Nuclear Physics*

[21] X. H. Shi, H. Jin, W. Z. Qiang, and D. Q. Zou, "An Adaptive Meta-scheduler for Data-Intensive Applications", *Proceeding of Grid and Cooperative Computing (GCC'03)*, page 830-837, 2003.

[22] H. Stockinger, A. Samar, B. Allcock, I. Foster, K. Holtman, and B. Tierney, “File and Object Replication in Data Grids,” *Journal of Cluster Computing*, 5(3):305-314, 2002.

[23] The Grid Project – http://www.globus.org/

[24] S. Vazhkudai, S. Tuecke, I. Foster, “Replica Selection in the Globus Data Grid,” *Proceedings of the 1st International Symposium on Cluster Computing and the Grid* (*CCGRID 2001*), pp. 106-113, May 2001.

[25] S. Venugopal, R. Buyya, and K. Ramamohanarao, “A Taxonomy of Data Grids for Distributed Data Sharing, Management, and Processing,” *ACM Computing Surveys, Vol.38 Article 3*, March 2006.

[26] C. T. Yang, I. H. Yang, K. C. Li and S. Y. Wang, "Improvements on Dynamic Adjustment Mechanism in Co-Allocation Data Grid Environments," *accepted and to appear in The Journal of Supercomputing*, Springer, vol. 40, no. 3, pp. 269-280, June 2007.

[27] C. T. Yang, S. Y. Wang, and C. P. Fu, “A Dynamic Adjustment Mechanism for Data Transfer in Data Grids,” *accepted and to appear in the Proceeding of Network and Parallel Computing (NPC2006)*, October 2006.

[28] C. T. Yang, S. Y. Wang, C. H. Lin, M. H. Lee, and T. Y. Wu, “Cyber-Transformer: A Toolkit for Files Transfer with Replica Management in Data Grid Environments,” *Proceedings of the Second Workshop on Grid Technologies and Applications (WoGTA’05)*, pp. 73-80, December 2005.

[29] C. T. Yang, I. H. Yang, C. H. Chen and S. Y. Wang, "Implementation of a Dynamic Adjustment Mechanism with Efficient Replica Selection in Co-Allocation Data Grid Environments," *Proceedings of the 21st Annual ACM Symposium on Applied Computing (SAC 2006) - Distributed Systems and Grid Computing (DSGC) Track*, vol. 1, pp. 797-804, Dijon, France, April 23-27, 2006.

[30] C. T. Yang, I. H. Yang, K. C. Li and C. H. Hsu, "A Recursive-Adjustment Co-Allocation Scheme in Data Grid Environments," *Distributed and Parallel Computing: 6th International Conference on Algorithms and Architectures for Parallel Processing, ICA3PP 2005, Lecture Notes in Computer Science*, vol. 3719, pp. 40-49, Springer, October 2005.

[31] C. T. Yang, C. P. Fu, C. J. Huang, “A dynamic file replication strategy in data grids,” *TENCON 2007, Procceding of IEEE Region 10 Conference*, pp. 1-5, Oct. 30-Nov. 2 2007.

[32] R. M. Rahman, K. Barker, R. Alhajj "A Predictive Technique for Replica Selection in Grid Environment," *Cluster Computing and the Grid: Seventh IEEE International Symposium on CCGRID 2007*, pp. 163 – 170, May 2007.

[33] S.Vazhkudai, J. Schopf. "Using Regression Techniques to Predict Large Data Transfers," *The International Journal of High Performance Computing APPLications (IJHPCA), special issue on Computing: Infrastructure and Applications,* August 2003.

[34] M. M. Wagner, A. W. Moore and R. M. Aryel, "*Handbook of Biosurveillance," chaper 18 Bayesian Methods for Diagnosing outbreaks*, pp. 273-287, Elsevier, 2006.

[35] J. Ding, N. Jiang, X. Li; B. Kramer, F. Davoli, Y. Bai, "Construction of Simulation for Probabilistic Inference in Uncertain and Dynamic Networks Based on Bayesian Networks," *6th International Conference on Telecommunications Proceedings, ITS 2006,* pp. 983 – 986, June 2006.

[36] N. Cruz-Ramirez, L. Nava-Fernandez, H.G.A.Mesa, E.B. Martinez, J.E. Rojas-Marcial, "A parsimonious constraint-based algorithm to induce Bayesian network structures from data," *Sixth Mexican International Conference on Computer Science, ENC 2005*, pp. 306 – 313, -30 Sept. 2005.

[37] Q. eng, M. Mng, W. Wu, R. Wang, "Variant Bayesian Networks," *Sixth IEEE International Conference on Data Mining Workshops, ICDM Workshops 2006*, pp. 258 – 262, Dec. 2006.

[38] K.W Przytula,D. Thompson, "Construction of Bayesian networks for diagnostics," *IEEE Aerospace Conference Proceedings*, vol.5 , pp.193 - 200, 18-25 March, 2000.

[39] Netica API -http://www.norsys.com/netica.html