VOL. 8, NO. 2, FEBRUARY 2013 ISSN 1819-6608
ARPN Journal of Engineering and Applied Sciences


www.arpnjournals.com

REPLICA REPLACEMENT ALGORITHM FOR DATA GRID ENVIRONMENT

K. Sashi and T. Santhanam
Department of Information Technology, SNR Sons College, Coimbatore, Tamil Nadu, India
Department of Computer Science, DG Vaishnav College, Chennai, India
E-Mail: sashi_sekar@yahoo.co.in

ABSTRACT

Grid computing is one of the fastest emerging technologies within the high performance computing environment. Grid deployments that require access to and processing of data are called data grids. They are optimized for data oriented operation. In a data grid environment, data replication is an effective way to improve data accessibility. However, due to limited storage, a replica replacement strategy is needed to make the dynamic replica management strategy efficient. In this paper, a dynamic replacement strategy is proposed for a region based network where a weight of each replica is calculated to make the decision for replacement. It is implemented by using a data grid simulator, Optorsim developed by European data grid projects.

Keywords: grid computing, data grid, replica replacement, storage.

1. INTRODUCTION

A Grid is a 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. 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. To speed up the data access for data grid systems data can be replicated in multiple locations, so that a user can access the data from the nearby locations. Data replications not only reduce data access costs, but also increase data availability in many applications. If the required files are replicated in some site in which the job is executed, job is able to process data without any communication delay. However, if required files are not in the site, they should be fetched from other sites and it usually takes very long time because size of single replica may reach giga-byte scale in some applications and network bandwidth between sites is limited. As a result, job execution time becomes very long due to delay of fetching replicas over Internet.

This paper presents a dynamic replica replacement algorithm. The factors to be considered for replica replacement are size, cost, bandwidth and prediction of future access of the particular replica.

The rest of the paper is organized as follows. Section 2 describes the related work in replica replacement. Section 3 discusses the dynamic replica replacement algorithm. Experimental results are shown in Section 4. Finally conclusion and future work is given.

2. RELATED WORK

The traditional replacement strategies are LRU and LFU which are used in Optorsim [2-4]. LRU (Least Recently Used) replaces the replica that has not been used for the longest period of time. LRU is based on temporal locality which means that the replica which has been referenced in the recent past is likely to be referenced again in the near future. However the algorithm does not considers the old replicas. But for scientific data even the old files are important.

In [5] the authors proposed an accurate model for evaluating various replacement policies and propose a new replacement algorithm referred to as Least Cost Beneficial based on K backward references (LCB-K). They use a utility function for determining which files need to be evicted from the cache.

Jose Aguilar *et al*., [6] proposed an adaptive cache coherence replacement scheme foe web proxy cache system that is based on several criteria about the system and applications, with the objective of optimizing the distributed cache system performance. The algorithm assigns a replacement priority value to each cache block according to a set of criteria to decide which block to remove.

In this paper [7] the authors introduced a replica replacement algorithm that combines prediction factors with replacement cost factors together. Through predicting

the popularity of replica in future time windows, hot spot replica is kept to improve mean job time. Cost factors are network latency, bandwidth, replica size and system reliability. A PC-based replacement algorithm is presented which achieves a balance between mean job time and bandwidth resource consumption.

The Replica replacement algorithm based on economic model and opportunity cost is proposed in [8]. The access history conforms to the Zipf like distribution and access time for every file in the future time window is calculated. Then the cost of transferring files is estimated. The file to be replaced is based on the weight of the replicas which is the product of the future access times and transfer cost of replica.

The authors in [9] proposed a colored Petri nets for modeling and performance analysis of grid architectures. The strategy is based on the automatic addition of clean up tasks to grid workflows. It helps to reduce the amount of storage space needed to execute a grid application. It covers all features important for performance analysis. It is fully adaptable and easily extendible. The clean up tasks doesn't take into account of the hot spot replication.

3. DYNAMIC REPLACEMENT ALGORITHM (DRA)

3.1. The framework

In this paper, a region based framework is proposed. The design of the framework is based on the centralized data replication management. Figure-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.

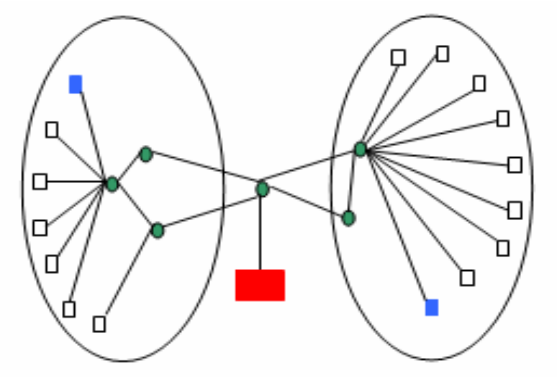

Figure-1. Replication framework.

3.2. DRA algorithm

In a data grid environment, data replication is an effective way to improve data accessibility. However due to storage constraints a replica replacement strategy is needed to make the dynamic replica management effective. In this work a dynamic replacement strategy is proposed for a domain based network where a weight of each replica is calculated to make the decision for replacement.

A traditional replacement strategy, LFU is based on the frequency with which a replica accessed. In this the less frequently accessed file is replaced. LFU - aging overcomes the disadvantage of LFU by considering both the frequency and age. However, both the algorithms are based on the historical statistical data, so it cannot predict the files which will be used frequently in the future. Random replacement algorithm randomly chooses one of the replicas in the local storage site. However, there is a possibility of removing the most frequently used replica.

Algorithm

Dynamic Replacement Algorithm predicts the popularity of the file and also considers the size of the replica with its bandwidth cost.

Algorithm

Step-1: Access Frequency of all files f is calculated as in [10] is represented as:

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

Let N_T be the number of time interval passed, F is the set of files that have been requested and a^t_f indicates the number of accesses for the file f at time interval t.

From the above the access frequency of less popular file is determined and let it be AF (lp).

Step-2: The cost of the replica to be replaced is calculated using the formula

$$Cost(r) = AF(lp) * S(r)/B(r)$$

where S(r) be the size of the replica, B(r) is the bandwidth of the replica and calculated as:

$$B(r) = \sum_{i=1}^{n} B_i / n$$

where n is the number of replicas of file f.
Compare the cost of all the replicas.

Step-3: Replace the new replica with least cost.

Since replicas are stored in different grid sites and storage is limited there is a need of effective replica replacement algorithm. The proposed algorithm replaces the replica by calculating the weight. The weight is based on future access of the replica, bandwidth and size of the file.

4. EXPERIMENTAL RESULTS

4.1. Simulation tool

OptorSim is used to evaluate the performance of this algorithm. OptorSim [11] was developed by European Data Grid projects 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-1. Data replication strategies commonly assume that the data is read only in Data Grid Environments.

Table-1. General configuration of parameters.

Parameters	Values
Number of jobs	100
Number of job types	6
Number of file accessed per job	10
Size of single file	1GB
Total size of files	100 GB
Job delay	2500 mS

The architecture used here is the European Data Grid CMS test bed architecture [11, 12]. 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. There are 8 routers which is used to forward request to other sites.

4.2. Metrics used

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

Data grid produces huge volume of data. When all these data are replicated a huge amount of storage is needed. Storage is one of the main resources of data grid. Monitoring the use of storage resources in the grid sites is also a valuable source of information. Storage is limited and it should be used effectively. When a remote replica has been selected for replication to the site's Storage Element, the Storage Element might not have sufficient space capacity. In this case, one or more replicas must be deleted. Existing replica replacement algorithms are LFU and LRU. In LFU the Least Frequently accessed file is replaced. But is scientific experiments even the old files are important. In LRU the least recently used file is replaced, so even the hot spot replicas may be deleted. The proposed replica replacement algorithm considers both the future access frequency and also cost incurred for the replica.

To show the results of DRA it is compared with Modified BHR [13] that uses Least Frequently Used file for replacement and Modified BHR that uses the proposed algorithm for replacement. When compared to the LFU the proposed algorithm performs better in all mean job execution time, number of replications, storage used and in network usage. Table-2 shows the results for the Sequential Access Pattern Generator.

An ordinary grid user would require the fastest possible turnaround time for the job, and so mean job execution time is considered the most important of the evaluation metrics. The mean job execution time for the Sequential access pattern generator is shown is Figure-2. When compared to LFU replacement the Modified BHR Region Based Algorithm performs better with DRA and has the shortest mean job execution time. Modified BHR with DRA reduces the number of files to be replicated but only to certain extend and is depicted in Figure-3.

Storage used specifies the amount of space used by files in MB. Storage usage can be calculated for each site as a percentage of the capacity reserved by files out of the total capacity of the underlying storage. The storage used for the Sequential access pattern generator is depicted in Figure-4. When compared to Modified BHR Region Based Algorithm with LFU replacement, the proposed replacement algorithm performs better.

Replicating a file takes time and uses network bandwidth. When a required replica is deleted, it should be transferred from some other node. Hence there is a network usage. The effective network usage ranges from 0 to 1. This algorithm is optimized to minimize the bandwidth consumption and thus reduce the network traffic. When compared to Modified BHR Region Based Algorithm with LFU replacement, the proposed replacement algorithm performs better and is shown in Figure-5.

Table-2. Optorsim results for sequential access pattern generator.

Performance metrics	Modified BHR with LFU	Modified BHR with DRA
Mean job time of all jobs in msec	1145	820
Total number of replications	200	190
Percentage of storage filled/available	28.45	24.22
Probability of effective network usage	0.22	0.19

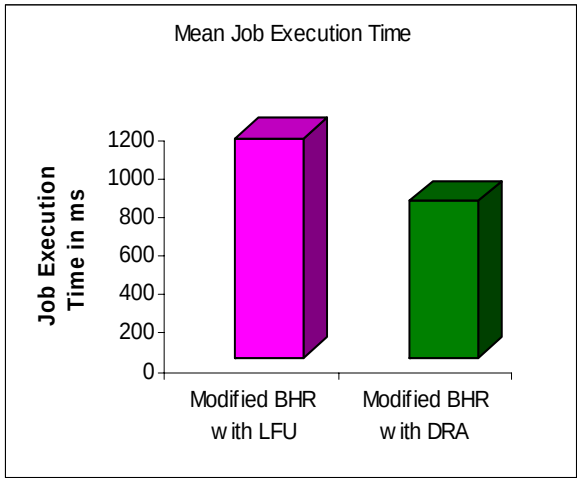

Figure-2. Mean job execution time.

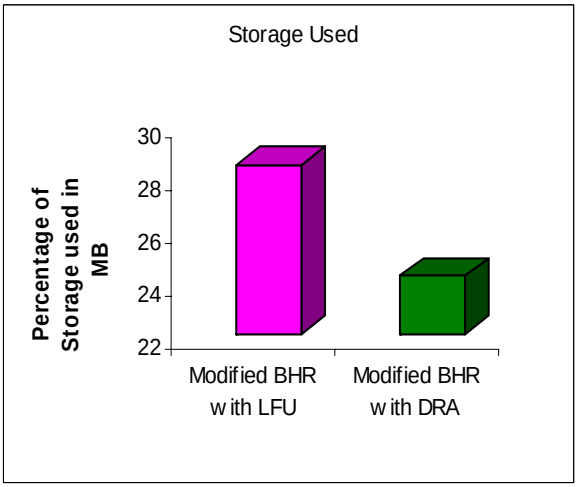

Figure-4. Average storage used.

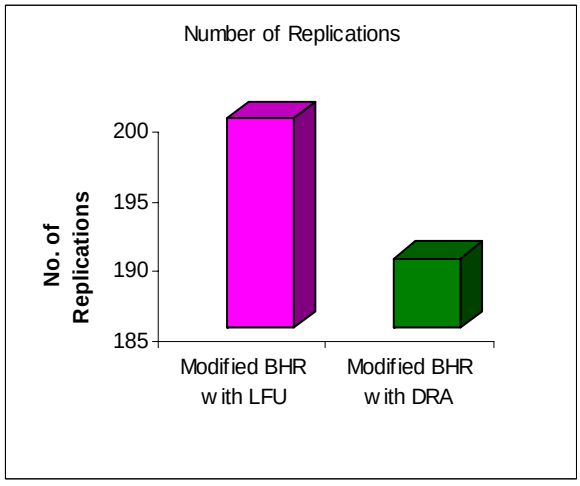

Figure-3. Number of replications.

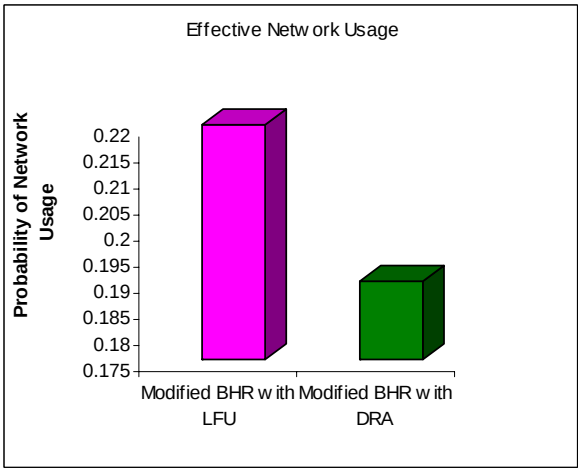

Figure-5. Effective network usage.

5. CONCLUSIONS AND FUTURE WORK

Replications are important mechanism in order to face the huge volume of data required by many grid users. Replica Replacement plays an important role in replica management strategy. Dynamic Replacement Algorithm predicts the popularity of the file in future and also considers the size of the replica with its bandwidth cost. In this work the performances are evaluated by mean job execution time, number of replications, storage used and effective network usage. As a future work, can be implemented in a real grid environment and other replica replacement strategies can be considered.

REFERENCES

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

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

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

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

[5] Ekow Otoo and Arie Shoshani. 2003. Accurate Modeling of cache replacement policies in a data grid.

[6] Jose Aguilar and Ernst L. Leiss. 2004. An adaptive Coherence-Replacement protocol for web proxy cache systems. Computation Systems. 8(1): 001-014.

[7] Ma Teng and Luo Junzhou. 2005. A prediction-based and cost based Replica Replacement Algorithm Research and Simulation. International Conference on Advanced Information Networking and Applications.

[8] Zhao Xu, Xiong and Wang. A weight based dynamic replacement strategy in Data Grids.

[9] Nikola Treka, Wil vander Aalst, Carmen Bratosin and Natalia Sidorava. Evaluating a Data removal strategy for grid environment using colored petrinets.

[10] Ruay-Shiung Chang, Hui-Ping Chang and Yun-Ting Wang. 2007. A dynamic weighted data replication strategy in data grids. IEEE Conference.

[11] D. G. Cameron, R. C. Schiaffino, J. Ferguson, P. Millar, C. Nicholson, K. Stockinger and F. Zini. 2004. OptorSim v2.0 Installation and User Guide, November. http://cern.ch/edg-wp2/optimization/optorsim.htm.

[12] K. Holtman. 2001. CMS data grid system overview and requirement. Tech report CERN, July.

[13] Sashi K. and Dr. Antony Selvadoss Thanamani. 2011. Dynamic replication in a data grid using a Modified BHR Region Based Algorithm. Future Generation Computer Systems, Elsevier. 27(2): 202-210.