Journal of Advances in Computer Research
Quarterly pISSN: 2345-606x eISSN: 2345-6078
Sari Branch, Islamic Azad University, Sari, I.R.Iran
(Vol. 11, No. 2, May 2020), Pages: 45-58
www.jacr.iausari.ac.ir

Increasing Performance in Data Grid by a New Replica Replacement Algorithm

Mahsa Beigrezaei [1], Abolfazl Torghi Haghighat ✉[1], Seyedeh Leili Mirtaheri [2], Narges Hajizadeh Bastani [1]
1) Department of Computer and Engineering, Yadegar -e- Imam Khomeini (RAH) Shahr-e-Rey Branch, Islamic Azad University, Shahr-e-Rey, Iran
2) Computer Engineering Department, Kharazmi University, Tehran, Iran
m.beigrezai@qiau.ac.ir; haghighat@qiau.ac.ir; mirtaheri@khu.ac.ir; narges_hajizade@yahoo.com

Received: 2020/08/12; Accepted: 2020/12/04

Abstract

Data Grid provides sharing services for very large data around the world. Data replication is one of the most effective approaches to reduce access latency and response time. In addition to the benefits, replication has costs such as storage and bandwidth consumption, especially when storage space is low and limited. Therefore, the data replacement should be done wisely. In this paper, we proposed a replacement method called FRA. The algorithm defines a weight for each replica that represents its value. This algorithm uses this weight to prevent the removal of valuable replicas. The results demonstrated that FRA algorithm has better performance than other replication methods in terms of the number of replications, the percentage of storage used, and the job execution time.

Keywords: *Data Grid, Data Replication, Replica Replacement*

1. Introduction

Today, scientific, engineering and commercial applications are producing very large amounts of data in terabytes and petabytes. This data is generated and shared worldwide. Users of these applications need fast data access. Therefore, managing this huge amount of data with the aim of increasing speed and accessibility is one of the most important issues for researchers. Distribution systems such as cloud and Grid are solutions for these problems [1-2]. They provide the infrastructure for the application to manage this huge amount of distributed data.

Grids and other distribution systems connect homogeneous and heterogeneous resources at different locations. They provide a scalable infrastructure that can provide remote sharing and access to resources [3]. In a general classification, the Grid is divided into two types of Data Grid and computational Grid. Computational Grid Provides infrastructure for performing computationally intensive tasks with appropriate performance. The Data Grid deals with data sharing, collaboration, and performing data-intensive tasks [4]. The Data Grid has a problem with data access efficiency due to the volume of data required by the data-intensive tasks and the high latency in this environment.[5] Data replication is one of the most important ways to increase performance and data availability in various distributed systems such as Data Grid and the Internet [6-7]. Data replication has many advantages. It can decrease data access time and bandwidth consumption.it also can enhance availability, scalability, and load balancing. Replication methods create multiple copies of files and place them near the data requester to decrease data access time [8]. The replication strategy has to answer some questions that include: What file and when should the file be replicated? Where should replication be stored? What files should be deleted when storage space is not enough? Depending on the answers, many various methods has been presented by researchers. Replication methods are divided into dynamic and static [1], [9]. Static approaches [1, 9] determine the number of replicas and their location at the design phase. After storing the replica, it will have remained until it is deleted by the user manually; these locations are unchangeable. Thus, static replication is not suitable when the user changes his/her behavior. Besides, dynamic replication [7], [18-25] creates and deletes copies of files based on users' behaviors. Indeed, it makes decisions based on access patterns during the time. Distributed systems are a dynamic environment, and their user's requirements access pattern change during the time; hence, dynamic replication is more tailored for distributed systems [7], [17].

Our proposed method is provided for a grid with a hierarchical structure. In this structure grid divide into some regions. The sites in the region are connected to each other with high bandwidth. Our proposed method uses a fuzzy-based replacement algorithm for finding a tailored replica for deleting. The algorithm selects the replicas with the least likely to receive file requests in the future. Indeed it finds low valuable replica base on file access history. The simulation results show our replacement algorithm can improve data access performance. The other sections of the paper are organized as follows. Section 3 presents the related works; in section 3, we describe our replacement method, then in section 4, we evaluate the method and show the results of our simulation. Section 5 concludes the paper and explain future works.

2. Related Works

Several recent studies have taken into account the data replication algorithm in the Data Grid. We mention some of them below.

In Ref. [14], the least frequently used (LFU), and two economic strategies were proposed that take place the replication when the file is accessed remotely. Simulation results showed, while replication is done, if free space in the replication site is not enough, LFU works better than economic strategies and LFU. Another similar replication algorithm called LRU was proposed In Ref. [15]. LFU and LRU delete the least frequently accessed file and least recently used file respectively.

In [18], we presented a fuzzy-based replication method named (Fuzzy_Rep). It utilizes a fuzzy inference system to select a tailored place for replication. The fuzzy inference system uses three input parameters, including bandwidth, the last access time, and the numbers of file access.

In Ref. [20], an effective replication method called bandwidth hierarchy replication (BHR) algorithm, was presented. BHR works based on the network level. A hierarchical structure consisting of several regions for the Grid is assumed in this method. The sites in the same network region are connected with high bandwidth. BHR significantly reduces the cost of data access by replication of the data required for a job near the place that the job is executed. When there is no space for replication on the selected site, it deletes files that have at least another copy in the region. It reduces inter-region transmission and enhances performance.

In Ref. [20], Sashi and Thanamani presented a modified version of the BHR algorithm called the modified BHR(MBHR). Simulation results show MBHR method improves the performance in comparison with BHR method by selecting a site with the maximum number of accesses in the requesting region. Also, it can prevent unnecessary replications.

Tehmina Amjad and et al. presented a comprehensive survey paper about data replication in the Grid [21]. They studied different replication techniques. In this paper, for comparing replication techniques, all those parameters are presented in a table.

In Ref. [22] a new replication approach was presented, which classifies the data. Based on this classification, the algorithm selects the appropriate location for the duplicate file and efficient job scheduling. It sends jobs to the sites that have files in the needed category to decrease file transfer costs.

Ranganathan and Foster [23] introduced six replication methods, including i) no-replication or caching ii) cascading, iii) best client, iv) caching plus cascading, v) plain caching replication, and vi) fast spread. These strategies were evaluated in random access, small temporal locality, and small geographical access patterns. The simulation results show that for increasing the performance in each access pattern, a suitable strategy should be used.

In Ref. [24] proposes a replication algorithm that replicates files in a site with the highest degree and frequency of access and lowest workload. It was named the dynamic replica placement algorithm with load balancing (RPLB). It used the optimism simulator for evaluation. The results showed RPLB algorithm could enhance performance parameters such as the Effective Network Usage (ENU), and job execution time.

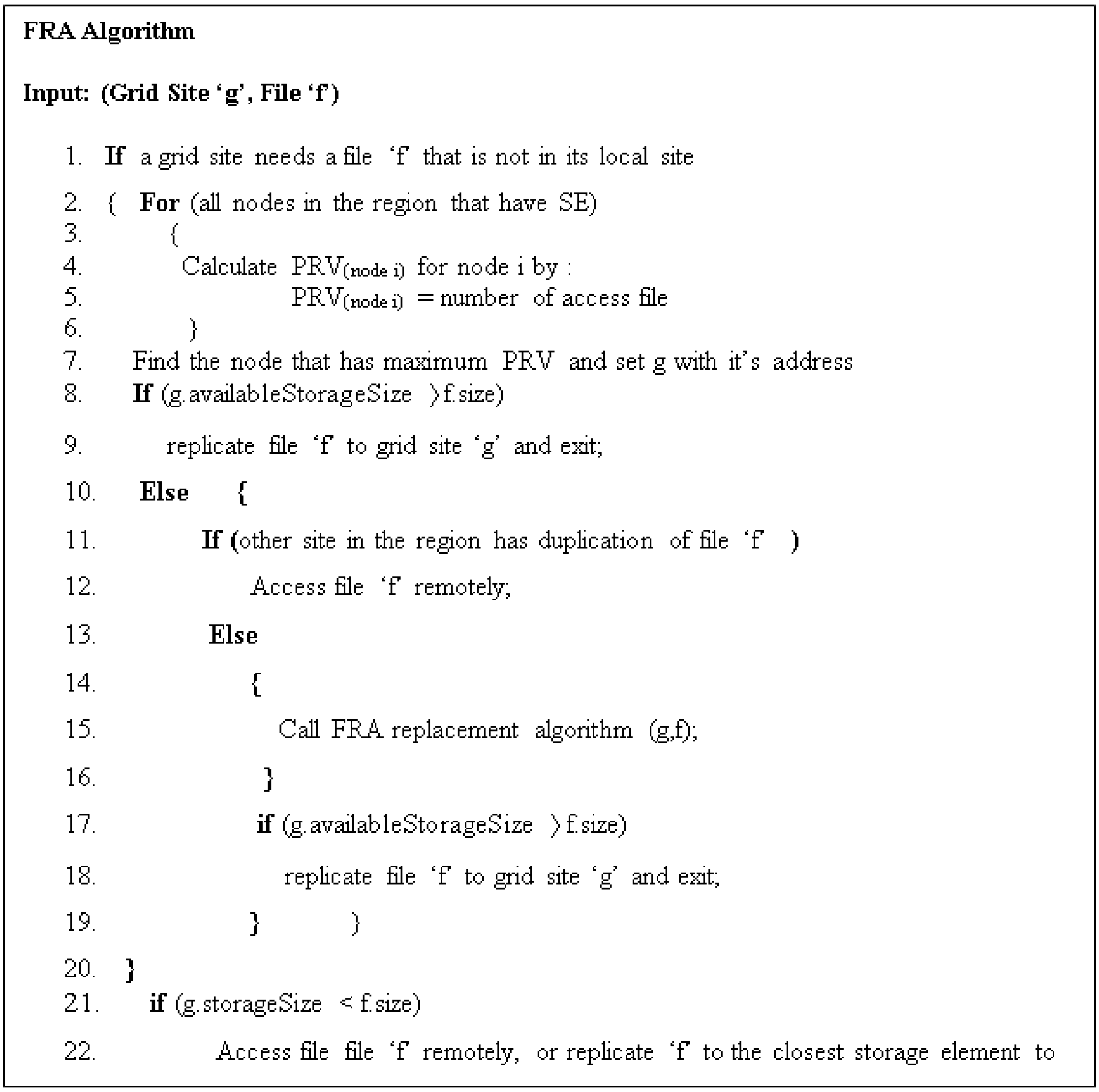
In Ref. [25], the authors presented three algorithms include a replication algorithm, a job scheduling algorithm, and a job migration method. This algorithm first selects the appropriate cluster for scheduling a job and then the best site for job execution. Decision making in this algorithm is based on the workload parameter. The migration algorithm also migrates jobs to balance the workload.

In Ref [26], replication algorithms were introduced that utilizes a hierarchical three-level architecture. It replicates popular files at scheduling time. It also uses a migration method to improve workload.

3. The Proposed Replication Algorithm

The resource broker gets jobs from users, and schedules them based on its used job scheduling algorithm, and finds suitable computing elements for job execution; afterward, the jobs are assigned to computing elements. When the resource broker schedules users' jobs and sends them to a suitable site for execution, the required data are produced in the master site; after that, the data is dispatched to each region header. Remote access should happen if the required files are not available locally. Data are huge in the data grid, so remote data access increases job execution time and reduces performance. Therefore, it is necessary to increase data locality in the data Grid. An effective replication algorithm can provide this increase. All parts of data replication, especially the replica replacement in the absence of storage space, have an impact on increasing data locality and performance. This effect is greater when storage space is less than the size of needed replicated files.

In this paper, we proposed a replication algorithm like MBHR method. The MBHR method is based on the network level locality. In the MBHR, the Grid was clustered into regions. In a region, sites connect to each other with high bandwidths. The MBHR algorithm tries to increase network locality at the regional level. It places new replicas in the requester region in a site with maximum access frequency. In replicating time, when there is not enough free space in the site, it uses LRU replacement algorithm. Our proposed replication algorithm acts like MBHR, but when free storage space is not enough in the replication site, it deletes replica by a new Fuzzy replacement algorithm.



```
FRA Algorithm

Input: (Grid Site 'g', File 'f')

1.  If a grid site needs a file 'f' that is not in its local site
2.  { For (all nodes in the region that have SE)
3.      {
4.       Calculate PRV(node i) for node i by :
5.               PRV(node i) = number of access file
6.       }
7.    Find the node that has maximum PRV and set g with it's address
8.    If (g.availableStorageSize ) f.size)
9.       replicate file 'f' to grid site 'g' and exit;
10.   Else {
11.        If (other site in the region has duplication of file 'f' )
12.            Access file 'f' remotely;
13.        Else
14.            {
15.              Call FRA replacement algorithm (g,f);
16.            }
17.            if (g.availableStorageSize ) f.size)
18.              replicate file 'f' to grid site 'g' and exit;
19.            }        }
20. }
21.   if (g.storageSize < f.size)
22.          Access file file 'f' remotely, or replicate 'f' to the closest storage element to
```

Figurer 1. FRA Replication algorithm

The steps proposed replication algorithm (called FRA) is illustrated in Figure 1. As shown in Figure 4, in step 4, our algorithm chooses the right site for replication like MBHR algorithm. In step 10, when there is not enough storage, call fuzzy replacement algorithm, the less value replicas is candidates for replacement is selected base on replicas weight. This weight computes with a fuzzy inference system. The fuzzy algorithm considers two factors for computing the weight of Replicas. These factors are Time interval between current time and the last replica access time (TI) and the number of file access(NA). These factors are described as follows:

- The number of file access (NA): Based on the principle of spatial and temporal locality, the higher the number of accesses to a file, the more likely the file will be requested soon. Therefore with the increase of this factor, the weight and future value of replica also increase.

- The time interval between the now and the last access time (TI): According to the principle of temporal locality, the closer the time interval between the last access to the file to the current time, the more likely it is to access the file in the near future. The replica weight is also higher.

Figure 2 shows an abstraction of the used fuzzy inference system. It shows a fuzzy system that has two inputs and one output, which determines the weight of the replica. The fuzzy inference system uses nine rules. We show some of them in Table 1. For example, rule 1 indicates that if "NA is high and TI is low, then the weight of the replica is very high.

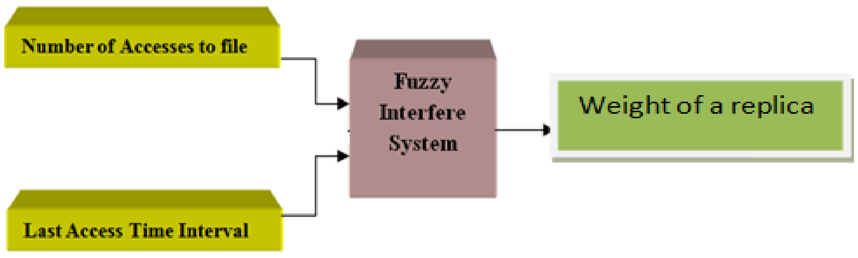


Figure 2. An abstraction of used fuzzy inference system.

Table 1. the Rules used in the fuzzy inference system

number	**Description**
1	If (NA is **high**) and ((TI is **low**) then (weight is very **high**)
2	If (NAis **average** and (TI is **low**) then (weight is **high**)
3	If (NAis **high**) and (TI is **high**) then (weight is **average**)
4	If (NAis **low**) and (TI is **low**) then (weight is **average**)
5	If (NAis **low**) and (and (TI is **low**) then (weight is **low**)

Figure 3 shows the FRA replacement algorithm. As shown in Fig 3, first, It checks the available storage space on the site 'g' selected for replication. If there is enough free space to replicate the file 'f,' then file 'f is replicated. Otherwise, for all files on the site that have another version in the current region, the weight(replica value) is calculated using the fuzzy inference system. After computing the weight of the candidate's files (stored in the "selected replicas" list), the algorithm sorts them according to their weight in ascending order. Then replicas with the lowest weight are selected and are removed until empty space is provided.

```
FRA Replacement Algorithm (Grid Site g, File f)

1. {
2. if (g.availableStorageSize(f.size)
        i. replicate file 'f' to grid site 'g' and exit;
   b. else
        i. {
        ii. "selected replicas"=among all replicas stored at 'g', select those that
            are also available in other grid sites of current region;
        iii. for all(all replicas stored at 'g' that available in current region)
        iv. {
                Calculate PRV for replica r :RPV (replica r) = call Fuzzy Function
            (Number of Accesses(r, Last Access Time Interval(r)) //fuzzy function
            with
        v. }
        vi. "selected replicas"=sort the replicas of "selected replicas" in ascending
            order, according to their RPV;
        vii. while ("selected replicas" != empty)
        viii. {
              select a replica from top of the "selected replicas" and delete it
              from grid site 'g';
              if(g.availableStorageSize(f.size)
              replicate file 'f' to grid site 'g' and exit;
        ix. }
        x. "selected replicas"=select remaining replicas of grid site 'g'; //they are
            not available in other grid sites of current region
        xi. go to line iii;
        xii. }
```

Figure 3. FRA replacement algorithm

4. Experimental Evaluation

4-1. Simulation Tools

Grid architecture in our proposed algorithm, a hierarchical architecture is assumed that consists of a set of regions. Figure 5 shows the grid topology in our simulation, which consists of two regions. Sites in the regions have computational or storage elements. We used the CMS [28-30] testbed architecture with a little change(we changed bandwidth and position of the master site). The original files are stored in one of the sites with a capacity of 100 GB called the master site. Most of the existing replication strategies assume data is read-only [49]. We also assumed data is read-only in our simulation. The used simulation parameters are shown in Table 1. We evaluated our method for the sequential, Gaussian Random, and random access patterns. Access patterns determine the order in which files are accessed in the jobs. The queue access cost is the Job scheduling algorithm that we used. We compared our proposed algorithm with the no-replication, LRU, LFU, and MBHR algorithms [14], [17], [7].

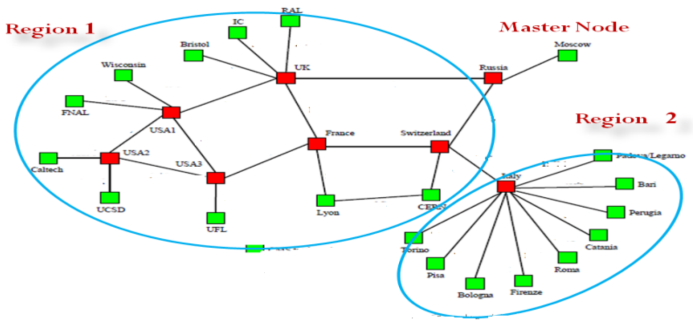


Figure 4. The network topology [18].

Table 1. Simulation configuration parameters.

Parameter	*Value*
Jobs no.	100
Job types no.	10
access pattern generators	Sequential
File size (GB)	1
Total size of files (GB)	97
Access history length(s)	1000
Experiments no.	10

4-2. Fuzzy inference system implementation

As mentioned before, our method chooses the right replica to remove when there is not enough space in our fuzzy replacement algorithm based on the amount of weight (value) allocated to each replica. For this purpose, a fuzzy function with two parameters is called. In order to implement the fuzzy inference system, we used the fuzzy toolbox of MATLAB software. The properties of the fuzzy inference system are specified in Table 3.

Table 3. The properties of the used fuzzy inference system

parameters	value
inference engine	Mamdani
And Method	min
Or Method	max
Defuzz Method	centroid
Implication Method	min
Aggregation Method	max

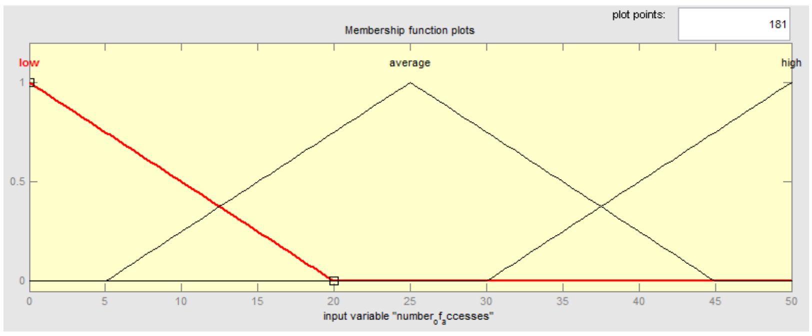


Figure 5. membership function of the first input (The number of accesses)

The first fuzzy input of the replica weight(value) in the inference system is NA(the number of replica accesses)per site. In our simulation, the membership function of this input is shown in Figure 5. The second input of this fuzzy system is TI; in our simulation, it is set to 10^6 milliseconds; its membership function is shown in Figure 6. The output of our fuzzy is the weight of the replica, which is a number between 0 and 1. Figure 7 shows the membership function of the weight of the replica(Replica value).

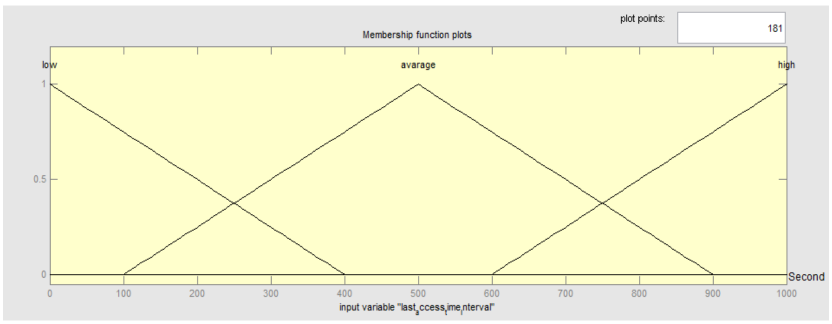


Figure 6. membership function of the second input (Last access time interval).

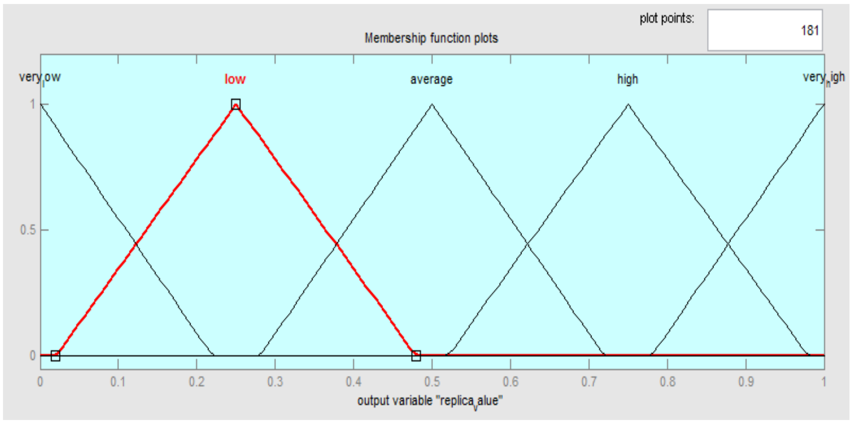


Figure 7. membership function of output (Replica weight/value)

4-3. Simulation results and discussion

In this section, we explain the simulation results. We evaluate our method in comparison with no-replication, LRU, LFU, and MBHR methods in term of the average job execution time, and the percentage of storage used, the total number of replication, and effective network usage

4-3-1.The mean job time

The mean job time is one of the most important metrics for evaluating replication and job scheduling algorithms. This metric is defined as the average execution time of all jobs in the Grid. The lower the value, the better the algorithm in the Grid.

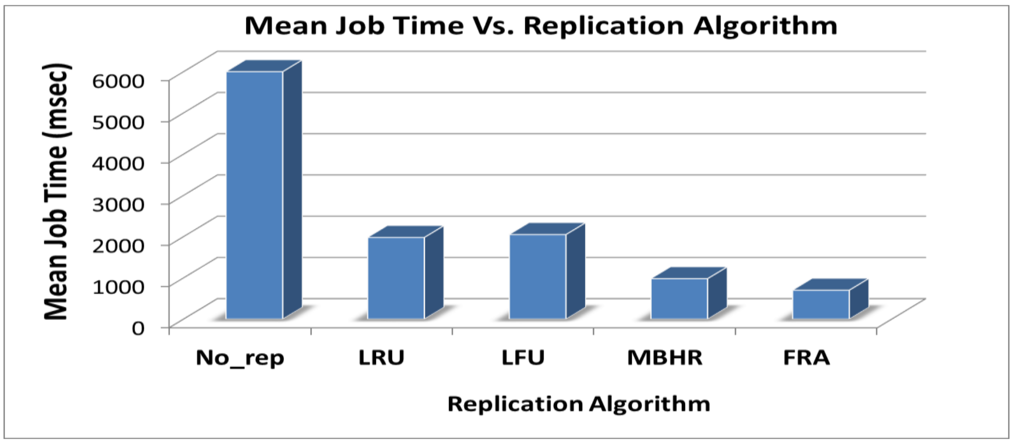


Figure 8. The mean job execution time for different algorithms.

In Figure 8, the simulation results indicate that the job execution time for our algorithm is lowest among the others. Because the use of the fuzzy replacement method help selects the lowest value replica for deleting, indeed, the files are more probably locally available to the jobs at the time of job execution. It can also put this way that replication in our method (FRA) is done more reasonably and correctly than the other methods. The right site is selected on the basis number of accesses file in comparison with the LRU, LFU and no_rep methods and use of fuzzy replacement algorithm in our algorithm lead better result in comparison with MBHR method.

4-3-2. The Effective Network Usage

The effective network utilization (ENU) parameter indicates the efficiency of the replication algorithm in terms of evaluation network metrics such as bandwidth consumption. It can be calculated measured using Eq. (2). In Figure 9, the ENU has been improved in the FRA method as compared to the MBHR method. Because the FRA replacement algorithm increases the number of local accesses to the files, and thus, the need for replication decreases. Therefore, ENU reduces.

$$\text{ENU} = \frac{N_{remote\ file\ accesses} + N_{file\ replications}}{N_{remote\ file\ accesses} + N_{local\ file\ accesses}} \qquad (2)$$

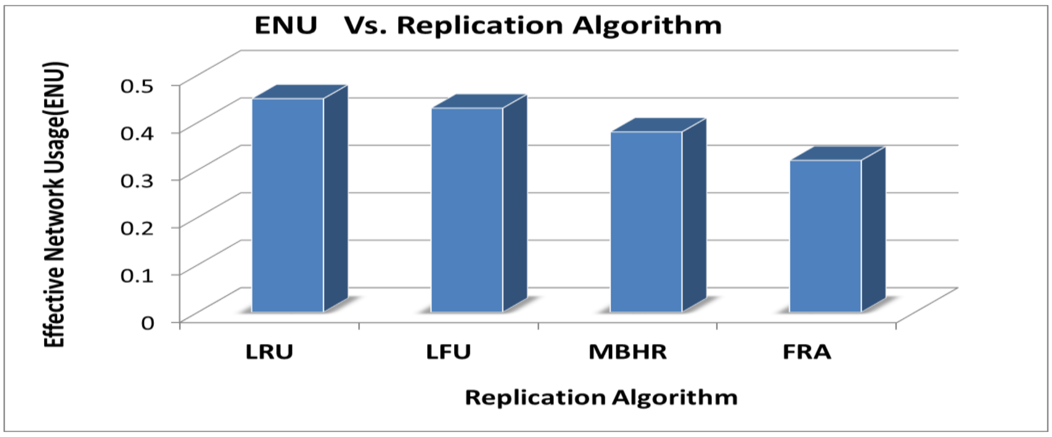


Figure 9. Effective network usage (ENU)

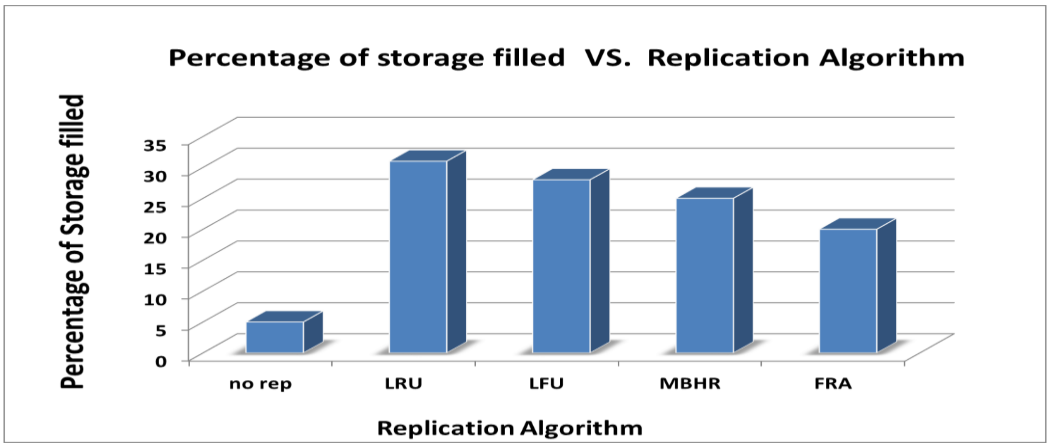


Figure 10. The percentage of the used storage space.

4-3-3. The percentage of storage filled

The percentage of storage filled is defined as the average percentage storage space that is used for saving files and replicas in all Grid sites. As shown in figure 10, the FRA algorithm consumes less storage compare with LFU, LRU, and MBHR algorithms

4.3. 3. The total number of replications.

The number of replications indicates the efficiency of the replication algorithm in choosing the right replication site for replication and low-value replicas for deleting. The lower the number of copies, the more locally accessed they are. Figure 11 indicates that FRA method has a lower total number of replications than other algorithms. Because of this reason that FRA prevents the removal of replicas that will soon be requested on the site. Therefore the need for replication reduce. We also note that no-Rep (no replication) method doesn't perform any replication, so its total number of replications is always zero.

5. Conclusion

Data Grid provides the infrastructure for sharing and managing huge amounts of data. Data access latency is one of the main challenging issues in Data Grid environments. Data replication is used as a key method to solve this problem. In this paper, we present a replication algorithm that improves the MBHR algorithm. Our proposed algorithm uses a fuzzy replacement algorithm that utilizes the fuzzy inference system with two parameters. Using this system it selects the low-value replicas for deleting. Indeed, it prevents the removal of valuable data and has better results than LRU, LFU, and MBHR algorithms. In future work, we intend to consider more effective metrics in the computational replica's value using the fuzzy inference system. Also, the use of parallel transfer in our algorithm is our future research.

References

[1] Khanli L.M., Isazadeh A., Shishavanc T.N., "PHFS: A dynamic Replication method, to decrease access latency in multi-tier data grid", Future GenerationComputer Systems 27, pp. 233-244, 2011.

[2] Bakhshad S, NOOR R, Akhunzada A, et al. A Dynamic Replication Aware Load Balanced Scheduling for Data Grids in Distributed Environments of Internet of Things.. Adhoc & Sensor Wireless Networks 2018; 40..

[3] jose M. Perez, Felix Garcia-Carballeira, Jesus Carretero, Alejandro Calderon,Javier Fernandez, "Branch replication scheme: a newmodel for data replication in large scale data grids", Future Generation Computer Systems 26 (1) (2010)12–20.

[4] Beigrezaei, M., Toroghi Haghighat, A. and Leili Mirtaheri, S., Minimizing data access latency in data grids by neighborhood-based data replication and job scheduling. International Journal of Communication Systems, 2020, p.e4552.

[5] Chang R-S., Chang J-S., Lin S-Y., "Job scheduling and data replication on data grids", Future Generation Computer Systems 23, pp. 846-860, 2007.

[6] Xiong L, Yang L, Tao Y, Xu J, Zhao L. Replication strategy for spatiotemporal data based on distributed caching system. Sensors 2018; 18(1): 222.

[7] S.M. Park, J.H. Kim, Y.B. Ko, 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.

[8] M. Tang, B.S. Lee, X. Tang, C.K. Yeo," The impact of data replication on job scheduling performance in the data grid", FutureGeneration Computer Systems" 22 (3) (2006) 254–268.

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

[10] M. Tang, B.S. Lee, C.K. Yao, X.Y. Tang, "Dynamic replication algorithm for the multi-tier data grid", Future Generation Computer Systems 21 (5) (2005(.۷۹۰–۷۷۵

[11] Atakan Dogan, "A study on performance of dynamic file replication algorithms for real-time file access in data grids", Future Generation Computer Systems 25 (8) (2009) 829–839.

[12] K. Ranganathan, I. Foster, Simulation studies of computation and data scheduling algorithms for data grids, Journal of Grid Computing 1 (1) (2003(74-63

[13] K. Ranganathan, I. Foster, "Identifying dynamic replication strategies for high performance data grids", in: roceedings of the Second InternationalWorkshop on Grid Computing, Denver, CO, ovember 2001, pp. 75–86.

[14] G. Camaron, A.P. Millar, C. Nicholson, R.C. Schiaffino, F. Zini, K. Stockinger, "Analysis of scheduling and replica optimisation strategies for data grids using optorsim", Journal of Grid Computing 2 (1) (2004) 57–69.

[15] Beigrezaei M, Haghighat AT, Meybodi MR, Runiassy M. PPRA: A new pre-fetching and prediction based replication algorithm in data grid. In: IEEE. ; 2016: 257–262..

[16] H. Lamehamedi, B.K. Szymanski, "Decentralized data management framework for data grids", Future Generation Computer Systems 23 (1) (2007) 109–115.

[17] X. Dong, J. Li, Z. Wu, D. Zhang, J. Xu, "On dynamic replication strategies in data service grids", in: Proceeding of 11th IEEE Symposiumon Object Oriented Real-Time Distributed Computing, ISORC, 2008.

[18] Beigrezaei M, Haghighat AT, Kanan HR. A new fuzzy based dynamic data replication algorithm in data grids. In: IEEE. ; 2013: 1–5..

[19] C. Nicholson, D.G. Camaron, A.T. Doyle, A.P. Millar, K. Stockinger, "Dynamic data replication in LCG 2008", in: UK e-Science All Hands Conference, 2006.

[20] Sashi K., Thanamani A.S., "Dynamic replication in a data grid using a Modified BHR Region Based Algorithm", Future Generation Computer Systems 27, pp. 202-210, 2081.

[21] Tehmina Amjad, Muhammad Sher, Ali Daud. "Asurvey of dynamic replication strategies formproving data availability in data grids", Future Generation Computer Systems, 2012.

[22] N. Nguyen Dang, S. Boem Lim2: "Combination of Replication and Scheduling in Data Grids", IJCSNS International Journal of Computer Science and Network Security, VOL.7 No.3 ,March 2007.

[23] Foster, K. Ranganathan, Decoupling computation and data scheduling in distributed data-intensive applications, in: Proceedings of the 11th IEEE International Symposium on High Performance Distributed Computing, HPDC-11, IEEE, CS Press, Edinburgh, UK, 2002, pp. 352–358.

[24] Rajaretnam K, Rajkumar M, Venkatesan R. Rplb: A replica placement algorithm in data grid with load balancing. International Arab Journal of Information Technology (IAJIT) 2016; 13(6).

[25] Meddeber M, Yagoubi B. Dependent tasks assignment and data consistency management for grid computing. Multiagent and Grid Systems 2019; 15(2): 179–196.

[26] Bakhshad, S., NOOR, R., Akhunzada, A., Saba, T., AHMEDY, I.A.B., Haroon, F. and Nazir, B., 2018. A Dynamic Replication Aware Load Balanced Scheduling for Data Grids in Distributed Environments of Internet of Things. Adhoc & Sensor Wireless Networks, 40.

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

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

[29] G. Camaron, A.P. Millar, C. Nicholson, R.C. Schiaffino, F. Zini, K. Stockinger, "Analysis of scheduling and replica optimisation strategies for data grids using optorsim", Journal of Grid Computing 2 (1) (2004) 57–69.

[30] S.M. Park, J.H. Kim, Y.B. Ko, W.S. Yoon, "Dynamic Data Replication Strategy Based on Internet Hierarchy BHR", in: Lecture notes in Computer Science Publisher, vol. 3033,