VOL. 14, NO. 1, JANUARY 2019 ISSN 1819-6608
ARPN Journal of Engineering and Applied Sciences

www.arpnjournals.com

A HYBRID APPROACH FOR IMPROVING ACCESSIBILITY IN DATA GRID ENVIRONMENT USING DYNAMIC REPLICATION AND CONSISTENCY METHODS

Bhuvaneswari R.[1] and Ravi T. N.[2]
[1]Manonmaniam Sundaranar University, Tirunelveli, India
[2]Department of Computer Science, Periyar EVR College (Autonomous), Trichy, Tamil Nadu, India
E-Mail: rbeswari17@gmail.com

ABSTRACT

Data Grid is a geographical-based distributed environment that deals with the extensive data-oriented applications. Data replication is used for reducing the data access latency and managing effective processing of huge data in the distributed environment. This research work presents a hybridized Modified Dynamic Hierarchical Replication Algorithm (MDHRA) and Efficient Replica Consistency Model (ERCM) for the data grid environment. The location of the best file replica is chosen based on the data transfer time, storage access latency, replica requests in the queue and distance between the nodes in the data grid environment. The ERCM model requires minimum execution time for the reading and writing operations that ensures high data availability. The main contributions of the ERCM approach are optimal allocation of replica and maintenance of the replica consistency. The experimental analysis shows that the combined approach requires minimum Effective Network Usage (ENU), storage usage, replication frequency, number of communications and maximum hit ratio than the existing replication techniques.

Keywords: data availability, data grid, data replication, dynamic hierarchical replication, effective network usage, replica consistency, replica management, replica placement.

1. INTRODUCTION

Nowadays, huge amount of data is generated around the world in the information technology enterprises, scientific field and engineering applications. Hence, management of large distributed data resources becomes more necessary. Data Grid is a solution for this issue. The grid can be classified as computational grids that require minimum data and data grids that deal with the applications that require analysis of the massive datasets [1-6]. Data grids combine a collection of distributed resources across the globe for enabling the users to share the data and computing resources [7].

A distributed system is a group of computing devices that appear as a single computing device to the users [8]. Two types of distributed system are grid and Peer-to-Peer (P2P). Grids can provide replication transparency that allows replication of software/data on multiple machines invisibly. In the P2P-based applications, data replication and caching are applied to improve the scalability and network performance. But, least attention has been paid to maintain the data consistency in the P2P systems [9]. In the P2P networks, data replication is a solution to ensure availability of a file when required and load-balance. Replica integrity is a main requirement. Hence, it is necessary to manage update of the replicas to maintain the consistency of the replicas and propagate the update messages containing information about the updates on the network [10]. The consistency of the replica is essential only in each peer. When a read or write operation is issued by a peer, the operation is performed to a replica held by the peer [11].

Though, the memory and storage size of the computers are increasing every year, there is a need to cope up with the request for storing huge data. Data replication is the process for managing enormous amount of data in the database by replicating the data stored in various replica sites. The main challenge is to decide the number of data replicas to be created and store the replicas. There is a need to develop new methods for creating the replicas that increase the availability without requiring unnecessary storage and bandwidth capacity. As it is highly expensive to maintain a huge number of data replicas, the number of data copies should be restricted. Apparently, finding a proper balance between the optimization of the access cost of the data requests and reduction of the replication cost is an important task. Each grid site has its own capabilities and characteristics. The process of selecting an appropriate site is an important decision. In the data grid environment, reduction in the turnaround time of job depends on the resource for assigning the job and location to obtain the required data files.

This paper presents a hybridized MDHRA-ERCM approach to improve the file access time. The MDHRA strategy maintains the valuable data replicas only and replaces the less significant replicas. The files are deleted when it is found that the free space is not sufficient for the new replica. Due to the storage space issue in the data grid environment, the size of replica is a valuable parameter in deciding whether the replica should be stored or not. But, the MDHRA approach does not address the replica consistency issues. To solve this issue, the ERCM is applied to maintain the consistency of replica. The main contributions of the ERCM approach are optimal allocation of replica and maintenance of the replica consistency. The aim of the ERCM approach is to reduce the job execution time in the data grid environment and ensure replica consistency.

The paper is well thought-out in the following order: Section II provides a summarizing overview of the prevailing data replication techniques in the data grid and cloud environments. The MDHRA and ERCM model are explained in Section III. Section IV presents the comparative analysis of the MDHRA and ERCM with the existing replication techniques. The concluding observations of the proposed research work are conferred in Section V.

2. RELATED WORKS

Replica management is a trending research topic in the data grid. By placing multiple replicas at different locations, replica management can reduce the network delay and bandwidth consuming of remote data access, help resolve load-balance of network, and improve the data security and reliability, and system resilient. A well-designed replica management is a key factor of improving the quality of service of data grid. Replica management includes replica creation, selection, replacement and maintenance. The replica management strategy is divided into simple, hierarchical model based and economical model based management strategies.

Edwin *et al.* [12] developed an Efficient and Improved Multi-Objective Optimized Replication Management (EIMORM) algorithm for stabilizing the replication optimization objectives. An efficient Knapsack algorithm is implemented to optimize the replication cost. The EIMORM algorithm can dynamically adapt to the preferences of the user. But, the EIMORM strategy did not determine the popularity of the data file in the replication process. Sun *et al.* [13] devised a Dynamic Adaptive Replica Strategy (DARS) for the high load Cloud-P2P environment based on the overheating similarity of nodes. By applying the Fuzzy membership function, the node can create the replica before the occurrence of overloading. The proposed DARS obtains superior access latency performance and better load balance under high load condition. However, it did not studied about the ways for replica replacement in the storage restriction.

Zhao *et al*. [14] presented an enhanced dynamic replica creation strategy based on the file heat rate and node load rate in the hybrid cloud environment. The number of copies is adjusted based on the average heat and load value. This strategy is highly sensitive to the replica access frequency to adaptively adjust the number of copies, reduce the average response time and achieve better load balance of the cluster. The weakness of this method is the absence of efficient replacement method. Mansouri *et al.* [15] introduced a Dynamic Popularity based Replication Strategy for replicating only a minor amount of frequently requested data file in the cloud system. This enhances the overall performance of the cloud system. But, this proposed strategy neglects the effect of file access pattern in the file replica choice.

Elango and Kuppusamy [16] developed Fuzzy FP-tree based mining algorithm for managing the data replication in cloud. Our proposed data replication method delivers better data access time without much delay due to the efficiency of using membership values to determine the frequency of each data itemset. But, this algorithm it did not determine which file to be replaced when the storage is full. Tos *et al.* [17] proposed a data replication strategy to satisfy the Service Level Agreement (SLA) objectives for the tenant and increase the benefits of the cloud provider. The number of data replicas is dynamically adjusted by determining whether the SLA objectives such as availability and performance are satisfied or not. This guarantees the better cost-effectiveness of the cloud service provider. The proposed strategy did not considered the queries that include dependent operations such as join operations.

Liu *et al.* [18] developed a two-level auditing architecture for regulating the operations in an audit cloud to confirm whether the data cloud offers the assured consistency to the users. A Heuristic Auditing Strategy (HAS) is devised to reveal the potential consistency breaches in the cloud. Field and Sakellariou [19] introduced a method for maintaining a replica of all information from the information source and a prototype implementation based on the enterprise messaging system. But, this method suffers due to the bootstrapping issue and fails to ensure synchronization in the event of failures. Ahangaran and Rahmani [20] proposed a Tree-Based Clustering method to organize the replicas having similarity into the hierarchical clusters and choose the shortest connection to reduce the communication cost, average response time and total data transferred through the data grid environment. This method comprising the synchronous and asynchronous approach maintained the replica consistency. However, data transfer is not better due to restricted single editable replica in one-way method.

3. MDHRA-ERCM

Data replication is the process of copying data from one server to another server for possessing the up-to-date data replica during the occurrence of disaster. Data replication is an essential tool for effectively managing a database in the distributed database environments. Maintaining the data replicas at the numerous sites improves the overall performance of the distributed environment by reducing the remote access delay and mitigating the single point of failure. But, there is a need for the storage devices to maintain the replicas and need to synchronize the data replicas. Any changes made at one of the replica sites should be reflected at other sites.

Dynamic Hierarchical Replication (DHR) strategy [21] initially checks the feasibility of the replica. The files are accessed remotely, if the size of the file is greater than the size of the Storage Element (SE). Among the replicas, the replica having highest bandwidth and least number of requests is selected. The best replica site having maximum number of access is also placed. If the available space in the best SE is greater or equal to requested file size, the file is replicated. Otherwise, the files that require minimum time for transfer should be deleted. MDHRA replicates the files within the region in a site where file has the highest access. Initially, the user is initialized and file is uploaded. Then, the data center is created. The Virtual

Machines (VMs) are created using the CloudSim. The best virtual machine is chosen based on their waiting time, storage capacity and process capacity. The file is converted into the data packets of equal length.

A unique ID is assigned to each file. The replica of each packet is obtained and the best VM for the replicas is chosen for storing them to avoid hacking. The read and write operations are performed in a serialized manner. When a user reads the replica, writing or updating the replica is not allowed. The files are updated based on the file ID. The file received from the first user is stored along with the ID. The original and replica data are stored in the VM based on the ID. During the modification process, the data is retrieved, updated and stored in the VM based on the ID. The packet in all replicas and original packet stored in the VMs are modified. The files are downloaded when requested by the user.

MDHRA-ERCM procedure
Step 1: Initialization of user

Step 2: File uploading

Step 3: Datacenter creation

Step 4: Creation of virtual machines using CloudSim

```
For (int i = 0; i ≤ VMcount; i + +)
{
        Creation of virtual machine;
}
```

Step 5: Choosing best virtual machine based on waiting time, storage capacity and process capacity

Step 6: Convert file into data packets of equal length

Step 7: Get the replica of the packet

Step 8: Choose best VM for the replica

Step 9: Store the replica in different VM to avoid hacking

Step 10: Update/Modify the files

Step 11: Modify the packet in all replica and original packet in VMs

Step 12: Download the files when requested by the authorized user

A. Hierarchical grid topology

Figure-1 depicts the three-level hierarchical grid topology. In the primary level, the regions are connected through internet i.e. low bandwidth. Secondary level comprises Local Area Network (LAN) within each region having higher bandwidth than the primary level. In the tertiary level, the nodes are located within each LAN that is connected to each other having a higher bandwidth.

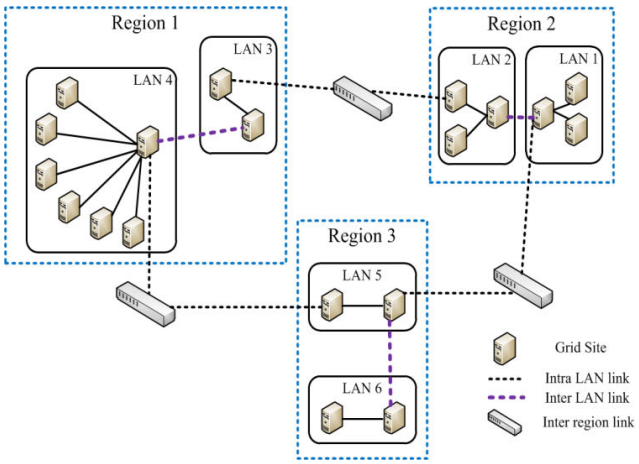

Figure-1. Grid topology

B. Modified dynamic hierarchical replication algorithm

a) Replica selection

Replica selection determines the best replica location for the data grid users. This is really useful when different sites have data replicas. Four factors used for choosing the best replica are termed below [22]

- Storage access latency
- Waiting time in the storage queue
- Inter node distance
- Central Processing Unit (CPU) process capacity

Storage access latency

The storage media speed and size of requests play a main role in the calculation of average response time T_1

$$T_1 = \frac{File\ size\ (MB)}{Storage\ speed\ (MB/s)} \quad (1)$$

Waiting time in the storage queue

Each storage media could execute only one request at a time. Hence, all the previous requests should wait in the storage queue for a time T_2.

$$T_2 = \sum_{i=0}^{n} T_1 \quad (2)$$

Where 'n' is the number of requests waiting in the queue.

Distance between nodes

$D(x, y)$ denotes the network distance between the nodes 'x' and 'y'. The information about the network distance can be stored to reduce the cost, when a replica is checked for the first time.

CPU process capacity

The process capacity of the CPU is defined as

$$C = F \times P_{CPU} \quad (3)$$

Where 'F' represents the CPU frequency and P_{CPU} denotes the usage capacity of the CPU.

$$F(x,y) = w_1 \times T + w_2 \times D(x,y) + w_3 \times C \quad (4)$$

The overall time is computed by adding the average response time and waiting time

$$T = T_1 + T_2 \quad (5)$$

This function can be personalized, as it is a weighted combination of these metrics. Taguchi method [23] is used for computing the weight values $w_1, w_2\ and\ w_3$. Taguchi method is applied in the field of Computer Aided Design (CAD), banking applications, service oriented sectors and manufacturing applications. It evaluates and improves the products, practices, resources, devices and facilities by improving the desired characteristics. The number of defects in the manufacturing field is reduced simultaneously by evaluating the key variables and increasing the procedures or design to yield the best results. Taguchi proposed a regular eight-step procedure to optimize any procedure. The objective of the Replica Manager (RM) in each site is to minimize the file transfer time for each job. The RM locates remote requested files and controls the presence of the file in the local LAN. A list of replicas is generated and a data replica with the minimum cost is chosen, if the file lying in the same LAN is found to be replicated.

If the file exists in different LAN, the proposed MDHRA algorithm searches for the data replica lying within the local region. Otherwise, a list of replicas that are available in other regions is generated and replica with minimum cost is selected from this list.

Taguchi procedure

Step-1: Identify the main function, side effects and failure mode

Step-2: Identify the noise factors, testing conditions, and quality characteristics

Step-3: Identify the objective function to be optimized

Step-4: Identify the control factors and their levels

Step-5: Select the orthogonal array matrix experiment

Step-6: Conduct the matrix experiment

Step-7: Analyze the data; predict the optimum levels and performance

Step-8: perform the verification experiment and plan the future action

b) Replica placement

The file replication is initiated when a requested replica is not available in the local storage unit. The replica is located in the best site in accordance with the sequential and topographical locality. The proposed algorithm creates a sorted list of all storage elements that request the specific file in the region, to choose the best replica site. The replica is positioned in the first storage element of the sorted list, if more than one storage element is selected in a random way. Thus, the replica is not positioned in all the requested sites. This results in the significant reduction of the storage cost and average job execution time. The proposed MDHRA algorithm finds the best site for storing the replica.

c) Replica management

The selected file is duplicated, if there is no sufficient space in the best replica site. Otherwise, if the file is available in the local LAN, the file is accessed remotely. If there is no abundant space for the file replication and the requested file does not available in the same LAN, the data files should be deleted based on the following rules:

- Create a LRU sorted list of the file replicas available on the replica site and local LAN. Currently, start deleting the files from the sorted replica list until the adequate storage space is available for the replica.
- If the space is still insufficient then the above step is repeated for each LAN in the current region in a random way. A LRU sorted list of the file replicas available in the replica site is created and a local region is generated.
- If the space is still inadequate, there is a need to delete a group of file replicas. But, certain valuable files may get deleted using the LRU. This will result in the increase of file transfer cost. In this step, three factors such as number of access, replica file size and last time the replica was requested are considered instead of LRU. These factors describe a sign about the probability of requesting the file replica another time. Clearly, it is more important to replace the large size file, as it can reduce the number of replica replacement. The replica value is computed using the following equation

$$RV = w_1 \times \frac{1}{S} + w_2 \times NA + w_3 \times \frac{1}{CT-LA} \quad (6)$$

Where the value of $w_1, w_2\ and\ w_3$ can be assigned by the users. ‘S’ represents the file size, ‘NA’ denotes the number of access to the file replica, ‘CT’ indicates the current time and ‘LA’ shows the last request time of the file replica. The replica with the minimum value of RV will be replaced with the new replica.

C. ERCM architecture

The ERCM architecture [24] shows the geographical distribution of the peers in P2P network and replica sites. The peer contains the original data and responsible for the transmission of updates to all regions in the data grid environment. For each region, the local consistency service coordinates the data operations for the nodes that they are located. Figure-2 shows the ERCM architecture and characterized by

- Optimal allocation of replicas
- Replica site
- Readable unit
- Writable unit
- Update propagation operation

a) Optimal allocation of replicas

The data grid environment is divided into multiple geographical regions, where the user requested a replica from the neighbouring replica. If the replica on the local service proximate to the user, it is fetched from the peers and maintained in the nearest storage site to the user for future purpose. If the user requested for new replicas, the historical record of the proximate replica site should be deleted, when it exceeds the threshold level of the storage for creating space to store newly requested replicas.

b) Replica site

The replica sites involve the following elements.

Controller: It is responsible for directing the requests to readable or writable operations and manages the record operation for local replica.

Local Consistency Service (LCS): It is responsible for replica consistency and implementation of user requests for local replicas.

Readable unit: It receives read requests of the user through the controller and provides the required replica to the user via the controller.

Writable unit: It receives a write request of the user through the controller and provides the required replica to user via the controller.

Local update unit: It propagates the update to the peer after the end of the operations in the local replica site.

Receive update: It receives updated data from the peers to the replica site containing a similar copy of the modified data.

c) Readable unit

When a replica is available in the site proximate to the user, the replica site receives the read request through the controller and sends the request to the readable unit. This configures the request for reading operation and sends the request to the Local Consistency Service (LCS) that maintains the information about the recent version of replica $\langle v_{id}, r_{id}, t_{id} \rangle$, where v_{id} denotes version identity (ID), r_{id} represents the replica ID and t_{id} indicates the transaction ID.

After the LCS provides a replica from the Storage Element (SE) and directly sends it to the user. But in case of unavailability of replica in replica site closest to the user, so the local consistency service fetches a copy from the peer and registers a new replica in the hash table list and maintains it for future purposes, but before that the LCS calculates the threshold value of storage element space. In algorithm 1 is described the process of reading replica from the replica site in our model, which leads to minimizing the execution time for reading process.

Read replica from a replica site

$R_d = \{n_i, n_j\}$;
Set of replica nodes holding recent version of the data item'd'
$H_R = \langle v_{id}, r_{id}, t_{id} \rangle$; hash table of replica node
SE: Storage element

Step 1: Function $Read\_replica\ \langle v_{id}, r_{id}, t_{id} \rangle$

Step 2:
$Get \leftarrow request\ from\ user\ of\ replica\ \langle v_{id}, r_{id}, t_{id} \rangle$

Step 3: If $(request\ replica\ in\ the\ same\ replica\ site)$

Step 4: $Get \leftarrow replica\ \langle v_{id}, r_{id}, t_{id} \rangle$

Step 5: Else {

Step 6: Fetch the file from peer $\langle v_{id}, r_{id}, t_{id} \rangle$

Step 7: Calculate the threshold value of SE in replica site;

Step 8: If $(the\ space\ of\ SE\ in\ replica\ site < threshold)$

Step 9: Store new replica $\langle v_{id}, r_{id}, t_{id} \rangle$

Step 10: Else {

Step 11: If $(access\ frequency\ of\ new\ replica > access\ frequency\ of\ old\ replica)$

Step 12: Delete old replica $\langle v_{id}, r_{id}, t_{id} \rangle$;}

Step 13: If $(enough\ in\ SE\ of\ ReplicaSite\ Server)$

Step 14: Go to Step 9;

Step 15: Else {return to Step 12;

Step 16: End If

Step 17: End If

Step 18: End If

Step 19: End If}}

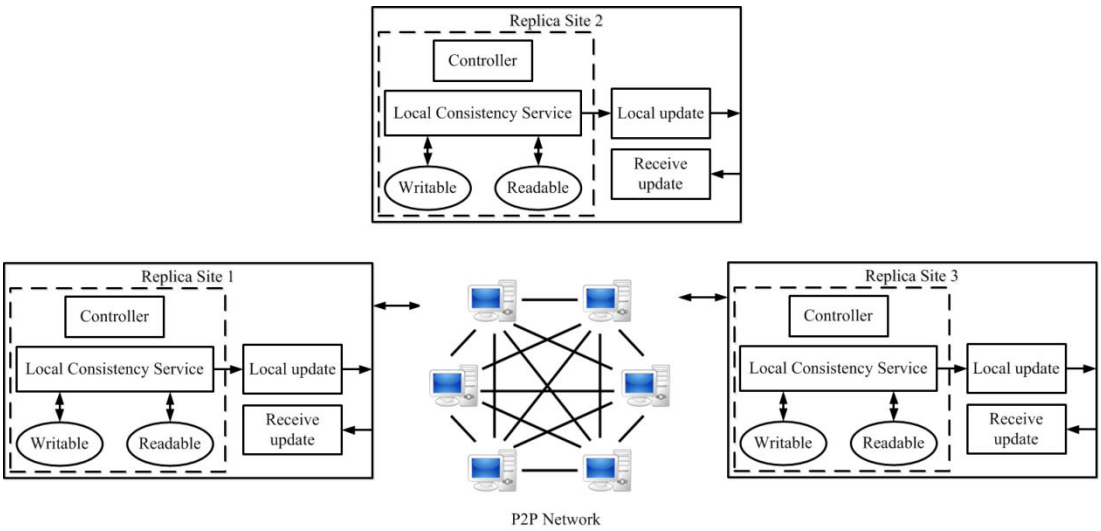

Figure-2. ERCM Architecture

d) Writable unit

Writing process is done after receiving the modification request of replica $\langle v_{id}, r_{id}, t_{id} \rangle$ available from a user. When the controller receives the write request, it forwards the write request to the writable unit that executes the writing operation based on the First in First out (FIFO) way. The writing operations are implemented in the asynchronous way to ensure the consistency of the replica in the cluster. If there are no other writing requests, the LCS locks the Logical Replica File (LRF) and other similar replicas from other users until the update process is finished. After the update process, the new replica $\langle v_{id+1}, r_{id+1}, t_{id+1} \rangle$ sends to the local update of the replica site. If there are more than one writing operations, it is implemented though the write set is ordered.

Write operation in local replica site

$R_d = \{n_i, n_j\}$;
Set of replica nodes holding recent version of the data item 'd'
$H_R = \langle v_{id}, r_{id}, t_{id} \rangle$; Hash table of replica node
SE: Storage element

Step 1: Function $Write\_to\_replica\ \langle v_{id}, r_{id}, t_{id} \rangle$

Step 2:
$Get \leftarrow request\ from\ user\ for\ replica\ \langle v_{id}, r_{id}, t_{id} \rangle$

Step 3: If $request = True\ and\ (waiting\ writable = \emptyset)$ Then

Step 4:
$LogicalReplicaFile = lock, replica\ \langle v_{id}, r_{id}, t_{id} \rangle$

Step 5: Increase the replica site's timestamp;

Step 6: $Get \leftarrow update\ replica\ \langle v_{id+1}, r_{id}, t_{id+1} \rangle$

Step 7:
$new\ update = \{insert\ replica\ \langle v_{id+1}, r_{id}, t_{id+1} \rangle\}$;

Step 8: Send update $\langle v_{id+1}, r_{id}, t_{id+1} \rangle$ to peer;

Step 9: Else {

Step 10: If set of writes received from other users during transaction Then

Step 11: Writeset is ordered;

Step 12: While
$(waiting\ writable = \emptyset)\ and\ not\ (waiting\ request)$
Do

Step 13: Return to Step 8

Step 14: End If;

Step 15: End If;

e) Update propagation operation

The consistency among replicas is required only in each peer in the network. When a read or write operation is issued by a peer, the operation is performed to a replica held by the peer. When the peer in the P2P network receives the replica $\langle v_{id+1}, r_{id}, t_{id+1} \rangle$ that is modified from any replica site in the data grid environment and if the replica exists in the Distributed Hash Table (DHT) list, the peer starts reading the metadata of this replica in the Replica Catalogue (RC). The RC contains the information about the location, size and version of the replica. Then, the Global Consistency Service (GCS) registers the new replica, saves the replica in the data storage and maintains the metadata for new replica in the RC. Then, the update is propagated to the replica sites having a similar replica in a synchronous way. The peer receives the acknowledgment from the replica sites.

Update propagation for peer

$R_d = \{n_i, n_j\}$;

Set of replica nodes holding recent version of the data item 'd'
$H_R = \langle v_{id}, r_{id}, t_{id} \rangle$; hash table of replica node

Step 1: $Receive \leftarrow update\ replica \langle v_{id+1}, r_{id}, t_{id+1} \rangle$

Step 2: If $arrived(request) = True$ Then

Step 3: Read the RC for replica $\langle v_{id}, r_{id}, t_{id} \rangle$

Step 4: Increase the server's timestamp;

Step 5:
$New\ update = \{insert\ replica\ \langle v_{id+1}, r_{id}, t_{id+1} \rangle\}$;

Step 6: For each RC record of replica $\langle v_{id+1}, r_{id}, t_{id+1} \rangle$ do

Step 7: Update propagation to all Replica Site (RS) for replica

Step 8: Receive acknowledgment;

Step 9: End If;

MDHRA-ERCM
//locate Unavailable Requested Files (URF)
Step 1: for each $(URF\ f_i\ in\ the\ local\ site)$\{
If $(f_i\ available\ in\ local\ LAN)$\{
Create list 'L' of f_j's that are available in local LAN
Select f_i from L with minimum value of F
Continue;
}
//end if
If $(f_i\ available\ in\ local\ region)$\{
Create list L of f_j's that are available in local region
Select f_i from L with minimum value of F
Continue;
}
//end if
Create list L of f_j's that are available in other regions
Select f_i from L with minimum value of F
Continue;
}
//end for each step 1
//Now all requested files are available in the local site

Step 2: Execute the job
//Now replicate each URF in best site

Step 3: for each $(R_i \in URF)$
Select best site for storing R_i
//In this step local LAN means the LAN that holds the best site
//Also local region means the region that holds the best site
$SR_i \leftarrow size\ of\ R_i$;
$SBSE \leftarrow storage\ size\ of\ best\ site$
//Do not replicate if size
of$R_i \geq storage\ size\ of\ best\ site$;
If $(SR_i \geq SBSE)$; continue;//not enough space
Access the file remotely
Switch (replica placement) {
If $(enough\ space\ exist\ for\ R_i\ in\ BSE)$
$\{Store\ R_i; continue;\}$
Exit;
New replica has a duplicate in other sites located within the same LAN
Exit;
Create List L1=f_j's that are both available in the best site and local LAN
Sort list L1 using LRU
While $(L1\ is\ not\ empty\ \&\&not\ enough\ space\ for\ R_i)$
For each file in L1
Delete first file from list L in best site;
If $(enough\ space\ exist\ for\ R_i\ in\ BSE)$ break;
Create List L2=Files that are available in current region randomly
Sort L2 using LRU
While $(L1\ is\ not\ empty\ \&\&not\ enough\ space\ for\ R_i)$
For each file in L2;
Delete file from list L2 in best site;
If $(enough\ space\ exist\ for\ R_i\ in\ BSE)$ break;
Create List L3=Remaining files in the BSE
Sort list L using RV in ascending order
While $(L\ is\ not\ empty\ \&\&not\ enough\ space\ for\ R_i)$
Delete first file from list L3 in best site;
Store R_i; }//end for each Step 3

Step 4: $Function\ Read\_replica\ R_i$
Fetch the file from peer
Compute access frequency of replica
If
$(access\ frequency\ of\ new\ replica >$
$access\ frequency\ of\ old\ replica)$
Delete Old replica
Else Store the replica

Step 5: $Write\_to\_replica$
$Get \leftarrow request\ from\ user\ for\ replica$
$Get \leftarrow update\ replica$
Send update to peer
If set of writes received from other users during transaction then
Order the write_set

Step 6: If update replica request is received then
Read the replica catalogue for replica
Increase the timestamp of peer
New update = insert replica
Update propagation to all Replica Site
Receive acknowledgment
End

Figure-3 shows the flow diagram of the proposed work. The main advantages of the ERCM are high replica consistency and minimum job execution time by updating propagation. Job execution time is reduced through the optimal allocation of replicas to minimize the retrieval

time. ERCM uses the local hash table list for finding the needed files. If necessary files are not available on the local site, the proposed strategy must transfer the necessary files from the peer to the local site and replicate it for the future execution. High replica consistency is achieved through the optimal allocation of replica consistency and updating propagation in the writing process. ERCM contains an asynchronous replica consistency inter-replica site to guarantee replica consistency irrespective of the frequent changes by the users. Then, the updates are distributed simultaneously, where the peer performs the propagation operations to all sites having a similar replica that is modified recently. The ERCM has a reasonable execution time for both the reading and writing operations with a high availability.

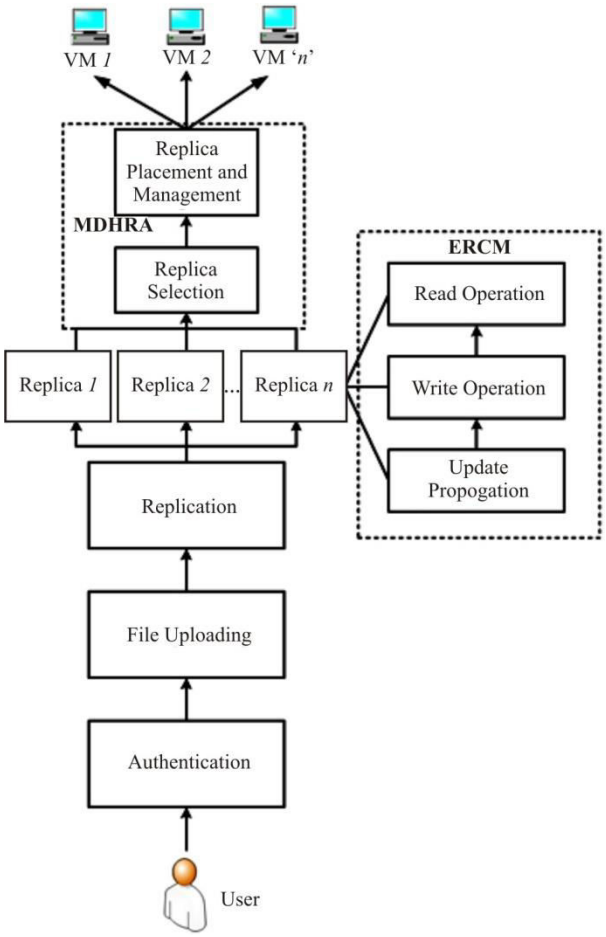

Figure-3. Flow diagram of proposed work

4. PERFORMANCE ANALYSIS

The proposed MDHRA-ERCM work is simulated using the CloudSim tool. CloudSim is a framework used for the simulation of the cloud computing infrastructures and services. It supports the behaviour modeling of the cloud components such as computing devices, data centers, VMs, servers and resource provisioning policies. The test bed environment of the proposed research work is presented in Table-1.

A. Performance metrics

The proposed work is evaluated with the following parameters such as ENU, storage usage, replication frequency and hit ratio.

a) ENU

ENU indicates the ratio of transferred files to the requested files. Hence, a small ENU value determines that the data replication technique is highly successful in storing the data files in the appropriate locations.

$$E_{enu} = \frac{N_{rfa}+N_{fa}}{N_{lfa}} \quad (7)$$

Table-1. Experimental setup

Topology parameters	
Number of data centres	2
Number of VMs (Minimum)	25
Storage space at each VM	100 MB
Job parameters	
Number of files	50
Size of single file	100 MB
Total size of files	User defined
Queue size	5
Delay Time	1000 ms

Where N_{rfa} denotes the number of remote file accesses, N_{fa} indicates the number of file replications, and N_{lfa} represents the number of local file access. The range of ENU lies from zero to one. Lower ENU value demonstrates that the network bandwidth is utilized more efficiently. Thus, it is clear that the data replication is a time and cost consuming process.

b) Storage usage

Storage usage in the replication scheme is defined as

$$\text{Storage usage} = \frac{\text{Filled\_space\_available}}{\text{Space}} \quad (8)$$

The main aim is to save the storage space and increase the storage utilization rate, as the resource cost is linearly proportional to the amount of storage space being utilized.

c) Replication frequency

Replication frequency indicates the number of file replications generated per data access. The replication frequency is high, when number of replicas. It is obvious that each replication process consumes bandwidth and storage resource. Thus, effective replication strategy

controls the replication frequency to avoid the server load and heavy network load.

d) Hit ratio

Hit ratio is defined as the amount of total number of local file accesses to all accesses.

B. Comparative analysis

The MDHRA-ERCM approach is compared with the Prefetching-aware Data Replication (PDR) [25], EIMORM [12], DARS [13], Dynamic replica creation strategy based on file heat and node load in hybrid cloud (DRCSBHL) [14], Dynamic Popularity aware Replication Strategy (DPRS) [15], Fuzzy-FP [16] and Performance and Profit Oriented Data Replication Strategy (PEPR) [17]. Figure-4 shows the ENU analysis of the MDHRA-ERCM approach and existing replication techniques. The ENU of the DARS strategy is minimized to about 10% and 15% when compared with the Fuzzy-FP and DRCSBHL, respectively, as the DARS strategy stores the file replica with the maximum node degree. Hence, the access path is reduced and probability of accessing the replica is improved. ENU of PEPR is reduced to about 38% when compared to the DRCSBHL algorithm. The PEPR creates new replicas when the response time and economic benefits for the cloud service provider are fulfilled. The PDR algorithm yields better performance as the necessary files are available in the local data center. The proposed approach requires minimum ENU than the existing techniques, as the required files are present at the grid sites during the time of need.

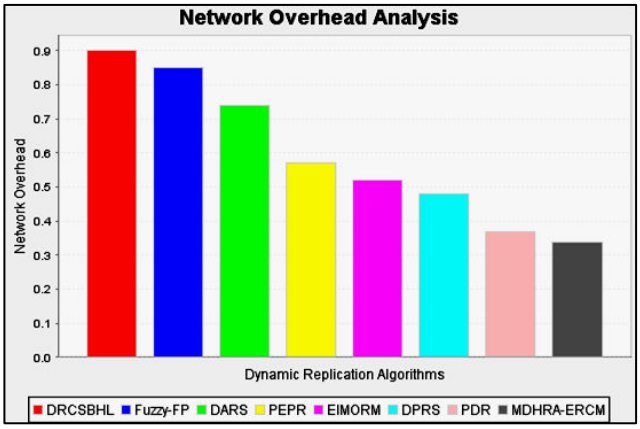

Figure-4. ENU analysis of the MDHRA-ERCM and existing techniques

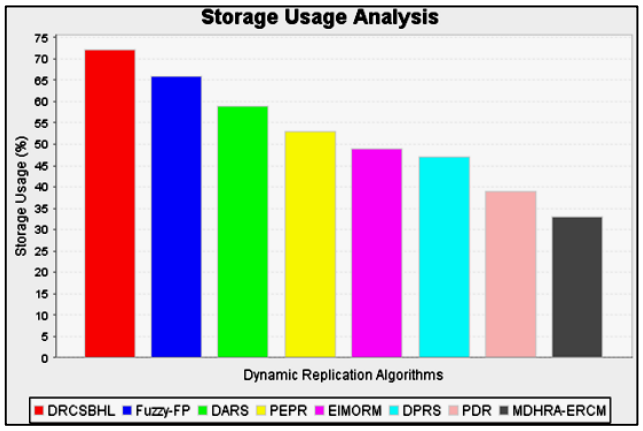

Figure-5. Storage usage analysis of the MDHRA-ERCM and existing techniques

Figure-5 shows the storage usage analysis of the MDHRA-ERCM and existing techniques. In the storage usage, the EIMORM and DPRS schemes show the affordable storage consumption while the DRCSBHL strategy fails in the storage consumption aspect. Storage usage of the DPRS decreases around 19% on average than the PEPR. The storage consumption of the file replication is reduced, as it replicates the most popular files. PDR algorithm requires minimum storage usage when compared with the existing replication algorithms, while ensuring high file availability in the cloud. The PDR strategy only maintains the critical files based on the value assigned to each file. Fuzzy rules choose the replicas with minimum value to replace when the storage space of a replica site is completely filled. The MDHRA-ERCM approach selects the location of the best replica based on the response time and ensures optimal allocation of replica to reduce the job execution time. Hence, the storage usage of the proposed approach is about 50% minimum than the DRCSBHL.

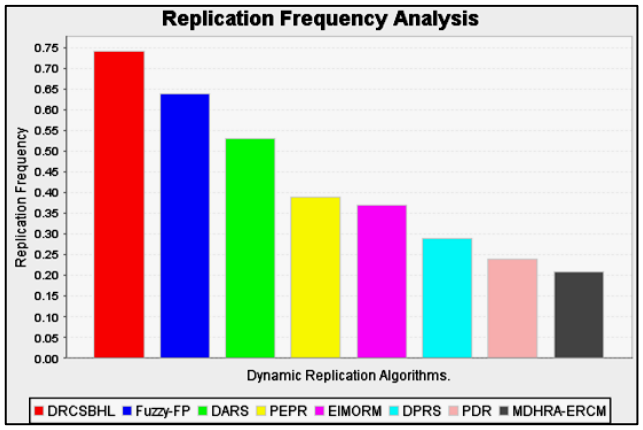

Figure-6. Replication frequency analysis of the MDHRA-ERCM and existing techniques

Figure-6 illustrates the replication frequency for different replication strategies. The replication frequency of the proposed MDHRA-ERCM approach is 0.21. The MDHRA algorithm keeps only the valuable replicas while

other insignificant replicas are substituted with the highly significant replicas. The replication frequency of the MDHRA-ERCM approach is lower than the existing techniques. Figure-7 depicts the comparative analysis of the hit ratio for the MDHRA-ERCM and existing techniques. As the access latency is improved by selecting the best replica, the MDHRA-ERCM approach yields maximum hit ratio than the existing techniques.

Table-2 shows the network overhead, replication frequency and storage usage analysis of the dynamic replication algorithms. Figure-8 shows the comparative analysis of the overall number of communications for different replication strategies. The MDHRA-ERCM approach requires minimum number of communications when compared to the existing dynamic replication strategies.

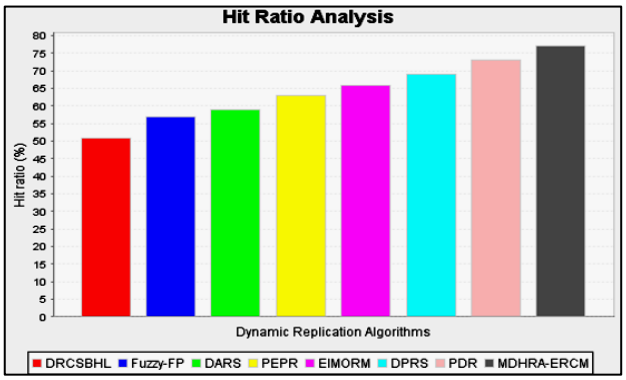

Figure-7. Hit ratio of the MDHRA-ERCM and existing techniques

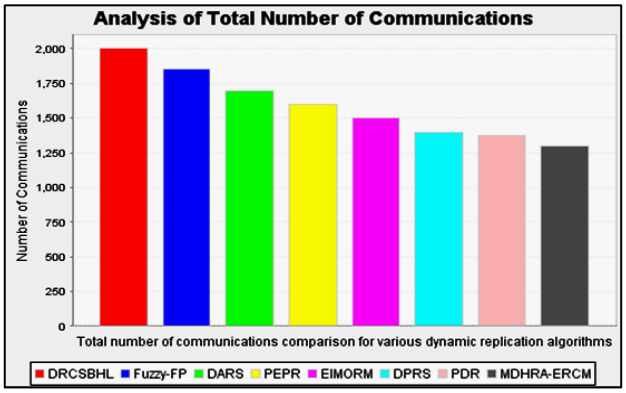

Figure-8. Number of communications for different replication strategies

Table-2. Network overhead, replication frequency and storage usage analysis of dynamic replication algorithms

Dynamic replication algorithm	Network overhead	Replication frequency	Storage usage (%)
DRCSBHL	0.90	0.74	72
Fuzzy-FP	0.85	0.64	66
DARS	0.74	0.53	59
PEPR	0.57	0.39	53
EIMORM	0.52	0.37	49
DPRS	0.48	0.29	47
PDR	0.37	0.24	39
MDHRA-ERCM	0.34	0.21	33

5. CONCLUSIONS

To address the file access latency issues, a dynamic replication strategy is highly necessary in the data grid environment. The MDHRA algorithm replaces the file replicas based on the factors such as last time the replica was requested by the user; file access rate and replica size. Therefore, the grid sites will have their required files locally at the time of need. This will reduce the response time, access latency, bandwidth consumption and increase system performance significantly. By selecting the best replica among numerous replicas based on the data transfer time, storage access latency, replica requests in the storage queue and the distance between nodes, the access latency of the replica is minimized. The MDHRA algorithm is optimized for the significant reduction of the bandwidth consumption and network traffic. MDHRA algorithm yields the best performance as it will not delete the file with high transfer time. The main objectives of the ERCM model are to reduce the response time of reading and writing operations and maintain the replica consistency in the data grid.

REFERENCES

[1] J. Balasangameshwara and N. Raju. 2012. A hybrid policy for fault tolerant load balancing in grid computing environments. Journal of Network and computer Applications. 35: 412-422.

[2] A. Chervenak, I. Foster, C. Kesselman, C. Salisbury and S. Tuecke. 2000. The data grid: Towards architecture for the distributed management and analysis of large scientific datasets. Journal of network and computer applications. 23: 187-200.

[3] C. Li and L. Li. 2009. Three-layer control policy for grid resource management. Journal of Network and Computer Applications. 32: 525-537.

[4] A. Folling, C. Grimme, J. Lepping, and A. Papaspyrou. 2010. Robust load delegation in service grid environments. IEEE Transactions on Parallel and Distributed Systems. 21: 1304-1316.

[5] O. Sonmez, H. Mohamed and D. Epema. 2010. On the benefit of processor coallocation in multicluster grid systems. IEEE Transactions on Parallel and Distributed Systems. 21: 778-789.

[6] H. Li. 2010. Realistic workload modeling and its performance impacts in large-scale escience grids. IEEE Transactions on Parallel and Distributed Systems. 21: 480-493.

[7] A. Navaz, C. Prabhadevi and V. Sangeetha. 2013. Data grid concepts for data security in distributed computing. arXiv preprint arXiv:1308.6058.

[8] A. S. Tanenbaum. 1995. Distributed operating systems: Pearson Education India.

[9] X. Chen, S. Ren, and H. Wang. 2005. SCOPE: Scalable consistency maintenance in structured P2P systems. in INFOCOM 2005. 24th Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings IEEE. pp. 1502-1513.

[10] N. N. Khasmakhi, S. Jamali, and M. A. Chenaghlu. 2016. A Solution for Replica Consistency Maintenance in Unstructured Peer-To-Peer Networks.

[11] T. Hara and S. K. Madria. 2005. Consistency management among replicas in peer-to-peer mobile ad hoc networks. in Reliable Distributed Systems, 2005. SRDS 2005. 24th IEEE Symposium on. pp. 3-12.

[12] E. B. Edwin, P. Umamaheswari and M. R. Thanka. 2017. An efficient and improved multi-objective optimized replication management with dynamic and cost aware strategies in cloud computing data center. Cluster Computing. pp. 1-10.

[13] S. Sun, W. Yao and X. Li. 2018. DARS: A dynamic adaptive replica strategy under high load Cloud-P2P. Future Generation Computer Systems. 78: 31-40.

[14] Y. Zhao, C. Li, L. Li, and P. Zhang. 2017. Dynamic replica creation strategy based on file heat and node load in hybrid cloud. in Advanced Communication Technology (ICACT), 2017 19th International Conference on. pp. 213-220.

[15] N. Mansouri, M. K. Rafsanjani, and M. Javidi. 2017. DPRS: A dynamic popularity aware replication strategy with parallel download scheme in cloud environments. Simulation Modelling Practice and Theory. 77: 177-196.

[16] P.Elango and D. Kuppusamy. 2016. Fuzzy FP-Tree based Data Replication Management System in Cloud. International Journal of Engineering Trends and Technology (IJETT). 36: 481-489.

[17] U. Tos, R. Mokadem, A. Hameurlain, T. Ayav, and S. Bora. 2016. A performance and profit oriented data replication strategy for cloud systems. in Ubiquitous Intelligence & Computing, Advanced and Trusted Computing, Scalable Computing and Communications, Cloud and Big Data Computing, Internet of People, and Smart World Congress (UIC/ATC/ScalCom/CBDCom/IoP/SmartWorld), 2016 Intl IEEE Conferences. pp. 780-787.

[18] Q. Liu, G. Wang and J. Wu. 2014. Consistency as a service: Auditing cloud consistency. IEEE Transactions on Network and Service Management. 11: 25-35.

[19] L. Field and R. Sakellariou. 2017. An Evaluation of Information Consistency in Grid Information Systems. Journal of Grid Computing. 15: 127-137.

[20] H. E. Ahangaran and A. M. Rahmani. 2010. An innovative approach of data grid consistency using tree-based clustering. in Computer and Network Technology (ICCNT), 2010 Second International Conference on. pp. 258-261.

[21] N. Mansouri and G. H. Dastghaibyfard. 2012. A dynamic replica management strategy in data grid. Journal of network and computer applications. 35: 1297-1303.

[22] N. Mansouri, G. H. Dastghaibyfard and E. Mansouri. 2013. Combination of data replication and scheduling algorithm for improving data availability in Data Grids. Journal of Network and Computer Applications. 36: 711-722.

[23] S. K. Karna and R. Sahai. 2012. An overview on Taguchi method. International Journal of Engineering and Mathematical Sciences. 1: 1-7.

[24] A. H. Guroob and D. Manjaiah. 2016. Efficient Replica Consistency Model (ERCM) for update propagation in Data Grid Environment. in Information

Communication and Embedded Systems (ICICES), 2016 International Conference on. pp. 1-7.

[25] N. Mansouri and M. Javidi. 2018. A New Prefetching-aware Data Replication to Decrease Access Latency in Cloud Environment. Journal of Systems and Software.