Optimization
Iranian Journal of Optimization
Volume 13, Issue 2, 2021, 147-160
Research Paper

Online version is available on: www.ijo.rasht.iau.ir

Islamic Azad University
Rasht Branch
ISSN: 2588-5723
E-ISSN:2008-5427

Minimizing Job Execution Time in Data Grid by a Fuzzy Dynamic Replication Algorithm

Mahsa beigrezaei [*1] and Seyedeh Leili Mirtaheri[2]

[1]Department of Computer and Engineering., Yadegar -E- Imam Khomeini (RAH) Shahr-E-Rey Branch, Islamic Azad University, Tehran, Iran
[2]Assistant Professor of Computer Engineering, Kharazmi University, Tehran, Iran

Revise Date: 18 November 2020
Accept Date: 19 December 2021

Abstract

Data replication is a well-known method for improving performance parameters such as data access time, availability, and load balancing in Data Grids. A novel dynamic algorithm is proposed that uses fuzzy inference systems to manage replication for increasing performance. The proposed algorithm uses a new comprehensive set of decision parameters and fuzzy logic in each phase to reduce the inefficiency caused by wrong decisions in different phases. The algorithm uses two fuzzy interfere systems to select an appropriate place for new replication and a less valuable file for deleting when storage space is full. It places the new replica in a suitable site where the file could possibly be requested soon with high probability. It also prevents deleting valuable files using a fuzzy valuation function. OptorSim simulator simulated the algorithm. The results demonstrated that the algorithm is more effective than other replication methods in terms of the number of replications, the percentage of storage used, and the job execution time.

Keywords:
Fuzzy
Grid
Data Replication
distributed system

***Correspondence E-mail**: M.beigrezai@gmail.com

INTRODUCTION

The Data Grid deals with data sharing, collaboration, and performing data-intensive and data-oriented tasks (Mahsa Beigrezaei et al., n.d.). 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 (Chang et al., 2007). Data replication is one of the most important ways to increase performance, data availability, and execution time of data-oriented jobs in various distributed systems such as Data Grid and cloud (Xiong et al., 2018) (Park et al., 2003). Data replication has many advantages. It can decrease data access time and bandwidth consumption. Replication methods 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 .(Tang et al., 2006) The replication strategy has to answer some questions: What file and when should the file be replicated? Where should replication be stored? Which files should be deleted when storage space is not enough?

Replication methods are divided into dynamic and static (Čibej et al., 2005) (Khanli et al., 2011). Static approaches determine the number of replicas and their location at the design phase. After storing the replica, it will have remained until the user deletes it manually; these locations are unchangeable. Thus, static replication is not suitable when the users change their behavior. Besides, dynamic replication creates and deletes copies of files based on users' behaviors (M Beigrezaei et al., 2013) (Nicholson et al., 2008)(Sashi & Thanamani, 2011)(Amjad et al., 2012)(Dang et al., 2007)(Ranganathan & Foster, 2002)(Rajaretnam et al., 2016)(Meddeber & Yagoubi, 2019).Indeed, it makes decisions based on access patterns during the time. Data Grid is a dynamic environment, and its user's requirements access pattern change over time; hence, dynamic replication is more tailored for Data Grid (Dong et al., 2008).

Replication methods consist of three main phases: replica selection, replica placement, and replica replacement. By looking more closely at replication algorithms, we can understand that replication algorithms in each part can be considered as a multi-metric decision problem. The resources have various dynamic and static properties in Data Grid.

Their specifications have direct and indirect effects on the efficiency of decision-making in these environments. Many dynamic replication algorithms have been proposed till now. These methods often assume that Data Grids are homogeneous or consider only one or a limited number of resource specifications as decision-making parameters. This shortcoming can lead to inefficiencies or reduced efficiency of replication algorithms in real Data Grid systems. To achieve maximum efficiency, it is necessary to consider a set of appropriate decision parameters in each part of the replication algorithm. Another difficulty is how to make decisions with multiple criteria (decision parameters) in each part of the algorithm. In each part, the algorithm must select the most appropriate item among existing items. One of the common methods is to evaluate items based on evaluation criteria by a function. The function can be presented innovatively or modeled using a fuzzy-based model. Fuzzy set theory can show real-world knowledge in the face of uncertainty. Fuzzy modeling, which has been considered in a small number of previous replication methods, can determine the effect of each metric individually on valuation. The Fuzzy-based method models valuation function by a fuzzy inferences system. The rules and knowledge of the fuzzy system can be changed with the lowest cost. The use of a fuzzy system can provide a high degree of flexibility. The fuzzy inference system is relatively fast and flexible, and its modeling is not very complex. The mentioned advantage motivates using the fuzzy-based functions to value items based on multi-decision parameters. In order to solve the mentioned problem, in this paper, we proposed a replication algorithm named Fuzzy-Based Replication Algorithm (FBRA). Our proposed method is provided for a Grid with a hierarchical

structure. FBRA algorithm tries to replicate files within a region and stores the replica in a suitable site with the maximum probability of future access location. For finding the best place for replication and the best replica for deleting, the FBRA method uses two fuzzy inference systems. The algorithm selects the location most likely to receive file requests in the future. Indeed it predicts future file access base on file access history in each site. The main contributions in this paper are as follows:

- Our proposed algorithm improves performance by considering a new set of more effective and comprehensive decision-making parameters in each phase.
- Due to the fuzzy decision parameters and the power of fuzzy decision-making, we presented a replication algorithm based on fuzzy logic. The proposed replication algorithm in two parts (replacement and placement) with a valuation using fuzzy logic increases decision-making efficiency with several parameters.
- Our proposed data replication algorithm (FBRA) selects the optimal replica when various replicas exist to decrease data access time. It reduces the search time by using the hierarchical search. FBRA considers proper parameters like transfer time, propagation time, waiting time in the storage queue to choose the best replica for remote access. It selects the suitable location for new replicas by considering the last access time, number of access, communication capacity, and transfer time. We present a new metric for estimating the communication capacity of a node. This metric is calculated by summing the bandwidth of the node with the other nodes.
- To evaluate the proposed method, we used the Optorsim simulator. The simulation results show our algorithm can improve the performance of the data grid in terms of the execution time of data-oriented jobs, the number of created replications, and the percentage of storage used.

The other sections of the article are organized as follows. Section 3 presents the related works; in section 3, we describe our proposed method, then in section 4, we evaluate the method and show the results of our simulation. Section 5 concludes the paper and explains future works.

RESEARCH BACKGROUND AND RELATED LITERATURES

Researchers in some previous works have addressed the problem of data replication. We mention some of them below.

In (Dang et al., 2007), the Least Frequently Used algorithm (LFU), and two other economic strategies were proposed to replicate popular files when the file is accessed remotely. Simulation results showed that if free space in the replication site is insufficient, LFU works better than economic and LFU strategies. Another similar replication algorithm called the Least Recently Used (LRU) was proposed. In (Ranganathan & Foster, 2002), LFU and LRU delete the least frequently accessed and least recently used files, respectively. An effective replication method called Bandwidth Hierarchy Replication (BHR) algorithm was presented that works based on the network level in (Park et al., 2003). A hierarchical structure consisting of several regions is assumed for the Grid in BHR method. The sites in the same network region are connected with high bandwidth. BHR significantly reduces the cost of data access by replicating the data required for a job near the place where the job is executed. When there is no space for replication on the selected site, it deletes files with at least another copy in the region. BHR reduces inter-region transmission and enhances performance. In(Sashi & Thanamani, 2011), Sashi and Thanamani presented a modified BHR algorithm called the modified BHR (MBHR). Simulation results show MBHR method improves the performance by selecting a site with the maximum number of accesses in the requesting region. Also, MBHR can prevent unnecessary replications. In (Rajaretnam et al., 2016), Ranganathan and Foster 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 a suitable strategy should be used to increase the performance in each access pattern. In (Cameron et al., 2003), the proposed replication algorithm with some popular job scheduling strategies is evaluated. These popular job scheduling strategies include: including the Shortest Turnaround Time (STT), the least Relative load (LRL), and Data Present

(DP), which were evaluated through various simulations. Two dynamic replication mechanisms, the Simple Bottom-Up (SBU) and Aggregate Bottom-Up (ABU) were presented in (Meddeber & Yagoubi, 2019). This strategy is provided for hierarchical and multi-tire Data Grids. Generally, in the SBU and ABU, bottom-up replication takes place. These methods create replicas base on files' popularity near the requester clients. They replicate the most accessed files in the nearest sites. Because of this reason, the access latency can be improved by SBU and ABU methods.

In .(Chang et al., 2007), Data Grid is consists of a set of clusters. Because the inter-cluster communication cost is less than the intra-cluster communication cost, the transition cost of a file from a cluster to another one is more than the cost of the site-to-site transition of the file within a cluster. The proposed replication algorithm was named Hierarchical Replication Strategy (HRS). HRS tries to keep a copy of the file in the local cluster to use other Grid sites. Hence, the sites placed in the same cluster can use it. Tehmina Amjad and et al. presented a comprehensive survey paper about data replication in the Grid (Amjad et al., 2012). They studied different replication techniques.

In (Rajaretnam et al., 2016), a replication algorithm was presented 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 (Meddeber & Yagoubi, 2019), the authors presented three algorithms: a replication algorithm, a job scheduling algorithm, and a job migration method. The proposed job scheduling 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 ,(Bakhshad et al., 2018) a replication algorithm was introduced that utilizes a hierarchical three-level architecture. It replicates popular files at scheduling time. It also uses a migration method to improve workload. In (Bakhshad et al., 2018), a new fuzzy-based replication algorithm named Fuzzy Replacement Algorithm (FRA) was presented. It uses a fuzzy inferences system for calculating file values and selects low-value replicas for removal and providing free space.

In (Mahsa Beigrezaei et al., 2020), John developed a swarm-based method called D2R-IWD. The proposed method determines the number of replicas for each file based on the number of file accesses, file size, the number of available replicas, and a threshold. It also uses the intelligent water drop algorithm to select the best replica for user requests. In (John & Mirnalinee, 2020), a Markov-based replication method at edge-cloud computing was proposed. A gray Markov chain prediction model calculates the required number of replica for each block of files. The proposed method determines replica location based on the Fast Non-dominated Sorting Genetic algorithm. The algorithm improves response time, write delay, read throughput, and workload balance on nodes. In (Li et al., 2020), a data replication technique called Binary Vote Assignment on Grid (BVAG) was proposed. Binary Vote Assignment on Grid with Checkpoint and Rollback-Recovery Transaction Manager (BVAGCRTM) is used to run the BVAGCR proposed method. The performance of the proposed BVAGCR is compared to standard BVAG in terms of total execution time for a single data replication transaction. The experimental results show that BVAGCR

enhances about 31.65% the BVAG in total execution time in a failure environment. In (Vashisht et al., 2019), a centralized replication algorithm (CRP) is proposed. CRP predicts the number of needed replica for each file in each region. When storage space is not enough for replication, RCP deletes the least recently used file.

In (Vashisht et al., 2019), a new replication method is present to improve cloud storage access efficiency and preserve cloud QoS attributes. This method uses a smart water drop algorithm to manage the replication method and considers file accesses and replica locations.

In , authors studied the joint optimization problem of data and replica placement for data-intensive services in geographically distributed clouds. In response to this problem, the author proposed an overlapping correlation clustering algorithm.

In, two strategies to manage IoT data replication and consistency in Fog infrastructures. The strategies choose for each data, the right replica number, and their location to reduce data access latency and replicas synchronization cost. This is done while respecting the required consistency level.

Resources in Data Grids are heterogeneous. A set of dynamic and static characteristics, directly and indirectly, affects the replication algorithm's performance in the Data Grid system. In Table 1, dynamic replication algorithms are compared in terms of the considered parameters in different phases of the replication algorithm (replica selection, replica placement, and replica replacement). As Table 1 shows, one of the shortcomings of the previous methods is that most of them have considered only one or a limited number of decision-making parameters. This shortcoming can reduce the algorithm's efficiency or lead to proper performance only in certain circumstances. On the other hand, some parameters such as communication capacity, file requests waiting in the storage queue (in replica placement and replacement phase) have not been considered.

Therefore, considering more comprehensive parameters for more effective decision-making is one of the main objectives of this study. Moreover, no works focus on the fuzzy theory in both placement and replacement part of replication decision for grid environment. The Fuzzy-based method models valuation function by a fuzzy inferences system. The use of a fuzzy system can provide a high degree of flexibility. The fuzzy inference system is relatively fast and flexible, and its modeling is not very complex.

Table 1: Comparison of various replica algorithms and considered parameters in replica selection, replica placement, and replica replacement sections (Transfer Delay (TD), storage queue workload (SWL), search overhead (SO), the storage speed (SS), communication capacity(CC), Number of Access to file (NA), Last access Time (LA, future file popularity (FPO), remote access time (RAT), file size (SF), and requested file waiting in the storage queues (RWQ), file size (FS)).

	Replica selection				**Replica placement**				**Replica replacement**			
	TD	**SWL**	**SO**	**SS**	**CC**	**NA**	**LA**	**FDM**	**NA**	**LA**	**RAT**	**FDM**
LFU(Ranganathan & Foster, 2001)	*	-	-	-	-	-	*	-	*	-	-	-
LRU(Dang et al., 2007)	*	-	-	-	-	-	*	-	-	*	-	-
BHR (Park et al., 2003)	*	-	-	-	-	*	-	-	*	-	*	-
HRS (Chang et al., 2007)	*	-	-	-	-	*	-	-	*	-	*	-
MBHR (Sashi & Thanamani, 2011)	*	-	-	-	-	*	-	-	*	-	*	-

FRA (M Beigrezaei et al., 2013)	*	-	-	-	-	*	-	-	*	*	*	*
RPLB (Rajaretnam et al., 2016)	*	-	-	-	*	*	-	-	-	*	-	-
Method in (Meddeber & Yagoubi, 2019)	*	-	-	-	-	-	*	-	*	-	-	-
DRLBS (Bakhshad et al., 2018)	*	*	-	*	-	*	-	-	*	*	*	-
D2R-IWD(John & Mirnalinee, 2019)	*	*	-	-	*	-	-	-	*	-	*	-
CRP(Ubaidillah et al., 2021)	*	-	-	-	-	*	*	-	-	*	-	-
Our proposed method	*	*	*	*	*	*	*	*	*	*	*	*

THE PROPOSED REPLICATION ALGORITHM (FBRA)

Jobs in a Data Grid, often need a large amount of data for execution. If a job does not have its required data locally, it must have remote access to it. Therefore, to minimize access costs, dynamic replication is required. Data replication is an important optimization step to manage large data through replicating data in geographically distributed data stores. In this paper, we propose a fuzzy-based replication algorithm (FBRA). It has three main parts, including replica selection, replica placement, and replica replacement. All part are explained as follow. Algorithm 1 shows the proposed method.

Algorithm 1: FBRA Replication Algorithm (File f , job m , Node i)

```
Begin
If job j in node i request file f, which is not available Then

//Replica Selection Part
  BeReplica= select best replica for remote access to file f based on selection policy
//BeReplica is a replica with minimum TC (remote access time);TC is calculated by equation 1
Access to file F from BeReplica

//Replica placement Part
    BetNode= call FuzzyPlacmentAlgorithm (i, f); // BestNode is suitable place for
replication;
     If f exists in BestNode, Then
          Exit;
    End If
    If BestNode.FreeStorageSize⟩=f.size Then
    Replicate file f to BestNode and exit;
    Else
//Replica Replacement Part
    Call FuzzyReplacemenAlgorithm(BestNode, f); //it calls replacement algorithm to
    provide free space for store new replica
    End if
    End if
End if
End
```

Replica selection

When the user submits a job to the Data Grid, the resource broker gets the job from users, schedules it based on its scheduling algorithm, and finds suitable computing elements for job execution; afterward, the jobs are assigned to computing elements. When a job is executed in the selected computing element, the job can get requested data directly if requested data file stored in the local storage site. Else, the requested data file is accessed remotely. If replicas exist for the requested file in the same region, then FBRA creates a list of candidate replicas and selects the site with minimum remote access time (TC). If the file doesn't exist in the same region, then FBRA algorithm searches within other regions. Searching for the best replica among a large number of replicas would lead to long latency. The FBRA has two steps in the replica selection process. It uses a hierarchical search that decreases searching time. The remote access time TC is obtained in Eq.1. TC represents remote access time to file 'r' from the source site 'a' to destination site 'g'. In this Equation, $size_r$, bw_{th}, DSa, DQD, and PDT_{ag} is the size of file 'r', available bandwidth between grid site 'a'and site 'b', disk speed in site 'a', the waiting time in the disk queue and Propagation delay time for sending file 'f' from site 'a' to site 'g', respectively.

$$TC(file_r, a, g) = \frac{size_r}{\min(DS, bw_{ag})} + PDT + DQD_a \quad (1)$$

Replica placement

When the job requires a file that exists in the local storage, it accesses it locally. If the file doesn't exist, replication takes place. The replicated files are not stored in all the requested sites; instead, the file is stored in the best site where the file will most probably be accessed in the future. The FBRA algorithm in the placement part evaluates the sites in the current region based on a fuzzy-based valuation function (FuzzyFunction (node I, file f)). Then, it selects a site with the highest value (NodeValue) as the best place for replication. In this paper, the valuation function is implemented using a fuzzy inference system that has three input parameters and one output. In fig. 1, an abstraction of the used fuzzy inference system is shown. The output shows the value of the site i. The inputs of the used system are 1) the number of accesses(NA) to the requested file in site i, 2) the available bandwidths (BW), which is calculated for the site i in region c by Eq.2, where n is the number of the sites within region c, and bw_{ij} is the available bandwidth of site i and j; and 3) the last access time to requested file in the site I (TL), which is the difference between the present time and the time of the last access to the file in the site i. Some of the used rules in this system fuzzy inference system are explained in Table 2. In used fuzzy system have 20 identical rules is considered. The pseudo-code of the Replica placement part of the FBRA algorithm is shown in algorithm 2.

$$BW_c = \sum_{1}^{n} bw_{ji} \quad (2)$$

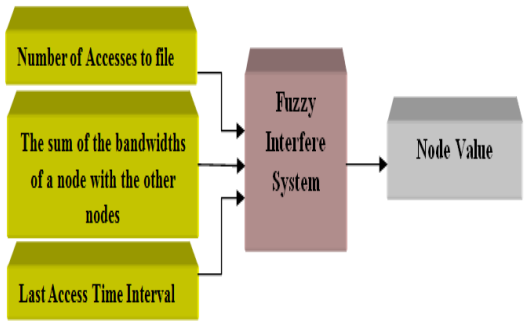


Fig. 1. An abstraction of the used fuzzy inference system for determining the value of a site.

Replica replacement

After selecting the appropriate location for replication, a new replication is created if enough free storage space is available on the selected site. Else, the replacement algorithm is called to provide enough storage space.

Our proposed replacement algorithm provides free space in the site by valuing and deleting low-value files that have low remote access costs from the selected site. We calculate file value (FileValue) using another fuzzy function Eq. 3 with two input parameters, including the last

access time interval (TI) and the number of file access(NA). Fig.2 shows an abstraction of the used fuzzy inference system. TI is the time between the current time and the last replica access time. It shows a fuzzy system with 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 3. For example, rule 1 indicates that if NA is high and TI is low, then the weight of the replica is very high.

$$\text{FileValue}(f,g) = RepFuzzyFunction(TI, NA) \quad (3)$$

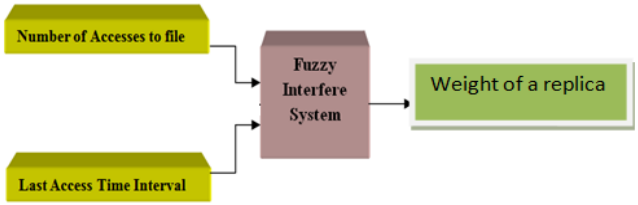


Fig. 2. An abstraction of a second used fuzzy inference system

Table 3: The Rules used in the fuzzy inference system (replacement part).

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

Algorithm 3: FuzzyReplacementAlgorithm (Node g, File f)

Begin
Return the recent // recentAccessHistoryList= getRecentAccessHistory (long dt); access history
For each replica file "f" in storage Node g **Do**
If a is removable and has another replica in Region, **Then** //a is replica file "f"
a.value= $RepFuzzyFunction(TI, NA)$ // it calculates value of file 'f' by Eq (3)
FilesIsInRigon.add(a); //add replica a in the FilesIsInRigon list
Else
FilesIsNotRigon.add(a);
End If
End For
Sort FilesInRigon in ascending order according Its File value;
while "FilesInRigon" != empty **Do**
Select a replica from top of the "FilesIsInRigon" and delete it from grid node 'g';
If g.availableStorageSize>=f.size **Then**
replicate file 'f' to grid node 'g' and exit;
End While
Sort FilesIsNotRigon in ascending order according Its File value using $RepFuzzyFunction(TI, NA)$; ;
while FilesIsNotRigon != empty **Do**
Select a replica from top of the "FilesIsNotRigon" list and delete it from grid node 'g' and list;
IF g.availableStorageSize>=f.size **Then**
replicate file 'f' to grid node 'g' and exit;
End While

End

In algorithm 3, the replacement part of the proposed replication algorithm is shown. In this part, the algorithm lists all sites' replicas that have at least one other copy in the region in the candidate list (FilesIsInRigon) and calculates the value of each candidate replicas (FileValue) using Eq.3 (second fuzzy function). It sorts the candidate list based on the value of the file in ascending order. Then replica is selected and deleted from the beginning of the list until enough space is provided. If free space is sufficient, replication is done. Otherwise, the remaining replicas are listed in the second candidate list (FilesIsInNotRigon), are valued using Eq. 3, and are sorted based on file value. Low-value files are then deleted until enough space is created. After that, the new replica is stored. Needed information like the history of access to files on each site and the bandwidths in each region is stored in the region header.

EVALUATION RESULTS

In this part, first OptorSim as simulation tool is explained, then simulation configuration input file and eventually, simulation result and discussion are given.

Simulation tools

To study the grid environment and evaluate different optimization algorithms as part of the European DataGrid (EDG) project, the grid simulator, OptorSim, was developed and used to simulate different grid projects as CMS (Fuhrmann et al., 2005)(Naas et al., 2021). Other simulation tools such as MicroGrid, Briks, SimGrid, GridSim for evaluating replica optimizer algorithms were also developed. The focus of OptorSim is not only, like the other simulators, on scheduling, but also, different from the others, on replication. We have also used this simulator for the simulation and performance evaluation of our proposed method.

Simulation network topology

Grid architecture in our proposed algorithm has a hierarchical architecture and consists of a set of regions. Fig. 3 shows the grid topology in our simulation, which consists of two regions. Sites in the regions have computational or storage elements. Most of the existing replication strategies assume data is read-only. We also assume data is read-only. The used simulation parameters are shown in Table 4. We evaluated our method for the random ZipF access pattern. Access patterns determine the order in which files are accessed in the jobs.

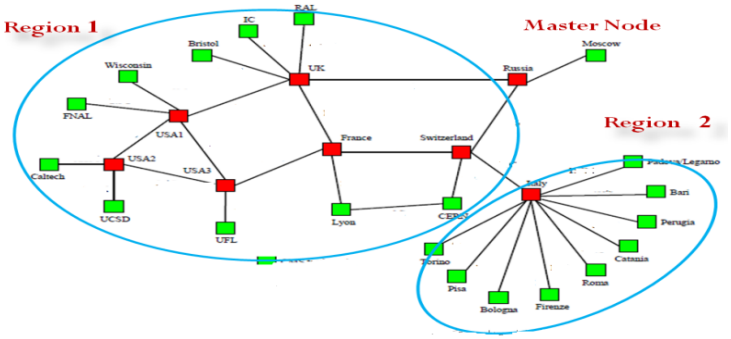


Fig. 3. The network topology

Table 4: The configuration parameters in the simulation

Parameter	*Value*
Number of sites(site)	20
Number of the computing element(CE)	18
Number of storage elements(SE)	20
Number of jobs	100
access pattern	Random ZipF
Minimum bandwidth between two sites(Mbit/s)	600
Maximum bandwidth between two sites(Mbit/s)	2000
Number of experiments	20

To implement the fuzzy inference systems, we used the fuzzy toolbox of MATLAB software. To connect Optorsim to MATLAB software, we added several classes to Optorsim. The input parameters of the fuzzy inference systems are determined during the simulation. The properties of used fuzzy inference systems like the inference engine, DEfuzzy Method, Implication Method, and Aggregation Method have been specified in Table 5.

Type	Mamdani
AndMethod	min
OrMethod	max
DefuzzMethod	centroid
ImplicationMethod	min
AggregationMethod	max

Table 5: The properties of the fuzzy inference system for determining the weight replica

SIMULATION RESULTS

In this part, our simulation results and discussion is mentioned. The proposed algorithm with the no-replication, LRU, FRA, and modified BHR, algorithms are compared.

The mean job time of all jobs on the Grid

Mean job time can be considered the most important measure for evaluating an algorithm's performance. This criterion, defined as the ratio of the total execution time of all the jobs to the number of the jobs. The lower the execution time average is, the better the algorithm will be, as the jobs are executed within a shorter period of time. It is obtained using Eq(4), where $tj(st)$, $tj(ct)$, and NOJ are the time that job j is submitted, the time that job j is completed, and the total number of the jobs.

$$MJET = \frac{\sum_{i=1}^{NJ} (t_j(ct) - t_j(st))}{NOJ} \qquad (4)$$

In Fig. 4, the replication algorithms with the random ZipF access pattern are evaluated Fig.4 indicates that the mean job execution time for the presented method is the lowest among the others. Because the FBRA method predicts the sites' future needs based on their previous access, it replicates their next needed files before they ask them. Indeed, the FBRA method predicts the future site getting access to the file, and consequently, the files are more probably locally available to the jobs at the course of execution. Since the required files' unavailability is one of the main factors increasing the job execution time, this time is considerably lowered in our proposed algorithm. Moreover, it can also put this way that replication in our method (FBRA) is done more reasonably and correctly than the other methods. The right site is selected based on more parameters with a more rigorous logic compared with the other methods, particularly the MBHR and FRA method, which only takes the number of accesses into account to select the right replication site. Therefore, the files are more probably locally and/or with lower cost available to the jobs during the execution, and the site where the file is replicated will be more probably the site requiring the file in the future; hence, the mean job execution time is reduced.

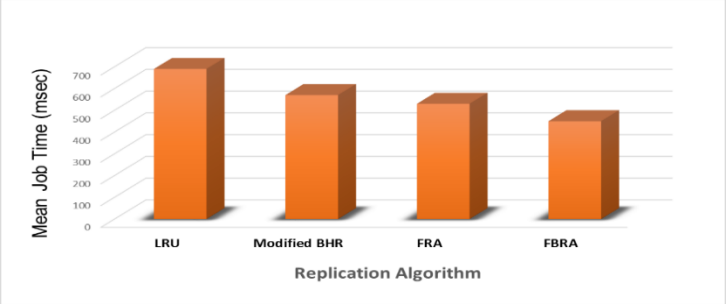


Fig. 4. Mean job Time of different replication methods with random ZipF access pattern

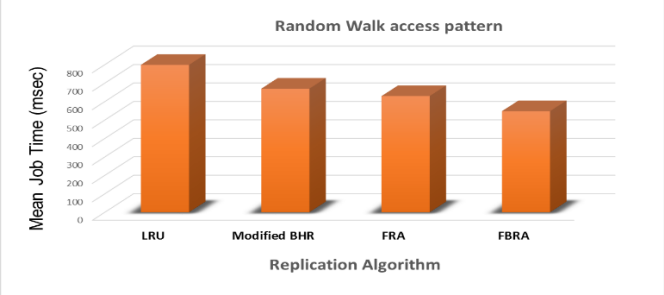
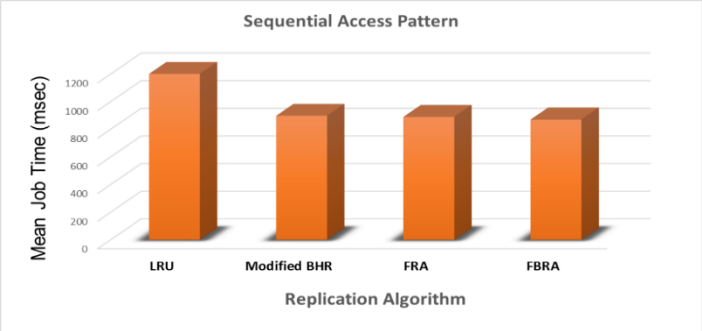


Fig. 5. Mean job Time of different replication methods with Random walk access pattern

Fig. 6. Mean job Time of different replication



methods with Sequential access pattern

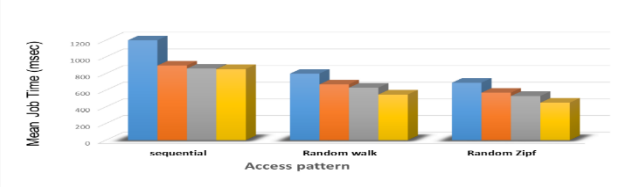


Fig. 7. The Mean job execution time with different access pattern generator

In Fig. 5, the replication algorithms with the random walk access pattern are evaluated. The results show that the proposed method gave the best result and the LRU method gave the worst result. In Fig. 7, the replication algorithms with the sequential access pattern are evaluated. In the sequential access pattern, the files are requested in a specific order. This Fig. shows that the proposed algorithm is not significantly more successful than the existing methods. In the access pattern, due to the non-repetitive access, the amount of remote access is higher. For further improvement, sequential pattern prediction and pre-replication methods should be used to access more requests locally at the time of request.

Fig. 7 gives the mean job time of four different replication algorithms with sequential, random walk, and random ZipF access patterns. This experiment shows, the NDDR algorithm has minimum mean job time in all access patterns. Through the analysis of experiments, it can be concluded FBRA strategy and also all the other strategies have more improvement for the random walk and random Zip file access patterns. In these access patterns, a particular set of files is more likely to be requested by grid sites, so a large percentage of needed files have been replicated. Besides, Fig. 7 shows that LRU and LFU have the worst performance in the three access patterns.

Total number of replications

During replication, storage space and bandwidth are consumed. Therefore, Data replication is costly. As a result, the lower the Total number of replications, the lower the cost to the system. On the other hand, the low number of replications indicates that the number of local accesses in the algorithm is higher. The number of replication operations must be limited to avoid overheads of replication. Therefore, a good algorithm has a low number of replications. As is obvious in Fig. 8, the FBRA algorithm performs better than other methods because replicas are placed in a suitable location. Indeed FBRA algorithm is more successful in finding a suitable file for replication and a tailored place for replication. Thus, the number of local access increase and the number of replication decrease. We should mention that the noRep (No replication) method doesn't replicate. So as shown in Fig. 8, the number of replications is zero.

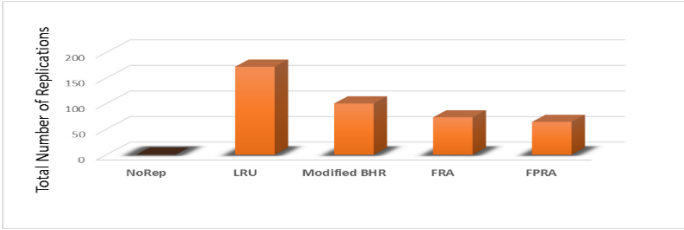


Fig. 8. Total Number of Replications of various replication algorithms

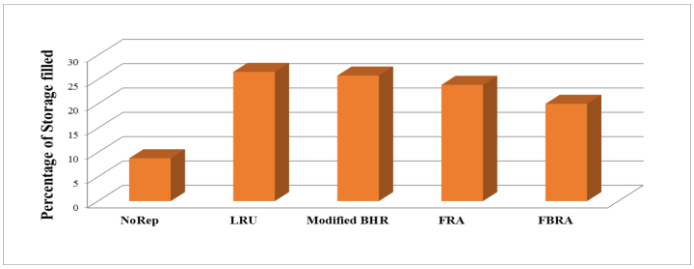


Fig. 9. Percentage of Storage filled

The percentage of storage used

The percentage of storage metric represents the average percentage of storage space used to store files and replicas in the grid sites. Fig. 9 shows that noRep (No Replication) method has the lowest percentage of storage used because it does not do any replication and only saves the original files. After noRep, our proposed method has the best percentage of storage used. In the proposed method, selecting the best place of replication and preventing the removal of valuable replicas reduces the need for data replication. As a result, storage consumption is reduced compared to LRU, MBHR, and FRA.

The Effective Network Usage (ENU)

Effective Network Usage (ENU) is a well-known metric for evaluating the efficiency of network resource usage. It is obtained from Eq. (5). It is defined as the ratio of the number of transferred files to the number of files requested. A low Effective Network Usage shows that the replication algorithm used is better at putting files in a suitable place.

$$ENU = \frac{N_{remote\ file\ accesses} + N_{file\ replications}}{N_{remote\ file\ accesses} + N_{local\ file\ accesses}} \quad (5)$$

In Fig. 10, the FBRA algorithm is compared with the other three algorithms in terms of the effective network usage criterion. AS shown in Fig. 10, the FBRA method improves ENU more than the Modified BHR and FRA methods. Because the FBRA algorithm reduces the number of remote accesses to the files, it lowers the need for their replication by placing the files in the sites with a higher probability of future access. Consequently, in the FBRA method, the jobs at the course of execution will probably have the required files locally available. Hence, the number of local accesses increases, and the number of replications reduces, lowering the ENU.

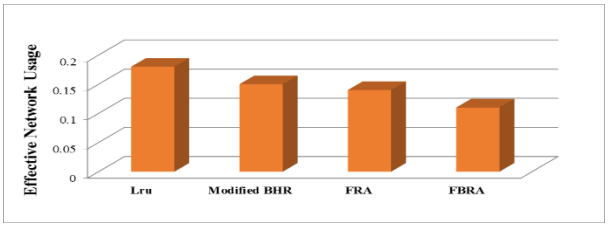


Fig. 10. Effective Network Usage of various replication algorithms

CONCLUSION

Recently, with the advancement of technology and the production of large amounts of data by new applications, the popularity of distributed systems such as Grid and Cloud is increasing. These systems provide the infrastructure for sharing and managing large amounts of huge data. Delay in accessing data is one of the major problems in Grid and cloud. Data replication is used as a common method to solve this problem. In this paper, we present a replication algorithm. The proposed algorithm uses the fuzzy decision to select the best place for replication and the best replicas to delete. Also, the parameters of the access number, output bandwidth on the site, and the latest file access time are used as fuzzy decision parameters. The proposed algorithm was compared with well-known replication algorithms using optimism simulation. Results show our proposed algorithm can improve performance parameters compared to LRU, Modified BHR, and FRA methods. Indeed, the proposed algorithm improves the performance by considering more efficient parameters in the replica placement and replacement phases and using the fuzzy system analysis power. We intend to use more decision parameters to predict the future file value for deletion in future works. And evaluating the proposed method in a real Grid is one of our next plans. We also plan to evaluate the combination of the proposed method with the new and popular job scheduling algorithms.

REFERENCES

Amjad, T., Sher, M., & Daud, A. (2012). A survey of dynamic replication strategies for improving data availability in data grids. *Future Generation Computer Systems*, *28*(2), 337–349.

Bakhshad, S., NOOR, R., Akhunzada, A., Saba, T., AHMEDY, I. S. M. A. I. L. B. I. N., Haroon, F., & 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*.

Beigrezaei, M, Haghighat, A. T., & Kanan, H. R. (2013). A new fuzzy based dynamic data replication algorithm in data grids. *2013 13th Iranian Conference on Fuzzy Systems (IFSC)*, 1–5. https://doi.org/10.1109/IFSC.2013.6675676

Beigrezaei, Mahsa, Haghighat, A. T., Hajizadeh Bastani, N., & Saadati, M. (2020). Increasing performance in Data grid by a new replica replacement algorithm. *Journal of Advances in Computer Research*, *11*(2), 1–10.

Beigrezaei, Mahsa, Toroghi Haghighat, A., & Leili Mirtaheri, S. (n.d.). Minimizing data access latency in data grids by neighborhood-based data replication and job scheduling. *International Journal of Communication Systems*, e4552.

Cameron, D. G., Carvajal-Schiaffino, R., Millar, A. P., Nicholson, C., Stockinger, K., & Zini, F. (2003). Evaluating scheduling and replica optimisation strategies in OptorSim. *Proceedings. First Latin American Web Congress*, 52–59.

Chang, R.-S., Chang, J.-S., & Lin, S.-Y. (2007). Job scheduling and data replication on data grids. *Future Generation Computer Systems*, *23*(7), 846–860.

Čibej, U., Slivnik, B., & Robič, B. (2005). The complexity of static data replication in data grids. *Parallel Computing*, *31*(8–9), 900–912.

Dang, N. N., Lim, S. B., & Yeo, C. K. (2007). Combination of replication and scheduling in data grids. *International Journal of Computer Science and Network Security*, *7*(3), 304–308.

Dong, X., Li, J., Wu, Z., Zhang, D., & Xu, J. (2008). On dynamic replication strategies in data service grids. *2008 11th IEEE International Symposium on Object and Component-Oriented Real-Time Distributed Computing (ISORC)*, 155–161.

Fuhrmann, P., Petravick, D., Bakken, J., Perelmutov, T., Fisk, I., Mkrtchyan, T., & Ernst, M. (2005). *Managed Data Storage and Data Access Services for Data Grids*.

John, S. N., & Mirnalinee, T. T. (2019). A novel dynamic data replication strategy to improve access efficiency of cloud storage. *Information Systems and E-Business Management*, 1–22.

John, S. N., & Mirnalinee, T. T. (2020). A novel dynamic data replication strategy to improve access efficiency of cloud storage. *Information Systems and E-Business Management*, *18*(3), 405–426.

Khanli, L. M., Isazadeh, A., & Shishavan, T. N. (2011). PHFS: A dynamic replication method, to decrease access latency in the multi-tier data grid. *Future Generation Computer Systems*, *27*(3), 233–244.

Li, C., Song, M., Zhang, M., & Luo, Y. (2020). Effective replica management for improving reliability and availability in edge-cloud computing environment. *Journal of Parallel and Distributed Computing*.

Meddeber, M., & Yagoubi, B. (2019). Dependent tasks assignment and data consistency management for grid computing. *Multiagent and Grid Systems*, *15*(2), 179–196.

Naas, M. I., Lemarchand, L., Raipin, P., & Boukhobza, J. (2021). IoT data replication and consistency management in fog computing. *Journal of Grid Computing*, *19*(3), 1–25.

Nicholson, C., Cameron, D. G., Doyle, A. T., Millar, A. P., & Stockinger, K. (2008). Dynamic data replication in lcg 2008. *Concurrency and Computation: Practice and Experience*, *20*(11), 1259–1271.

Park, S.-M., Kim, J.-H., Ko, Y.-B., & Yoon, W.-S. (2003). Dynamic data grid replication strategy based on internet hierarchy. *International Conference on Grid and Cooperative Computing*, 838–846.

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

Ranganathan, K., & Foster, I. (2002). Decoupling computation and data scheduling in distributed data-intensive applications. *Proceedings 11th IEEE International Symposium on High Performance Distributed Computing*, 352–358.

Ranganathan, K., & Foster, I. (2001). Identifying dynamic replication strategies for a high-performance data grid. *International Workshop on Grid Computing*, 75–86.

Sashi, K., & Thanamani, A. S. (2011). Dynamic replication in a data grid using a modified BHR region based algorithm. *Future Generation Computer Systems*, *27*(2), 202–210.

Tang, M., Lee, B.-S., Tang, X., & Yeo, C.-K. (2006). The impact of data replication on job scheduling performance in the Data Grid. *Future Generation Computer Systems*, *22*(3), 254–268.

Ubaidillah, S. H. S. A., Alkazemi, B., & Noraziah, A. (2021). An Efficient Data Replication Technique with Fault Tolerance Approach using BVAG with Checkpoint and Rollback-Recovery. *International Journal of Advanced Computer Science and Applications*, *12*(1).

Vashisht, P., Kumar, V., Kumar, R., & Sharma, A. (2019). Optimizing Replica Creation using Agents in Data Grids. *2019 Amity International Conference on Artificial Intelligence (AICAI)*, 542–547.

Xiong, L., Yang, L., Tao, Y., Xu, J., & Zhao, L. (2018). Replication strategy for spatiotemporal data based on distributed caching system. *Sensors*, *18*(1), 222.