6th International Conference on Computer and Knowledge Engineering (ICCKE 2016), October 20-21 2016, Ferdowsi University of Mashhad

PPRA: A new pre-fetching and prediction based replication algorithm in data grid

Mahsa Beigrezaei
Department of Computer and IT Engineering, Qazvin Branch, Islamic Azad University, Qazvin, Iran
m.beigrezai@gmail.com

Abolfazl Toroghi Haghighat
Department of Computer and IT Engineering, Qazvin Branch, Islamic Azad
Abolfazl.Toroghi Haghighat@gmail.com

Mohammad Reza meybodi
Department of Computer and IT Engineering Amirkabir University, Tehran, ran.
mmeybodi@auc.ac.ir

Maryam runiassy
Computer engineering Dep,Charles W.Davidson school of engineering, an state university, sanjose, 1California
Maryam.runiassy@sjsu.edu

Abstract

Today, scientific and business applications generate huge amounts of data. Users of data grid, who are distributed all over the grid geographically, need such data. So ensuring the access to this distributed data efficiently is one of the most important challenges in Data grid network. Data replication algorithms are known as the most common method used to overcome this problem. They distribute several copies of a file in the proper site to reduce access time, transfer cost, and bandwidth consumption. In this paper, we put forward a new dynamic replication algorithm called pre-fetching and prediction based replication algorithm (PPRA). PPRA replicates the popular file in suitable sites where next file accessing will happen with more likelihood and statistical methods are used to predict the number of future accesses to a file in each site. It also pre-fetches future needs to requester grid sites to increase file accessing locally. Therefore, it leads to lower file access time, means of response time and bandwidth consumption. OptorSim, as a common grid simulator, is used to evaluate the efficiency of this dynamic replication algorithm. The simulation results show that PPRA can give better average job execution time and bandwidth consumption as compared with NoRep, LRU, LFU, BHR and Modified BHR algorithms.

Keywords—prediction; data mining; data replication; data grid

I. INTRODUCTION

Today, most applications generate a huge amount of data and their users, who are distributed geographically around the world, need these data. Data grid is a distributed system that provides collaborative environments for applications and users. Huge amounts of data that are distributed among different grid sites are managed, controlled and shared in data grid environment [1][2][3][4]. Some data grid projects, including the Southern California Earthquake Center (SCEC) and Biomedical Informatics Research, employ data grids for their computational needs. Data grid faces such problems as high latency and low data availability. Replication is one of the most important techniques used to increase the performance of query latency, reliability and data availability. Indeed, replication methods try to decrease bandwidth consumption and file access delay in data grids [5][6][7][9][10].

In a famous classification, data replication algorithm is classified into two ones: static and dynamic. In the Static data replication algorithm, replication occurs based on prior knowledge and does not consider changed behavior users. Therefore, they cannot be adapted to the changed network, behavior of users and new access patterns. Data grid is dynamic environment in which changing is common. On the other hand, Dynamic replication algorithms create replica based on the condition of network and users access pattern, so they are more suitable and efficient for grid environments where the behavior of their user is changing. Dynamic data replication algorithm tries to answer three main questions: (i) what file should be replicated? (ii) When should data replication take place? (iii) What site is suitable for replication [?13][14].

The most effective parameter in the performance of a data grid is delay time. A low value of delay indicates the better performance of the system. The main reason leading to delay time in the grid system is remote file accessing, so if we predict the files that will be accessed in the near future and can be aware of their location of accessing, we can replicate them and decrease the delay of system. Indeed, by having information about future accesses, we can control replication operation more easily.

Although memory and storage size of computers are increasing every year, they are still not keeping up with the request of storing large number of data. The major challenge is a decision problem i.e. how many replicas should be created and where replicas should be stored. Placing the replicas in the appropriate site reduces the bandwidth consumption and reduces the job execution time. Also instead of storing files in many sites, they can be placed in an appropriate site so storage space is saved. The response time is an essential parameter that influences the replica selection and thus the job turnaround time. Each grid site has its own capabilities and characteristics; so, choosing appropriate site from many sites that have the required data is an important decision.

In this paper, we have proposed a dynamic data replication algorithm called pre-fetching and prediction based replication (PPRA). PPRA uses a statistical prediction method for predicting the future place and suitable place where the future file request is likely to occur. Our proposed method use a hierarchical search for reducing the number of searches and increasing the performance. PRRA also uses an association rule mining method called FP-Growth to extract the related file for perfecting and replicating them before they are needed in the requester sites. The rest of the paper is organized as follows. Section 2 gives a background and the related previous works on data replication used for grids. Section 3 describes the architecture of the data grid and the details of the PRRA algorithm. Section 4 shows simulation results. Finally, conclusion of this paper and suggestions for future works are presented in Section 5.

II. RELATED WORK

There are many research works that have addressed data replication in a Data Grid. Some of them are surveyed in this section.

An Exhaustive survey about data replication in data grid is presented by Tose et al. [15].This paper verifies the impact of data grid architecture on the efficiency of Data replication algorithm. Park et al.[16] (2004) have presented an algorithm named Bandwidth Hierarchy Replication (BHR). BHR takes advantage of network level locality. BHR replicates popular files in a site located in the requester region that has higher bandwidth connection to the job execution site. Sashi and Thanamani (2010) enhanced the BHR strategy and presented a new algorithm called Modified BHR Region based algorithm (MBHR) [17]. In MBHR, grid network is divided into several regions where the site is closely located. BHR stores file access history and the number of the file accessing in the region header and replicates the popular file in the site with maximum number of accesses to the file; and the file can be accessed locally for the longest time. So, the Mean Job Execution time and Network consumption are decreased, as compared to BHR.

Beigrezaei et al.[13] have proposed a new fuzzy replication algorithm (Fuzzy_rep) that could enhance the modified BHR algorithm. Fuzzy_rep uses a fuzzy inference system with tree input and one output in order to select the best location for replication. This inference system is based on the fuzzy rule that calculates a value for each site, indicating how much valuable it is. Replica placement in Fuzzy_rep perform more accurately than the modified BHR. Therefore, they lead to the enhanced performance of Data grid. Mansouri et al. [18] put forward a Dynamic Hierarchical Replication (DHR) algorithm. It selects the best replica for requester job based on the Hierarchical method. It considers the waiting time in storage and data transfer time for calculating replica access cost. Simulation results showed DHR algorithm was more effective in comparison with other algorithms when data grid sites had a small storage size. Another algorithm called MDHRA is presented by Mansouri et al.[19]. MDHRA uses effective parameters like the last time the replica are requested, the number of accesses, and the size of replica in the replica replacement phase. Replica selection is based on the transfer time. EDHR algorithm was also presented by these researchers [20]. This strategy uses an economic model based on the future value of a data file for file deletion when there is not enough space for the replica.

III. THE PROPOSED ALGORITHM

A. The Data Grid Architecture of the proposed algorithm

Data Grid architecture of the proposed algorithm divides the grid into regions. This architecture provides a hierarchical structure. In this structure, each region is composed of a set of sites or nodes. The sites within a region are located at a slight distance from each other. This architecture is like that of BHR method [16]. In this architecture, each region has a header node (server zone) that keeps information such as the number of accesses, last access time, and communication bandwidth between sites (nodes) within the region and other details of their activities. This information is sent from sites to the header server when the network traffic is minimum in the specified time inter*vals.*

B. The proposed replication algorithm based on prediction and prefetching (PPRA)

We have proposed a new algorithm called PPRA (pre-fetching and prediction based replication algorithm. The PPRA algorithm used data mining and predictive statistical methods to predict the best location of the future file access and the extract related files. The algorithm extracts the helpful information through data mining and uses them in its future decisions. This information includes the number of future accesses and the associated files with the requested file.

The Files that have relationship with each other are being accessed together with more probability. Our proposed algorithm, according to temporal locality accessing, tries to replicate the requested file and its associated files. Data mining method searches on the database of past access history in the region and extracts the adjacent files.

C. Collecting and storing the file access sequences

When the site requests a file, the information of the request is saved in its local data base. The local server of each region stores this information in its data base as a record. In the mentioned architecture, regions sites send new records to the data base of the region server at specified time intervals. This information with site name records holder is saved in the Data base of servers. After receiving the access history, region local server integrates them with its own data base information.

Thus, the file access sequences of sites are collected in one place and server could predicate future access files and extract the related files more accurately. This information includes access time, local or remote accessing and the name of the requester site.

D. Requesting a file and performing the replication

Jobs in the sites need the files to complete themselves. Running jobs request their needed files; if the site has the requested file; jobs can access them locally; otherwise, the request is sent for the region server. Replica manager in the region server make the selection unit aware of the request. Selection unit, based on replica selection algorithm choices, chooses the best replica for the requester site. The first version of the single unit duplicate (Replica Catalog) RC duplicates listings in the area for extracting the requested file.

E. Replica selection

At first, for each requested file, Replica Selection unit gets its replica list of requesting region from replica catalog (RC); then, a replica with the least transfer cost is selected from the list and a message is sent to the selective replica site in order to send the requested replica to the requesting site. First, it searches the best replica in current region replica list. If there exists any replica in the requesting region, replica in the other region is examined. The selection algorithm performs a hierarchical search for increasing speed and reducing the number of searches. Transfer cost is calculated according to the following formula.

$$\text{Transfer\_Cost}_{(\text{file } r, \ , \ \text{a\_source\_site}, \ \text{g\_destination})} = \frac{size_r}{bandwidth_{ag}} + Propagation\ delay\ time_{ag} + size/\ storageSpeed \quad (1)$$

Replication $\text{Cost}_{(r)}$: replication cost

- size_r: size of file ‘r’
- bandwidth_{ag}: Bandwidth between sites ‘a’ and sites ‘g’. Site ‘a’ contains a replica for file ’r’ and site g is destination site (or requesting site).
- Propagation delay time_{ag}: delay time for sending replica from source site(a) to destination site (g)
- *storageSpeed* : speed of storage

in the absence of the needed file in the requesting site, replica manager examines the number of access files in the site and its neighbors and makes a decision to replicate the file or not. It acts as follows:

1. If the number of access files in the site and its neighbors is greater than a threshold value, the replica manager replicates the file in a more appropriate site such that future file access possibility is more in it. This estimation is achieved via a statistical prediction method called Single Exponential Smoothing [21].
2. If the number of access files in the site and its neighbors is less than the threshold value, the file will not be replicated and the needed file is sent to the requesting site from a site that has the file and the least transfer cost to it.

A. Replica placement

The appropriate site for replication is the place that has the maximum future file access. File Access history may indicate the file access in the future. Because Users usually have a routine manner to access the files, statistical methods use the past history to predict the future. Simple Exponential Smoothing predicts future via weighting the past information based on Geometric progression. It means new information has more weight than the older information. In the following, the mathematical model of this method is described.

(2)

$$F_{t+1} = \alpha \cdot A_t + F_t - \alpha \cdot F_t \qquad (3)$$

$$F_{t+1} = F_t + \alpha \cdot (A_t - F_t) \qquad (4)$$

$$F_{t+1} = \alpha \cdot A_t + (1-\alpha) \cdot F_t \qquad (5)$$

In the mentioned formulas, A_i indicates the number of file accesses in the site j for the period t, and F is the predicted number of the file accesses for the period t + 1.Per each prediction, F_{i+1} is computed and stored in the statistical data base of the region server. Here an example showing the use of this method for prediction is mentioned

F. File pre-fetching

Users usually needs a set of specific files. So when a user requests a file, the probability of access to the related file in the future is more. According to the principle of spatial locality, the related files with the needed file will be requested as soon as possible. Therefore, if the related files is guessed and replicated before it is needed, the number of local accesses is increased and the cost and time of access to the files are reduced. In fact, in order to increase the efficiency of access time, file is perfected in suitable sites. PPRA algorithm has utilized Data mining methods to gain the related file.

Also, to determine the proper location for replicating the files in advance, PPRA uses a statistical based method called Simple Exponential Smoothing.

G. Extracting the related files for the pre-fetching

In this paper, PPRA used FP-Growth algorithm [14] as the association rule mining algorithm in Data mining science to extract the related files. For this purpose, the access history data base of each site and their neighbors was divided into equal time intervals base on the time during which they happened. Files accessed in each interval were considered as a transaction. The requested files in each time interval were considered as transaction items. FP-Growth algorithm searched these transactions to extract a set of association rules. Based on the association rule, we could determine accessing to what file could have an influence on accessing the other files. Then by extracting the name of related files, the name of requested file could be sent to a function which extracted the related files based on the extracted association rule. PPRA algorithm, if related file did not exist in the current region, could replicate them in the appropriate place.

H. Replica Replacement

As it is clear, the storage space of grid sites are bounded, implying that sometimes there is a lack of space for replication. When the act of replication is faced with the lack of storage, the replication algorithm should delete some replicas of site to obtain enough free space. Replica deleting should be performed carefully because replication is an expensive operation. In PPRA algorithm, if the selected site has sufficient storage space, replica is stored; otherwise, if replica did not exist for the file in the site, replication would not be performed and the algorithm is terminated. In the absence of the replica in the region, the algorithm sorts all replica of the site with LFU replacement algorithm and deletes files until obtaining enough space. In this deleting operation, first the files that have another replica in the region are deleted sooner than other files. When enough free space is created, replica can be stored in the site storage element.

IV. SIMULATION AND RESULTS EVALUATION

A. Simulation tools

To evaluate different optimization algorithms as part of *European* DataGrid (EDG) project, the grid simulator, OptorSim[22], was developed [23]. Other simulation tools such as MicroGrid[24], Briks[25], SimGrid[26]and GridSim for evaluating replica optimizer algorithms were also developed. Optorsim is a grid simulator that has been written by java for developing CMS project. The grid configuration that we have used in our simulation is the CMS Data Challenge 2002 test bed [24] (Fig. 1). We also used this simulator for simulation and performance evaluation of our proposed method.

The overall structure of optorsim is shown in Fig. 2. Some of the components have been listed as follows: i) the resource Broker that receives and schedules the job submitted from users; the main purpose of scheduling algorithms is to decrease the execution time of running a job; ii) the storage element (SE) is the place where the data are stored; having a SE is optional for a site; iii) the computing element (CE) represents computational resource; this element is also optional for a site; iv) the replica manager (RM) manages the replication and data transition between the sites. This element maintains a replica catalogue. v) The replica optimizer (RO) is a part of RM replica, and the RO controls file replication based on the replication policy.

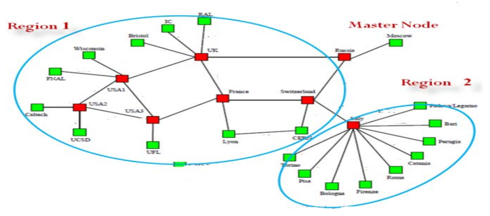

Fig.1. The network topology (CMS Data Challenge 2002 Grid topology)[24].

TABLE1. SIMULATION PARAMETERS.

parameters	value
number of job	10000
type of job number	6
job delay	2500
access pattern	sequential
prediction parameter	0,5
number of simulation	30

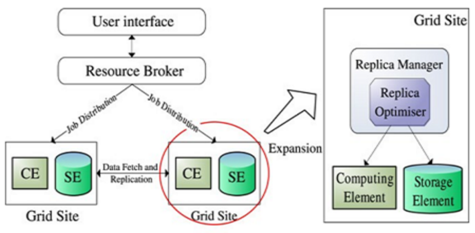

Fig.2. Structural of Optorsim [2].

B. *Configuration*

The simulation grid network topology used here is shown in Fig. (2). the data grid network is divided into two regions; the first and second regions contain 10 and 9 nodes respectively; they may have the computing or storage capacity. In this topology, a node containing all the original files is named the master; a capacity of 100 GB is considered for the master node. This architecture contains twenty sites, eighteen of which have the computing and storage capacity of 50 GB, and the other two nodes (site) have only the storage elements with 100 GB capacity, with one of them acting as master node. Eight routers delivering requests to other sites were used. The data files in our simulation were assumed to be read-only.

The minimum bandwidth between two sites was 45 and the maximum was 10000 MB. We considered the sequential access patterns by letting the files to be accessed in the order specified in the job configuration file. The main assumed parameters in this simulation have been shown in table 1. We compared our proposed algorithm with no-replication, LRU, LFU, and Modified BHR algorithms used in [16], [17], [26],[27],.

C. *Results*

In order to evaluate experimental results and the performance of the replication algorithms, various criteria have been used by researchers, such as means of execution time, the Effective Network Usage, the percentage of storage filled, number of replica, etc. in this paper, the performance of the proposed algorithm was evaluated by means of execution time, the Effective Network Usage, the percentage of storage filled and the number of replica. We compared our proposed algorithm with no-replication, LRU, LFU, and Modified BHR algorithms used in [16], [17], [26],[27].

- The mean job time of all jobs on grid

This criterion is one of the most important measures which is defined as the ratio of the total execution time of all the jobs (in terms of milliseconds) to the total number of the jobs; so the lower the execution time average, the better the algorithm, as the jobs are executed within a shorter period of time. In the first experiment, for 100 jobs under the same conditions, the lower value of this measure showed better performance and the algorithm was more suitable.

The simulation results shown in Fig. 3 indicated that the job execution time for the PPRA algorithm was lowest among no-replication, LRU, LFU, and Modified BHR algorithms. Because of the better replica placement in PPRA and the pre-fetching of the related file, PPRA showed the lower execution time. PPRA for predication had the best place for replication by using the effective principle for the spatial and temporal locality; indeed predicted the future needs of the sites on the basis of their past access history. So PPRA could replicate the future needed file before requesting in the appropriate site.

Indeed, PPRA selects right nodes with more effective parameters such as the number of past accesses to the file and the time of accessing, in comparison with other methods. Also, pre-fetching related files can lead to the files that are more likely to be locally available for the jobs.

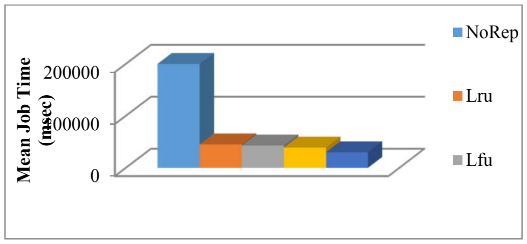

Fig. 3. The mean job time of all jobs on grid

- The Effective Network Usage

The Effective Network Usage Represents the effective use of bandwidth. It is defined as the ratio of files transferred to the files requested. The lower value of ENU shows better performance. In fact, due to placing replica in the appropriate location, the number of replica is decreased, leading to raising the performance of algorithm. ENU can be calculated by the following formula.

$$ENU = \frac{N_{remote\ file\ accesses} + N_{file\ replications}}{N_{remote\ file\ accesses} + N_{local\ file\ accesses}}$$

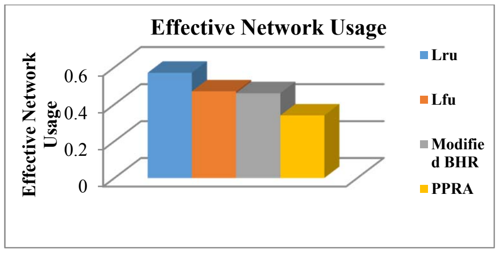

Fig. 4. The Effective network usage.

As shown fig 4, the ENU value for PPRA algorithm is lower than those of other algorithms. Due to pre-fetching the needed file before requesting and replicating files in the appropriate site, PPRA algorithm indicates the better value of ENU than other algorithms. So Bandwidth consumption is decreased in the Data Grid.

- The percentage of storage filled

Storage usage can be calculated as the average percentage usage of storage element for all grid sits by files. As shown in Figure 5, the Modified BHR and NoRep algorithms have a lower percentage of storage filled. The PPRA algorithm is the second best one. The reason for this increase is pre-fetching the related file before requesting the file in the sites. The lowest value belonged to NoRep algorithm because NoRep did not perform replication; thus, the storage capacity could not be consumed and all access files which is not exist in the requesting site were performed remotely.

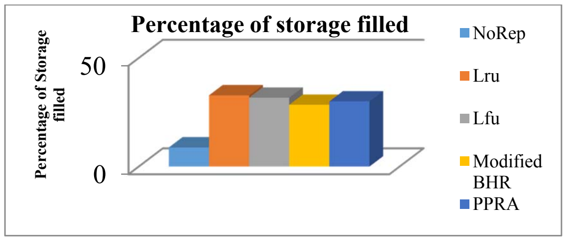

Fig. 5. The percentage of storage filled

- Total number of replications.

Replication is an expensive operation, as shown by the total number of replications. The grid system suffers from a high cost and the replication algorithm is not efficient. In this condition, the needed file is not accessible locally in the requester node. Consequently, replication will be needed. As shown in fig 6, the PRRA had lowest total number of replications, in comparison to other algorithms. Because PPRA predicts future needs and replicates them in future requester sites, most file accesses exist in the needed site and replications are decreased. It should be noted that NoRep algorithm does not replicate the needed file and all needed non-existing files are accessed remotely.

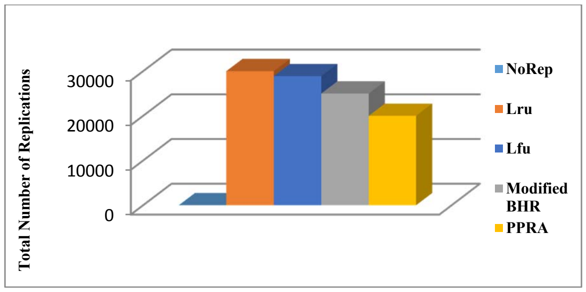

Fig.6. the total number of replications.

V. CONCLUSION AND FUTURE WORKS

In this paper, we presented a new dynamic algorithm called PRRA for Dynamic data replication in data grids. This algorithm used a statistical method for predication which was named Simple Exponential Smoothing method could be i=used for determining the suitable location of replication. PRRA, for preparing files before they are needed in the requesting sites, used the pre-fetching technique for replication. It predicted the future needs based on file access history; indeed, it extracted the related file and pre-fetched them to the requester grid. So the site could have access to the needed file locally. Here, for extracting the related file, PRRA used the FP-Growth method as one of the association rule mining methods in Data mining science. In order to evaluate the performance of our proposed algorithm, we used a data grid simulator named OptorSim. We compared PRRA with famous algorithms like No replication, LRU, BHR, Modified BHR. The experimental results showed that PRRA excelled in Mean Job Time and Effective Network Usage metrics.

In future work, we want to combine PPRA with a suitable scheduling to enhance performance. We aim to compare PPRA with more replication algorithms. We also plan to add more parameters like security to our proposed algorithm.

REFERENCES

[1] Ranganathan K, Foster I (2001) Identifying dynamic replication strategies for a high-performance data grid. In: Proceedings of the international grid computing workshop, vol 2242, Springer, pp 75–86

[2] Hoschek W, Jaen-martinez J, Samar A, Stockinger H, Stockinger K (2000) Data management in an international data grid project. In: IEEE, ACM international workshop on grid computing, pp 77–90

[3] International Organization of Standardization (2011) ISO/IEC/IEEE 42010:2011—systems and software engineering–architecture description. Technical report

[4] Moore R, Baru C, Marciano R, Rajasekar A, Wan M. Data-intensive computing in "the grid: blueprint for a new computing infrastructure". J Netw Comput Appl" Morgan Kaufmann: San Francisco 1999:105–29.

[5] O. Wolfson, S. Jajodia, Y. Huang, An adaptive data replication algorithm, ACMTransactions on Database Systems 22 (2) (1997) 255–314.

[6] M. Rabinovich, I. Rabinovich, R. Rajaraman, Dynamic replication on the internet, Technical Report, HA6177000–980305-01-TM, AT&T Labs, March .1998

[7] J.M. Perez, F. Garcia-Carballeira, J. Carretero, A. Calderon, J. Fernandez, Branch replication scheme: a new model for data replication in large scale data gridsFuture Generation Computer Systems 26 (1) (2010) 12–20.

[8] M. Vrable, S. Savage, G.M. Voelker, Cumulus: filesystem backup to the cloud, ACM Transactions on Storage 5 (4) (2009(

[9] H. Shen, An efficient and adaptive decentralized file replication algorithm in P2P file sharing systems, IEEE Transactions on Parallel and Distributed Systems 21 (6) (2010) 827–840.

[10] Gray J, Helland P, O'Neil P & Shasha D. The dangers of replication and a solution. In: Proceedings of the ACM SIGMOD international conference on management of data. ACM Press; 1996. p. 173–82.

[11] Khanli L.M., Isazadeh A., Shishavanc T.N., "PHFS: A dynamic Replication method, to decrease access latency in multitier Data Grid", Future GenerationComputer Systems 27, pp. 233-244, 2011.

[12] Ranganathan, K., Foster, I. (2001), Design and evaluation of dynamic replication strategies for a high performance Data Grid. In: International Conference on Computing in High Energy and Nuclear Physics .

[13] Beigrezaei m., torohji haghighat A.,rashidi kanan M.,(2013), "a new fuzzy based algorithm in data grid", 13th Iranian Conference on Fuzzy Systems (IFSC),IEEE.

[14] U. Cibej, B. Slivnik, and B. Robic," The complexity of static data replication in data grids", Parallel Computing 31 (8) (2005) 900–912.

[15] Amjad, T.; Sher, M.; Daud, A.: A Critical Survey of Data Grid Replication Strategies Based on Data Mining Techniques. J. Sup. Comput. 71, 4116-4140 (2015)

[16] S.M. Park, J.H. Kim, Y.B. Ko, and W.S. Yoon, "Dynamic Data Replication Strategy Based on Internet Hierarchy BHR", in: Lecture notes in Computer Science Publisher, vol. 3033, Springer-Verlag, pp. 838-846, 2004.

[17] Sashi K., Thanamani A.S., (2011),"Dynamic replication in a Data Grid using a Modified BHR Region Based Algorithm", Future Generation Computer Systems 27, pp. 202-210.

[18] Mansouri N, Dastghaibyfard GH (2012) A dynamic replica management strategy in data grid. J Netw Comput Appl 35(4):1297–1303

[19] Mansouri, N.; Dastghaibyfard, G.; Mansouri, E.: Combination of data replication and scheduling algorithm for improving data availability in Data Grids. J. Netw. Comput. Appl. 36(2), 711–22 (2013)

[20] Mansouri, N.; Dastghaibyfard, G.: Enhanced dynamic hierarchical replication and weighted scheduling strategy in Data Grid. J. Parallel. Distrib. Comput. 73(4), 534–43(2013)

[21] Gardner, E.S., Jr. (1985). "Exponential smoothing: the state of the art." Journal of Forecasting, 4, 1-38. J. Han, H. Pei, and Y. Yin. Mining Frequent Patterns without Candidate Generation. In: Proc. Conf. on the Management of Data (SIGMOD'00, Dallas, TX) .ACM Press, New York, NY, USA 2000

[22] OptorSim – A Replica Optimiser Simulation, http://grid-data-management.web.cern.ch/grid-data-management/optimization/optor.

[23] CMS Data Challenge, http://www.uscms.org/s&c/dc04

[24] MicroGrid – A Grid Simulator, http://www.microgrids.eu/default.php/

[25] Bricks – A Performance Evaluation System for Grid Computing Scheduling Algorithms, http://ninf.apgrid.org/bricks/

[26] SimGrid – A Grid Simulator, http://simgrid.gforge.inria.fr/