IACSIT International Journal of Engineering and Technology, Vol.2, No.4, August 2010
ISSN: 1793-8236

Dynamic Replica Management for Data Grid

K. Sashi and Dr. Antony Selvadoss Thanamani

Abstract—**Data grids provide distributed resources for dealing with large scale applications that generate huge volume of data sets. Data replication, a technique much discussed by data grid researchers in past years creates multiple copies of file and stores them in conventional locations to shorten file access times. One of the challenges in data replication is creation of replicas, replica placement and replica selection. Dynamic creation of replicas in a suitable site by data replication strategy can increase the systems performance. When creating replicas a decision has to be made on when to create replicas and which one to be created. This decision is based on popularity of file. Placement of replicas selects the best site where replicas should be placed. Placing the replicas in the appropriate site reduces the bandwidth consumption and reduces the job execution time. Replica selection decides which replica to locate among many replicas. This paper discusses about dynamic creation of replicas, replica placement and replica selection. It is implemented by using a data grid simulator, Optorsim developed by European data grid projects.**

Index Terms—**Grid Computing, Data grid, Replica Creation, Replica Placement, Replica Selection**

I. Introduction

A Grid is large scale resource sharing and problem solving mechanism in virtual organizations [1]. Grid Computing is emerging as a key enabling infrastructure for a wide range of disciplines in science and engineering, including high energy physics, molecular biology, astronomy and earth sciences. Grid computing has the potential to support different kinds of applications. They include compute-intensive applications, data-intensive applications and applications requiring distributed services. Various types of Grids have been developed to support these applications and are categorized as Computational Grids, Data Grids and Service Grids.

Data grids are predicted to be the solution to the large computational power and data storage requirements of many current projects. The emerging trend in scientific applications in many areas such as high energy physics, data mining, and climate simulation shows that these applications process and produce large amounts of data [2-6]. The resulting output data in turn to be stored for further analysis and shared with collaborating researchers within the scientific community who are spread around the world. Managing this data in a centralized location increases the data access time and hence much time is taken to execute the job. So to reduce the data access time replication is used.

This paper presents a dynamic replica creation and placement algorithm which can automatically maintain the data by the status of data. Replica creation decides which file to be replicated and how many replications to be created. Replica placement deals with the placement of replicas in appropriate locations. Two factors are considered for the placement of replicas, the response time and the bandwidth.

The rest of the paper is organized as follows. Section II describes the related work in replica management. Section III discusses the dynamic replica creation and placement algorithm. Experimental results are shown in Section IV. Finally conclusion and future work is given.

II. Related Work

Kavitha *et al*. [7] proposed a strategy for creating replicas automatically in a generic decentralized peer-to peer network. The goal of their model is to maintain replica availability with some probabilistic measure. Ranganathan and Foster [8] discuss various replication strategies. All these replication strategies are tested on hierarchical Grid Architecture.

Several research efforts [9] assume user requests as the only parameter considered for replica placement so network latencies are ignored. However, network bandwidth plays a vital role in file transfer. Transfer Time can be saved by placing replica of a file at a site that is connected to its neighbors with limited bandwidth and if its request for that file is above average. The authors [10, 11] adapt the economic model for replication strategies. They focus on optimizing the replication of data in grid environment to achieve the final goal which is reducing the job turnaround time in long term.

Ruay-Shiung Chang, Hui-Ping Chang, Yun-Ting Wang [23] proposed an algorithm called LALW where the replicas are created dynamically based on the weights. Here the replica placement is done only in the cluster level and not in the site level.

Ming Tang, Bu-Sung Lee, Chai-Kiat Yeo, Xueyan Tang [12] suggests two replication algorithms Simple Bottom Up (SBU) and Aggregate Bottom Up (ABU) for Multi-tier data grids. To minimize the data access time and the network load, replicas of the data should be created and spread from the root center to regional centers, or even to national centers. The strategies in this work are applicable only to multi-tiered grids.

Lin [13] focuses on the important problem of choosing locations for placing replicas in data grids. They address the problem of data replica placement in data grids, given the traffic pattern and locality requirements.

In paper [14] the authors addresses the system wide data availability problem assuming limited replica storage. In this two new metrics are discussed: System File Missing Rate (SFMR) and System Byte Missing Rate (SBMR)..

Sang Min Park, Tai Hoon Kim [15] proposed an algorithm called Bandwidth Hierarchy Replication (BHR) which reduces data access time by avoiding network congestions in a data grid network. In [16] the authors proposed the BHR algorithm by using three level hierarchical structures. They address the problem of both scheduling and replication.

In [17] the author discusses about replica placement policy and replica selection, which all are embedded in their

proposed system in order to reduce job turnaround time, reduce storage cost, and reduce network bandwidth consumption. Their system is termed as Replica Management in Grid (RmGrid).

Mohammed Rashedur Rahman [18] uses P-median model for the replica placement problem. P-median model finds the locations of P candidate sites to place a replica that optimize the aggregated response time.

A different cost model was proposed by Lamehamedi et al. [19] to decide the dynamic replication. This model evaluates the data access gains by creating a replica and the costs of creation and maintenance for the replica, and it is applied by the Replica Manager in each intermediate storage site in a decentralized manner.

Rahman et al [24] considered the storage access latency while selecting the best replica.

III. Dynamic Replica Creation, Placement and Selection Algorithm(DRCPS)

A. The Framework

In this paper, a region based framework is proposed. The design of the framework is based on the centralized data replication management. Fig – 1 shows the framework. There is a region master responsible for triggering the replica management. Group of sites are located on the same network region. A network region is a network topological space where sites are located closely. Each region has a region header. It is used to manage the site information in a region. The region master collects all information about accessed files from all region headers. Each site maintains the history of files accessed in that site. The access history gives the details of fileid, regionid and the time, which indicates that the file has been accessed by a site located in the region (regionid) at a particular time. At a regular time interval each site sends its access history to its region header. The region header maintains the detail of the number of times the file has been accessed in the particular region. Region header sends information to the region master at a constant time interval. The Region master maintains the global information regarding the file and the number of time it has been accessed. The selection of the popular file is based on the weights given to the file.

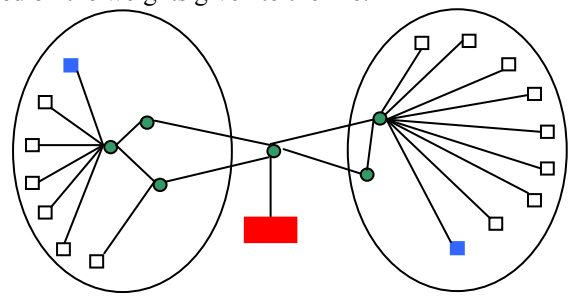

Figure 1. Replication Framework

B. Proposed Methodology

1 .Dynamic Replica Creation Algorithm

In [23] the authors proposed an algorithm called LALW where the replicas are created dynamically based on the weights and placed in the cluster. This paper focuses on placing the replicas in best site within the cluster. Replica creation and Replica placement within the cluster is based on LALW.

Replica creation involves two stages. First stage determines which file to be replicated and the second stage calculates the number of copies to be replicated.

In the first stage region master gets the information from the region header at a constant time interval. Information gathered at different time interval has different weights. Here the weight represents the quantity and the time interval. Recently accessed files have larger weight and older files have less weight. Access Frequency (AF)[23] is calculated to exhibit the importance for access history in different time intervals. Let N_T be the number of time interval passed, F is the set of files that have been requested and a_{ij} indicates the number of accesses for the file i at time interval j. The Access Frequency for file F is represented as

$$AF(f) = \sum_{t=1}^{N_T} (a^t{}_f \times 2^{-(N_T - t)}), \forall f \in F \quad \ldots\ldots\ldots\ldots\ldots\ldots (1)$$

By calculating the Access Frequency the popular file can be identified.

The second stage is to calculate the number of replicas needed. This can be done by comparing the average AF per time interval for the popular file with all other files in F. Let the popular file be p. Then the average Access Frequency of the popular file p [23] is

$$AF_{avg}(p) = \frac{AF(p)}{N_T} \quad \ldots\ldots\ldots\ldots\ldots\ldots (2)$$

The average Access Frequency[23] of all other files can be calculated as

$$AF_{avg}(f) = \frac{[AF(f)]_{sum}}{N_F \times N_T}, \forall f \in F \quad \ldots\ldots\ldots\ldots\ldots\ldots (3)$$

$AF(f)_{sum}$ [23] indicates the sum of AF for all files requested. The number of replicas needed for the popular file is calculated as

$$Num_{system}(p) = \left[\frac{AF_{avg}(p)}{AF_{avg}(f)} \right] \quad \ldots\ldots\ldots\ldots\ldots\ldots (4)$$

2. Replica Placement Algorithm

Replica has to be placed in the appropriate location so as to keep the environment in the higher performance. Replica placement has two stages. The former determines in which region header the replica has to be placed and how many replica has to be placed. The later stage determines in which site within the region the file has to be placed.

In the first stage the region master will make a request to every region header in order to get the information about the popular file. The information contains the region details and

TABLE I. General Configuration of Parameters

Parameters	Values
Number of Jobs	100
Number of Job Types	5
Number of file accessed per job	10

Size of single file	1GB
Total Size of files	100 GB
Job Delay	2500 mS
Maximum Queue Size	200

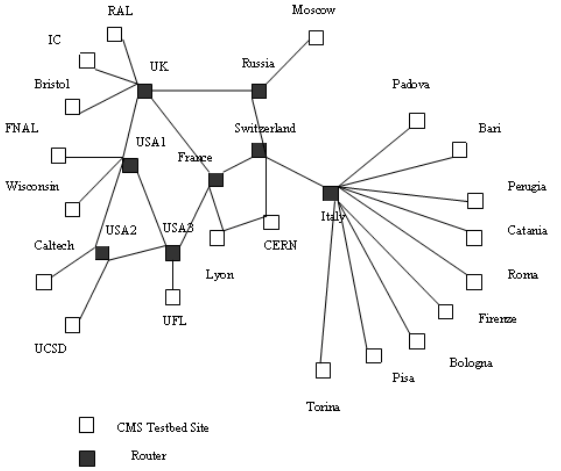

Figure 3. CMS Data Challenge 2003 grid topology

the number field. These files still have different weights according to different time intervals. The Region master calculates AF(p) in different regions after getting the information for popular file from region headers. Then it sorts its records in descending order based on the AF(p). The region in the first item of the sorted list has the highest priority to place the replicas. Number of replicas to be placed at region c [23] is calculated as

$$Num_c(p)\left[n \times \frac{AF_c(p)}{[AF_c(p)]_{sum}}\right], = c = 1,2,...,N \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots(5)$$

The replicas are replicated in region 1, 2,. . N in turn until all the replicas is allocated.

The second stage is to place the replica in the appropriate site within the region. To place the replicas placement cost is calculated. Two factors considered here are the number of request made for the particular file and the response time, where the response time includes the transfer time divided by the network bandwidth.

Then the Placement cost is calculated as

*Placement cost = Number of Request * Response Time*

Replica can be placed in the site where the placement cost is low.

If there is no space to store the replica Least Frequently Used file is deleted from the site.

3. Replica Selection Algorithm

Replica selection is one of the key components of data management in Data intensive application. It decides which replica location is the best for the grid users. Several replicas can exist for a given file. The optimization algorithm determines the replica that should be accessed from a given location. Replica selection is complicated because it can involve several components including network, CPU and disks. Selection procedure is based on whether to select the replica from the master site or from the region header or from any neighboring site. The decision is based on the Weighted Euclidean Distance

$$D(x,y)=SQRT(\sum_{i=1}^{m}(x_i-y_i)^2 * w_i)$$

Where w_i is the weight of each attribute.

IV. Experimental Results

A .Simulation Setup and Metrics

OptorSim is used to evaluate the performance of this algorithm. OptorSim was developed by European Data Grid projects [20] and is written in Java. It provides a framework to simulate the real grid environment. It is developed to test the Dynamic Replication strategies.

In Data Grid Environment various job execution scenarios are present. The Job Execution scenario used for this algorithm is shown in Table – I.

The architecture used here is the European Data Grid CMS testbed architecture [22]. In this there are twenty sites in which two of them have only storage element and which acts as master node. CERN and FNAL are considered as master site where data is produced initially. Jobs are processed in the remaining sites which have both storage and computing element. There are 8 routers which is used to forward request to other sites. The topology of CMS testbed is depicted in Fig – 3. Data replication strategies commonly assume that the data is read only in Data Grid Environments [21].

The Performance Evaluation Metrics used in this system are Mean Job Execution Time, Average Storage Used and the Effective Network Usage.

The *Mean Job Execution Time* is defined as the total time to execute all the jobs, divided by the number of jobs completed.

Storage usage can be calculated for each site as a percentage of capacity reserved by files according to the total capacity for the underlying storage.

Replicating a file takes time and uses network bandwidth. *Effective Network Usage* is the ratio of files transferred to files requested, so a low value indicates that the optimization strategy used is better at placing files in the right places.

TABLE –II OPTORSIM RESULTS

	No Replication	LRU	LFU	DRCPS
Mean Job Time of all Jobs in MS	5990	1022	971	432
Total Number of Replications	0	193	172	129
Total Number of Local File Accesses	0	1505	1364	1016
Total Number of Remote File Accesses	1348	0	0	0
Percentage of Storage Filled/Available	8. 81	26.12	26.56	29.53
Probability of Effective Network Usage	1. 0	0. 13	0. 14	0. 12

B .Simulation Results

The order in which a job requests files is determined by the job's access pattern. There are various access pattern generators used in OptorSim. In this work Zipf Access

Pattern Generator is used. Table – II shows the results for replication using access cost for current job and all queued jobs scheduling algorithm and Zipf access pattern generator.

When compared to No replication, LRU and LFU, DRCPS Algorithm performs better. This algorithm minimizes the Mean Job Execution time and thus the data access time can be improved. Fig – 4 depicts the Mean Job Execution Time.

Fig -5 depicts the average storage used. Average storage used is best in 'no replication', 'LFU' and 'LRU' because in the case of 'No replication' the data is stored only in one location where the files are produced initially and in LFU and LRU the files are replicated only when there is a need. But in the proposed algorithm the replica is created dynamically in advance. Fig – 6 shows the Effective Network Usage. It is better in the proposed algorithm when compared to other algorithms.

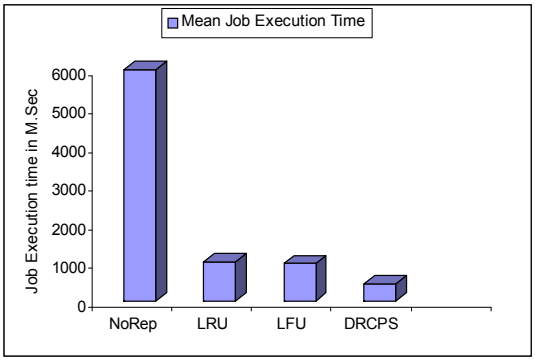

Figure 4. Mean Job Execution Time

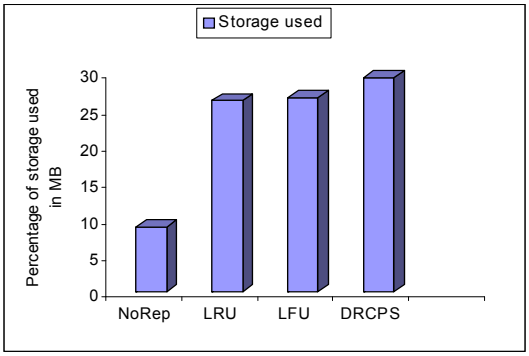

Figure 5. Average Storage Used

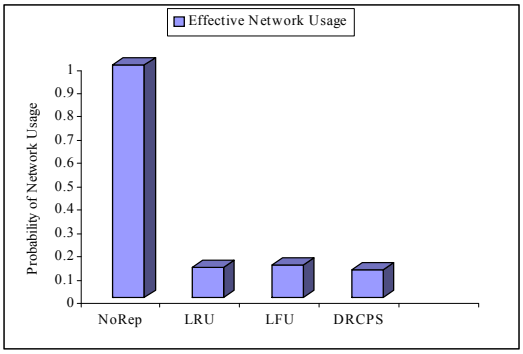

Figure 6. Effective Network Usage

V. Conclusion and Future Work

Grid computing is emerging as one of the best solution to utilize existing resources for catering the massive computational and data handling requirements of today's high performance applications. The data grid provides the ability to access and manage data and data resources on the grid. Replication involves the creation of identical copies of data files and their distribution over various grid sites needing access to these replicas. Replicas must be managed in terms of creation, and placement. In this paper a Dynamic Replica Creation, Placement and Selection algorithm is proposed. The proposed Algorithm increases the data availability by making dynamic replica creation. It also reduces the unnecessary replication. It places the replica in a appropriate location so as to reduce the placement cost. By using this algorithm Mean Job Execution time can be minimized, and there is an effective network usage.

This approach considers only the number of requests and response time for placing the replicas. As part of future work we plan to consider additional parameters for placing the replicas. Replica selection can also be extended by considering additional parameters such as security. Currently only LFU is applied as the replica replacement policy, so more replica replacement strategies can be investigated to complement the dynamic replacement algorithms to further improve the overall system performance

References

[1] I. Foster, C. Kesselman, S. Tuecke, The Anatomy of the Grid: Enabling Scalable Virtual Organizations, IJSA 2001.

[2] B. Allcock, J. Bester, J. Bresnahan, A. L. Chervenak, I. Foster, C. Kesselman, S. Meder, V. Nefedova, D. Quesnal, and S. Tuecke. Data management and transfer in high performance computational grid environments. Parallel Computing Journal, 28(3), May 2002.

[3] W. 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. In IEEE Mass Storage Conference, 2001.

[4] W. Allcock, I. Foster, V. Nefedova, A. Chervenak, E. Deelman, C. Kesselman, J. Lee, A. Sim, A. Shoshani, B. Drach, and D. Williams. High-performance remote access to climate simulation data: A challenge problem for data grid technologies. In Proceedings of SC 2001, November 2001.

[5] A. Chervenak, I. Foster, C. Kesselman, C. Salisbury, and S. Tuecke. The data grid: Towards architecture for the distributed management and analysis of large scientific datasets. Journal of Network and Computer Applications, 23:187.

[6] A. Chervenak, E. Deelman, I. Foster, L. Guy, W. Hoschek, A. Iamnitchi, C. Kesselman, P. Kunst, M. Ripeanu, B. Schwartzkopf, H. Stockinger, K. Stockinger, and B. Tierney. Giggle: A framework for constructing scalable replica location services. In *Proceedings of Supercomputing (SC2002)*, 2002.

[7] Kavitha, R., A. Iamnitchi, and I. Foster. Improving Data Availability through Dynamic Model Driven Replication in Large Peer-to- Peer Communities. Proceedings of Global and Peer-to-Peer Computing on Large Scale Distributed Systems Workshop, Berlin, Germany, May 2002.

[8] Ranganathan and I. Foster. Design and evaluation of Replication Strategies for a High Performance Data Grid, in Computing and High Energy and Nuclear Physics 2001 (CHEP'01) Conference.

[9] Bell, W., D. G. Cameron, L. Capozza, A., P. Millar, K. Stockinger, and F. Zini. OptorSim- A Grid Simulator for Studying Dynamic Data Replication Strategies. International Journal of High Performance Computing Applications, 17(4), 2003.

[10] W. H. Bell, D. G. Cameron, R. Carvajal-Schiaffino, A. P. Millar, K. Stockinger, and F. Zini. Evaluation of an Economy- Based File Replication Strategy for a Data Grid. In *Proc. Of 3rd IEEE Int. Symposium on Cluster Computing and the Grid (CCGrid'2003)*, May 2003. IEEE CS-Press.

[11] W. H. Bell, D. G. Cameron, R. Carvajal-Schiaffino, A. P. Millar, K. Stockinger, and F. Zini. Evaluating Scheduling and Replica Optimization Strategies in Data Grid. IEEE 2003.

[12] Tang, M., Lee, B., Tang, X., and Yeo, C. "The impact of data replication on job scheduling performance in the Data Grid". Future Generation Computing System 22, 3 (Feb. 2006).

[13] Lin, Y. F., P. Liu and J. J. Wu, Optimal placement of replicas with locality assurance. International conference on Parallel and Distributed Computing. 2006
[14] Ming Lei, Susan V. Vrbsky, Xiaoyan Hong , An on-line replication strategy to increase availability in data grids, Future Generation Computer Systems. 2007 (FGCS-Elsevier).
[15] Sang-Min Park, Jai-Hoon Kim, Young-Bae Ko, Won-Sik Yoon. Dynamic Data Replication Strategy based on Internet Hierarchy BHR , Springer-Verlag Heidelberg Volume 3033, 2004, pp. 838-846, 2004.
[16] A. Horri, R. Sepahvand, Gh. Dastghaibyfard , A Hierarchical Scheduling and replication strategy, International Journal of Computer Science and Network Security, Vol. 8, August 2008.
[17] Husni Hamad E, AL-Mistarihi and Chan Huah Yong Replica Management in Data Grid, International Journal of Computer Science and Network Security, 2008.
[18] Mohammed Rashedur Rahman, A dynamic Replica Placement Strategy in Grid Environment, University of Calgary, 2006.
[19] Lamehamedi H., Shentu Z., SzymanskiB, and Deelman E "Simulation of Dynamic Data replication Strategies in Data Grids, Proceedings of the 17th International Parallel and Distributed Processing Symposium, IEEE Computer Society 2003.
[20] D. G. Cameron, R. C. Schiaffino, J. Ferguson, P. Millar, C. Nicholson, K. Stockinger, and F. Zini, OptorSim v2. 0 Installations and User Guide, November 2004. http://cern.ch/edg-wp2/optimization/optorsim.htm
[21] W. Hoschek, F. J. Jaen-Martinez, A. Samar, H. Stockinger, K. Stockinger, Data management in an international data grid project, *Proceedings of the First IEEE/ACM International Workshop on Grid Computing(GRID '00)*, pages 77-90, Bangalore, India, December 2000.
[22] K. Holtman. CMS data grid system overview and requirement, Tech report CERN July 2001
[23] Ruay-Shiung Chang, Hui-Ping Chang, Yun-Ting Wang: A dynamic weighted data replication strategy in data grids. IEEE Conference.
[24] Rahman, R.M., K. Barker and R. Alhajj: Replica selection in Grid Environment: A Data-mining approach. Proceedings of the 2005ACM Symposium on applied computing, 2005.

K. Sashi is a Sr. .Lecturer in the Department of Information Technology at SNR Sons College, Coimbatore, India. She received her MCA degree at Bharathiar University in 2000 and M.Phil degree at Bharathidasan University in 2004. She is currently pursuing her Ph.D at Mother Teresa University, India. Her research interests include Grid Computing and current research focus on Dynamic Replications in Data Grid.

Dr. Antony Selvadoss Thanamani is presently working as Reader in the Dept of Computer Science, NGM College, India. He has published many papers in national/journals and written many books. His areas of interest include E-Learning, Data Mining, Networking, Parallel and Distributed Computing and etc. He has about 20 years of teaching experience.