A Novel Replication Strategy in Data Grid Environment with a Dynamic Threshold

Sheida Dayyani[1], Mohammad Reza Khayyambashi [2]
[1] Department of Computer Engineering, Sheikh Bahaei University, Isfahan, Iran
[2] Department of Computer Engineering, University of Isfahan, Isfahan, Iran

Abstract. Data Grid is a type of Grid Computing systems whichis designed to provide geographically distributed data resources to large computational problems that require mining and evaluating large amounts of data. Managing this data in a centralized location increases the data access time and hence much time is taken to execute the job. So to reduce the data access time, "Replication" is used. Data replication is known as an important optimization technique that aims to improve data access time and toutilize network and storage resources efficiently.Since the data files are very large and the Grid storagesare limited, managing replicas in storage for the purpose of more effective utilization requiresmore attention.In this paper, a novel data replication strategy, called Dynamic Hierarchical Replicationwith Threshold (DHRT) is proposed. This strategy is an enhanced version of the Dynamic Hierarchical Replication (DHR)strategy that uses a new threshold for characterizing the number of appropriate sites for replication. Appropriate sites have the higher number of access for that particular replica from other sites. It also minimizes access latency by selectingthe best replica when various sites hold replicas. The proposed replica selection strategy selects thebest replica location for the users' running jobs by considering the replica requests that are waite in thestorage and number of stored file. The simulated results with OptorSim, i.e. European Data Grid simulatorshow that the DHRT strategy gives better performance compared to the other algorithms and preventsthe unnecessary creation of replicas which leads to efficient storage usage.

Keywords: Grid Computing, Data Grid,Data replication, Dynamic Threshold, OptorSim.

1. Introduction

Grid computing is a wide area distributed computing environment that enables sharing, selection, and aggregation of geographically distributed resources. Also, it is an important mechanism for utilizing distributed computing resources. These resources are distributed in different geographical locations, but are organized to provide an integrated service. The term "grid computing" refers to the emerging computational and networking infrastructure that is designed to provide pervasive and reliable access to data and computational resources over wide area network, across organizational domains [1, 2, 3].

Nowadays, there is a tendency of storing, retrieving, and managing different types of data such as experimental data that is produced from many projects [1]. These data play a fundamental role in all kinds of several scientific applications such as particle physics, high energy physics, data mining, climate modelling, earthquake engineering and astronomy [2, 3, 4]. Storing such amount of data in the same location is difficult, even impossible. Moreover, an application may need data produced by another geographically remote application. For this reason, a grid is large scale resource sharing and problem solving mechanism in virtual organizations and is suitable for the above situation [5, 6, 7].

One class of grid computing is "Data Grids"; that provide geographically distributed storage resources to large computational problems that require evaluating and mining large amounts of data [8, 9]. The Grid resources, including computing facility, data storage and network bandwidth, are consumed by the jobs. For each incoming job, the grid scheduler decides where to run the job based on the job requirements and the system status. In data-intensive applications, the locations of data required by the job impact the Grid scheduling decision and performance greatly. Creating data replicas can reroute the data requests to certain replica servers and offer remarkably higher access speed than a single server. At the same time, the replicas provide broader decision space for the grid scheduler to achieve better performance from the perspective of the job [2, 3, 9]. Managing this data in a centralized location increases the data access time and hence much time is taken to execute the job. So to reduce the data access time, "Replication" is used [2, 3].

The Data Replication is the process of creation and placement of the copies of entities of data. The phase of creation consists in reproducing the structure and the state of the replicated data, whereas the phase of placement consists in choosing the suitable site of this new replica, according to the objectives of the replication [9, 10].

The experiments on the distributed systems show that the replication promotes higher data availability, lower bandwidth consumption, increase in fault tolerance and improvement in scalability [11]. By storing the data at more than one site, if a data site fails, a system can operate using replicated data, thus increasing availability and fault tolerance. At the same time, as the data is stored at multiple sites, the request can find the data close to the site where the request originated, thus increasing the performance of the system. But the

benefits of replication, of course, do not come without overheads of creating, maintaining and updating the replicas. If the application has read-only nature, replication can greatly improve the performance. But, if the application needs to process update requests, the benefits of replication can be neutralised to some extent by the overhead of maintaining consistency among multiple replicas [11].

Various combinations of events and access scenarios of data are possible in a distributed replicated environment. The three fundamental questions any replica placement strategy has to answer are as follow and depending on the answers, different replication strategies are born [3, 12]:

- When should the replicas be created?
- Which files should be replicated?
- Where should the replicas be placed?

In this work a novel data replication strategy, called Dynamic Hierarchical Replicationwith Threshold (DHRT) algorithm is proposed. DHRT extends proposed algorithm in [13] and stores replicas in appropriate sites inthe requested region that have the higher number of access for that particular replica from other sites.

The paper is organized as follows. Section 2 gives an overview of some relatedwork on data replication strategies. In Section 3 a DHRT algorithm is proposed.Section 4 shows the simulations results. Finally, conclusions and future researchworks are presented in Section 5.

2. Related works

The role of a replication strategy is to identify when a replica should be created, where to place replicas, when to remove replicas and how to locate the best replica. These strategies are guided by factors such as demand for data, network conditions, cost of transfer and storage cost. Replication has been an interesting topic in World Wide Web, peer-to-peer networks, ad hoc and sensor networking, mesh networks and distributed databases [14-17].

Foster and Ranganathan [12], have proposed six distinct replica strategiesfor multi-tier data. These strategies are described briefly as follow:

- NO Replication: This strategy will not create replica and therefore, the files are always accessed remotely. One example of the implemented strategy is the SimpleOptimizer algorithm [26], which never performs replication; rather it reads the required replica remotely. SimpleOptimizer algorithm is simple to implement and performs the best in relative to other algorithms in terms of the storage space usage, but performs the worst in terms of job turnaround time and network usage [12].
- Best client: Thiscreates replica at the client that has generated the most request for a file, this client is called best client. At a given time interval, each nodes checks to see the number of requests for any of its file has exceeded a threshold. If so, the best client for that file is identified [12].
- Cascading Replication:This strategy supports tree architecture, since the data files generated in the top level and once the number of accesses for the file exceeds the threshold, then a replica is created at the next level, but on the path to the best client, and so on for all levels, until it reaches to the best client itself [12].
- Plain Cashing:In this strategy, the client that requests a file stores a copy locally. If these files are large and a client has enough space to store only one file at a time, then files get replaced quickly [12].
- Caching plus Cascading:This Strategy combines cascading and plain cashing strategies. The client caches file locally, and the server periodically identifies the popular files and propagates them down the hierarchy. Note that the clients are always located at the leaves of the tree but any node in the hierarchy can be a server. Specifically, a Client can act as a Server to its siblings. Siblings are nodes that have the same parent [12].
- Fast Spread:In this method a replica of the file is stored at each node along its path to the client. When a client requests a file, a copy is stored at each tier on the way. This leads to a faster spread of data. When a node does not have enough space for a new replica it deletes the least popular file that had come in the earliest [12].

In [18] the authors have presented a novel dynamic replication strategy called Bandwidth Hierarchy Replication (BHR)which reduces data access time by avoiding network congestions in a data grid network. This strategy takes benefits from ‘network-level locality’ which represents that required file is located in the site which has broad bandwidth to the site of job execution. The BHR strategy isevaluated by Optorsim simulator. The simulation results show that BHR strategy can outperform other optimization techniques in terms of data access time when hierarchy of bandwidth appears in Internet. BHR extends current site-level replica optimization study to the network-level.

Tang et al. [19] proposed two dynamic replica strategies, SimpleBottom Up and Aggregate Bottom Up, for the multi-tier architecture for Data Grids. The SBU algorithm replicates the data file that exceeds a pre-defined

threshold for clients. The main shortcoming of SBU is the lack of consideration to the relationship with historical access records. For the sake of addressing the problem, ABU is designed to aggregate the historical records to the upper tier until it reaches the root. The performance of algorithms was evaluated and improvements shown against Fast Spread dynamic replication strategy. The values for interval checking and threshold were based on data access arrival rate, data access distribution and capacity of the replica servers.

In [20], Chang et al. presented a dynamic data replication mechanism called Latest Access Largest Weight (LALW) that selects a popular file for replication and calculates a suitable number of copies and grid sites for replication. By associating a different weight to each historical data access record, the importance of each record is differentiated. A more recent data access record has a larger weight. It indicates that the record is more pertinent to the current situation of data access.

A replication algorithm for a 3-level hierarchy structure and a scheduling algorithm are proposed in [21]. They considered a hierarchical network structure that has three levels. In their replication methodamong the candidate replicas they select the one that has the highest bandwidth to the requested file. Similarly,it uses the same technique for file deletion. This leads to a better performance comparing with LRU (Least Recently Used) method. For efficient scheduling, their algorithm selects the best region, LAN and siterespectively. Best region (LAN, site) is a region (LAN, site) with most of the requested files.

In [22], Shorfuzzaman et al. proposed an Adaptive Popularity Based Replica Placement (APBRP)algorithm. APBRP is a new dynamic replica placement algorithm for hierarchical data grids which is guided by file “popularity”. The goal of this strategy is to place replicas close to clients to reduce data access time while still using network and storage resources efficiently. The effectiveness of APBRP depends on the selection of a threshold value related to file popularity. APBRP determines this threshold dynamically based on data request arrival rates.

In [13] the authors presented aDynamic Hierarchical Replication (DHR)algorithm that places replicas in appropriate sites, i.e. best site that has the highest number of access for that particular replica. The algorithm minimizes access latency by selecting the best replica when various sites hold replicas. The proposed replica selection strategy selects the best replica location for the users running jobs by considering the replica requests that waiting in the queue and data transfer time. It stores the replica in the best site where the file has been accessed most, instead of storing files in many sites [13].

Modified Latest Access Largest Weight (MLALW) is an enhanced version of Latest Access Largest Weight strategy that proposed in [23]. MLALW deletes files by considering three important factors such as least frequently used replicas, least recently used replicas and the size of the replica. Also, it stores each replica in an appropriate site, i.e. appropriate site in the region that has the highest number of access in future for that particular replica.

In a data grid reducing job’s waiting time in queue and execution time dependson where to assign job for execution and where to get the required data files.Therefore, scheduling jobs at proper sites and getting replicas from proper sites areimportant factors from user’s point of view. We leave the job scheduling problem[24], which studies how to dispatch jobs into Grid sites for execution, and it’scoupling with data replication, as our future research. Some researchers havestudied the relationship between data replication and job scheduling [24-27].

3. Proposed replication strategy

In this section, first network structure of data grid is described, and then a novel DHRT algorithm is proposed.

3.1. Grid Network Structure

The grid topology of the simulated platform is given in Fig. 1,which isbased on European DataGrid CMStestbed architecture [28] and has three levels similar to what is given by Horri et al.[21].

1) First level are Regions that are connected through internet i.e. have low bandwidth.
2) Second level comprises of LAN’s (local area network) within each region that have moderately higher bandwidth compared to the first level.
3) The third level are the sites within each of the LAN’s, that are connected to each other with a high bandwidth.

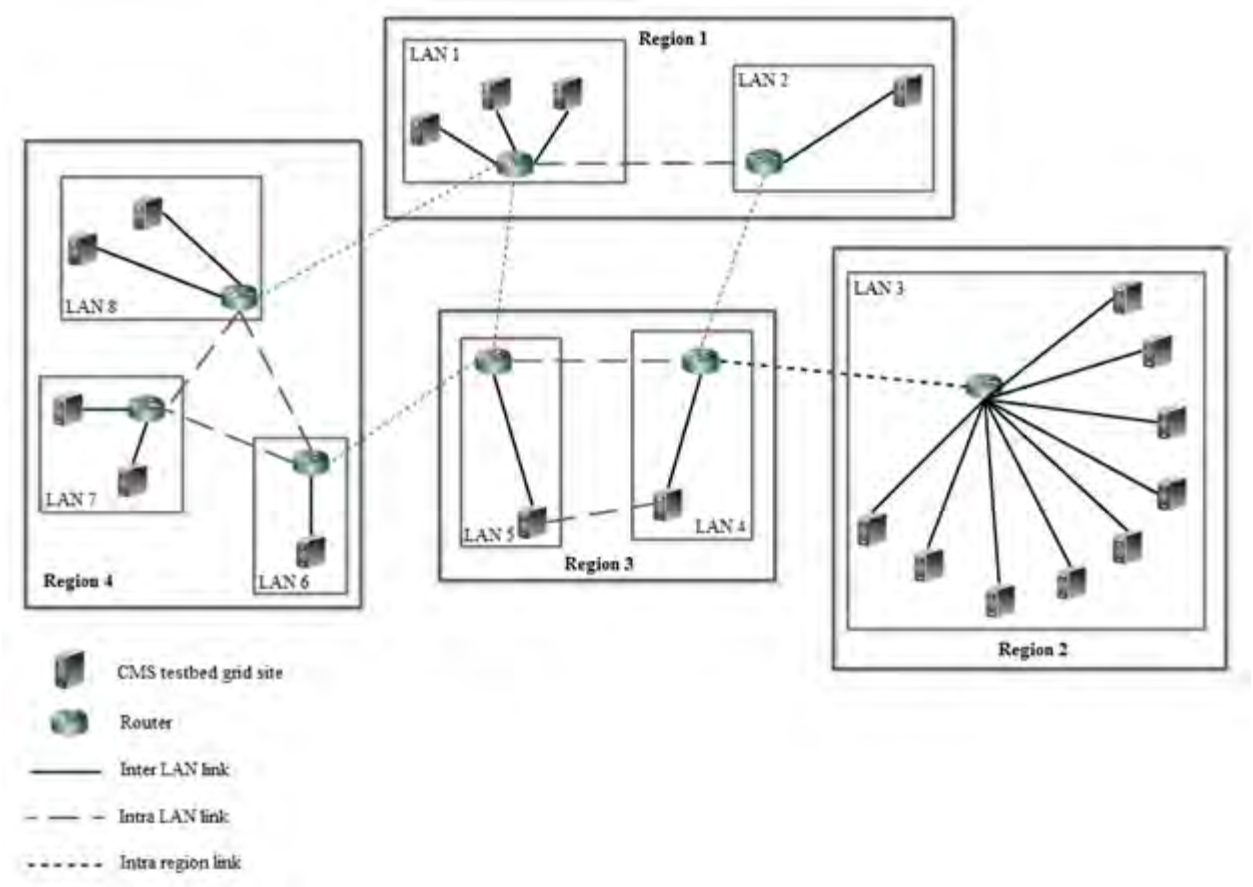

Fig. 1. Grid topology in the simulation

3.2. Dynamic Hierarchical Replication with Threshold

When a job is allocated to the gridscheduler, before job execution the replica manager should transfer all the required files that are not available in the local site. So, the data replication enhances the job schedulingperformance by decreasing job execution time. DHRTis an enhanced version of Dynamic Hierarchical Replicationstrategy[13] andhas three parts:

3.2.1. Replica selection

Generally when several replicas are available within the local LAN, the local region or other regions, DHRT selects the site that has the least number of stored files, since the bandwidth in each level of the network is fixed.Fig. 2 describes DHRT's selection algorithm.

```
//locate unavailable requested files (URF)
//replica selection
1. Foreach (URF f_i in the local site) {
   If (f_i available in local LAN){
      Create list L of f_i`s that are available in local LAN.
      Select f_i from L that has least number of requests.
      Continue; } //end if

   If (f_iavailable in local region) {
      Create list L of f_i`s that are available in local region.
      Select f_i from L that has least number of requests.
      Continue; } //end if

   Create list L of f_i`s that are available in other regions.
   Select f_i from L that has least number of requests.
   } //end foreach step 1
```

Fig. 2. Selection algorithm of DHRT strategy

3.3.2. Replica decision

When a requested replica is not available in the local storage, replication should take place. According to the temporal and geographical locality the replica is placed in the best sites (BSEs). To select the BSEs, DHRT characterizes the number of appropriate sites for replication by a dynamic threshold. Decision algorithm calculates this threshold from Eq. (1).

Then, DHRT creates a sorted list (by number of replica access) of all SEs that requested the particular file in the region. Now the replica will be placed in the storage elements of the above sorted list that threshold shows.

$$Dynamic\ Threshold = \frac{NA(f) \times NS(f)}{NR(f) \times S(f)} \quad (1)$$

The dynamic threshold allows to candidate more than one SE and increase number of replicas. On th hand, replica is not placed in all the requested sites. Hence, DHRT helps to find an appropriate nur replicas based on fourfactors, such as:

- NA(f): Number of access to that particular file
- NS(f): Number of sites that will require that particular file
- NR(f): Number of existing replicas of that particular file
- S(f): Size of that particular file

Fig. 3 describes DHRT's decision algorithm.

```
//Now all requested files are available in the local site
//replica placement
2. Execute the job
//Now replicate each unavailable requested files (URF)
//in the best SEs (BSE) that dynamic threshold shows
3. Foreach ( Ri ϵ URF) {
   Run dynamic threshold function to choose best
        number of replicas (BNR) for storing Ri
   While (BNR>0){
      Select a BSE for storing Ri.
      //in this step local LAN means the LAN that holds BSE
      //also local region means the region that holds BSE
      SRi ← size of Ri;
      SBSE ← storage size of best SE;
      //don't replicate if size of Ri ≥ storage size of BSE;
      If (SRi≥SBSE)  Continue;//not enough space
      If (enough space exist for Ri in BSE) {
          Store Ri;
          Continue; }
      //don't replicate if Ri is available in the local LAN
      If (Ri is available in the local LAN) Continue;
   }//end of while
```

Fig. 3. Decision algorithm of DHRT strategy

3.3.3. Replica replacement

The proposed strategy performs replacement process like DHR algorithm.If enough storage space exists in the local site, the selected file will be replicated. Otherwise if the file is available in the local LAN, then it will be accessed remotely. Now, if enough space for replication does not exist and requested file is not available in the same LAN, one or more files should be deleted using the following rules:

- Generate a LRU (least recently used) sorted list of replicas thatare both available in the current site as well as the local LAN.
- Start deleting files from the above list till space is available for replica.
- If space is still insufficient, then repeat previous step for each LAN in current region, randomly. In other word, generate a LRU sorted list of replicas that are both available in the site as well as the local region.
- If space is still insufficient, generate a LRU sorted list of the remaining files in the site and start deleting files from the above list till space is available for replica.

Fig. 4 describes DHRT's replacementalgorithm.

```
//Now delete those files from BSE that are also
available in the local LAN
Create list L of fi`s that are both available in the BSE
        as well as in the local LAN.
Sort list L using LRU.
While (L is not empty && not enough space for Ri)
   Delete first file from list L in BSE;
If (enough space exist for Ri in BSE){
   Store Ri;
   Continue; }

//Now delete those files from BSE that are also available
//in the local region
Create list L of fi`s that are both available in the BSE
        as well as in the local region.
Sort list L using LRU.
While (L is not empty && not enough space for Ri)
   Delete first file from list L in BSE;
If (enough space exist for Ri in BSE) {
   Store Ri;
   Continue; }
//Now delete from remaining files in BSE
Create list L from remaining files in BSE.
Sort list L using LRU.
While (L is not empty && not enough space for Ri)
    Delete first file from list L in BSE;
Store Ri;
}//end foreach step 3
```

Fig. 4. Replacement algorithm of DHRT strategy

4. Experimental Evaluation

4.1. Simulation Tool

OptorSim is used to evaluate the performance of DHRT algorithm.OptorSim [29] was developed by European Data Grid projects andis written in Java. It provides a framework to simulate the realgrid environment. It is developed to test the dynamic replicationstrategies.In data grid environment various job execution scenarios are present.

OptorSim has severalimportant components such as computing element (CE), storage element (SE),resource broker (RB), replica manager (RM), and replica optimizer (RO). Computing elements and storage elements are used to execute grid jobs and store thefiles respectively. Additional details about OptorSim are available in the literatures[30, 31].

4.2. Evaluation Parameters

The architecture used here is the European DataGrid CMStestbed architecture [28]. In this there are twenty sites inwhich two of them have only storage element and which acts asmaster node. There are 8 routers which isused toforward request to other sites.

With OptorSim, it is possible to simulate any grid topology and replicationstrategy. So OptorSim code has been modified to implement the hierarchicalstructure, since it uses a flat network structure. It is assumed the network has fourregions and on average two LAN's in every region. The average storage capacity is 24.25 GB.Bandwidth in each level is given in Table 1. Also, Table 2specifies the simulation parameters and their values used in our study. Data replication strategies commonlyassume that the data is read only in Data Grid Environments.

Table 1. Bandwidth configuration

Parameter	Value (Mbps)
Inter LAN bandwidth	1000
Intra LAN bandwidth	100
Intra Region bandwidth	10

Table 2. General configuration parameters

Parameter	Value
Number of jobs types	6
Job Delay (ms)	2500
Maximum queue size	200
Number of jobs	100
Average size of storage elements (GB)	54.25
Size of each file (GB)	10

4.3. Results and Discussion

The proposed DHRT algorithm is compared with four replication algorithmsnamely, No Replication, Least Frequently Used (LFU), Least Recently Used(LRU), and DHR. In No Replication strategy files are accessed remotely. Whenstorage is full, LRU deletes least recently accessed files and LFU deletes leastfrequency accessed files. The DHR algorithm places replicas in appropriate sites i.e. best site thathas the highest number of access for that particular replica. It also minimizes access latency by selecting thebest replica by considering the replica requests that waiting in the storage and data transfer me.

Figure 5 shows the mean job execution time for the various replication algorithms.Obviously, the No Replication strategy has the worst performance as all the filesrequested by jobs have to be transferred from main site. In this simulation LRU andLFU have almost the same execution time. DHR improves data access time byconsidering the differences between intra-LAN and inter-LAN communication. DHRT mean job execution time isfaster than DHR since it considers a dynamic threshold to distinguish the best number of replicas. If the available storage for replication is not enough, DHRT will notdelete those file that have a high transferring time. It alsoimproves the mean job execution time by storing the replica in the most frequentlyaccessed site of the requested region.Valuable information can be gained by monitoring storage resources usage.Since resource cost is proportional to the resource used, so minimizing storageusage is a must.

Figure 6 shows the storage usage which is the percentage ofavailable spaces that are used. No Replication strategy has best storage since it getsfiles remotely. LFU and LRU are always replicate when a request is made;hence they use a lot of space. DHR strategy performs better than the previous threestrategies since it keeps at most one copy of file in the region. The proposedDHRT strategy has minimum storage usage among the current algorithmsbecause it place replicas in the appropriate sites so reduces unnecessary replication.

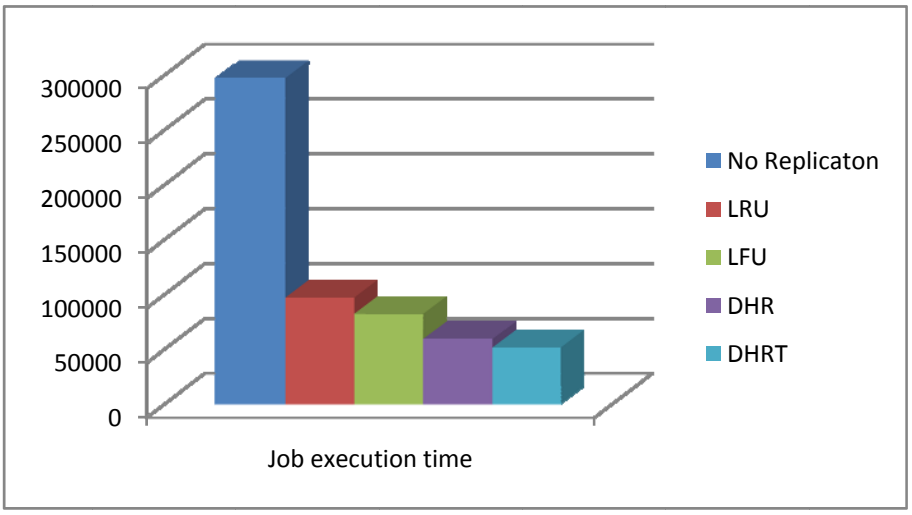

Fig. 5. Mean job execution time for various replicationalgorithms

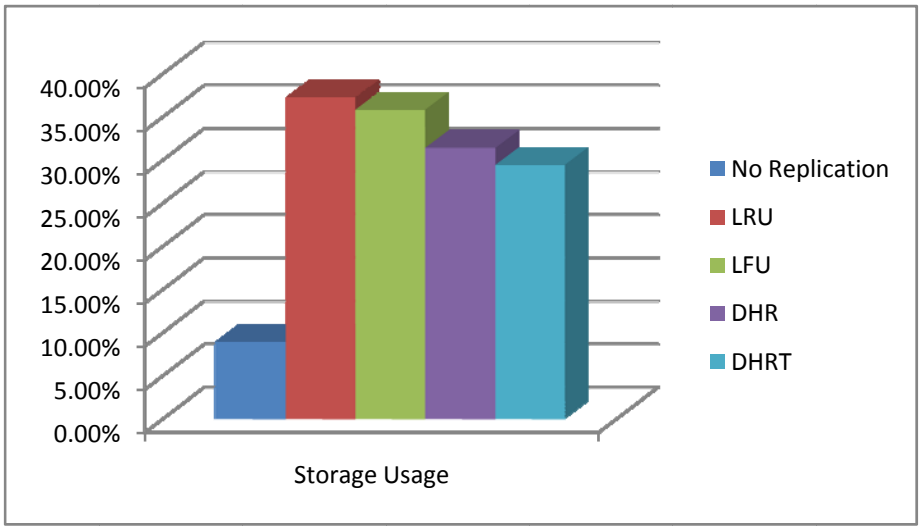

Fig. 6. Mean storage usage for various replicationalgorithms

Figure 7 displays the mean job time based on changing number of jobs for LRU, DHR and proposed algorithm. It is clear that at the job number increases, DHRT is able to process the jobs in the lowest mean time in comparison with othermethods. It is similar to a real Grid environment where a lot of jobs should be executed.

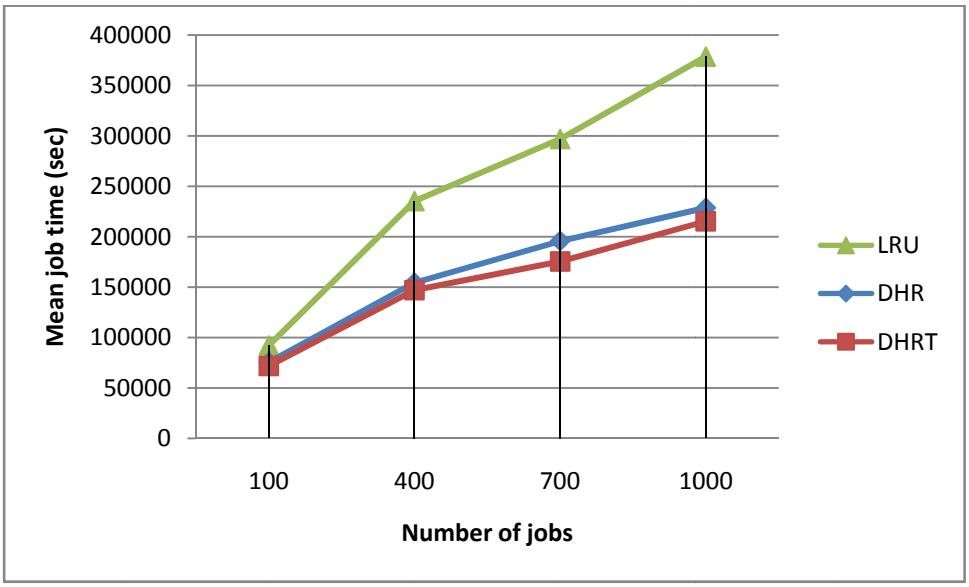

Fig. 7. Mean job execution time based on varying number of jobs

5. Conclusion and Future works

Data Grid is the highlight in the development of the Gridtechnology, which can be treated as a suitable solution for highperformance and data-intensive computing applications.Replication strategies are important mechanism used in Data Grid environments that helps to reduce access latency and network bandwidth utilization. Replication also increases data availability thereby enhancing system reliability. This technique appears clearly applicable to data distribution problems in large-scale scientific collaborations, due to their globally distributed user communities and distributed data sites.

.In this paper, a new replication strategy named DynamicHierarchical Replication with Threshold (DHRT) for a 3 level hierarchical structurenetwork is proposed. The goal is to increase data availability and thus improves dataaccess time. DHRT stores the replica in appropriation by defining a dynamic threshold, so the Mean Job ExecutionTime can be minimized and the storage usage is also reduced. It deletes those file that exist in local LAN and has minimum transfer time, when free space is not enough for the new replica.Also, It stores the replicas in the best sites where the file has beenaccessed most, instead of storing files in many sites. We alsopresented a replica selection strategy that selects the best replicalocation for the users' running jobs by considering the number of stored file on each location. Toevaluate the efficiency of the proposed replication

strategy, gridsimulator OptorSim is configured to represent a real world datagrid testbed. The simulation results shows that proposed algorithm performs better when compared to other traditionalalgorithm such as LRU, LFU and no replication and a new algorithm such as DHR.

In future work, more realistic scenarios and user accesspatterns can be investigated and DHRT can be combined with a proper schedulingto improve performance. We also plan to investigate more replicareplacement strategies to further improve the overall systemperformance.Data transferring between different grid sitesis time consuming and consequently scheduling jobs to the appropriate sites isnecessary. Replica selection can also be extended by considering additional parameters such as security. Searching for advanced replica replacement methods certainly enhances replication strategies.

References

[1] M. Li and M. Baker, "*The grid core technologies"*, *John Wiley & Sons*, 2005.

[2] N. Mohd. Zin, A. Noraziah, A. Che Fauzi, and T. Herawan, "Replication Techniques in Data Grid Environments", in *Intelligent Information and Database Systems*, vol. 7197, Eds. Springer Berlin, Heidelberg, pp. 549–559, 2012.

[3] K. Ranganathan and I. Foster, "Decoupling computation and data scheduling in distributed data-intensive applications", in *11th IEEE International Symposium on High Performance Distributed Computing*, pp. 352–358, 2002.

[4] S. Naseera and K. V. M. Murthy, "Agent Based Replica Placement in a Data Grid Environement", in *First International Conference on Computational Intelligence, Communication Systems and Networks*, pp. 426–430, 2009,.

[5] F. Ben Charrada, H. Ounelli, and H. Chettaoui, "Dynamic period vs static period in data grid replication", in *International Conference on P2P, Parallel, Grid, Cloud and Internet Computing (3PGCIC)*, pp. 565–568, 2010.

[6] W. Hoschek, J. Jaen-Martinez, A. Samar, H. Stockinger, and K. Stockinger, "Data management in an international data grid project", in *Grid Computing*, Springer, pp. 77–90, 2000.

[7] A. Chervenak, I. Foster, C. Kesselman, C. Salisbury, and S. Tuecke, "The data grid: Towards an architecture for the distributed management and analysis of large scientific datasets", in *Network Computing Application*, vol. 23, no. 3, pp. 187–200, 2000.

[8] I. Foster and C. Kesselman, "*The Grid 2: Blueprint for a New Computing Infrastructure*", Morgan Kaufmann, 2003.

[9] S. Venugopal, R. Buyya, and K. Ramamohanarao, "A taxonomy of data grids for distributed data sharing, management, and processing", in *Acm Computing Surveys*, vol. 38, no. 1, p. 3, 2006.

[10] K. Ranganathan, A. Iamnitchi, and I. Foster, "Improving data availability through dynamic model-driven replication in large peer-to-peer communities", in *2nd IEEE/ACM International Symposium on Cluster Computing and the Grid*, pp. 376–376, 2002.

[11] S. Goel and R. Buyya, "Data replication strategies in wide area distributed systems", in *Enterprise Service Computing: From Concept to Deployment*, vol. 17, 2006.

[12] K. Ranganathan and I. Foster, "Identifying dynamic replication strategies for a high-performance data grid", *Grid Computing*, pp. 75–86, 2001.

[13] N. Mansouri and G. H. Dastghaibyfard, "A dynamic replica management strategy in data grid", in *Journal of Network and Computer Applications*, vol. 35, no. 4, pp. 1297–1303, 2012.

[14] O. Wolfson and A. Milo, "The multicast policy and its relationship to replicated data placement", ACM Transactions on Database Systems-TODS, vol. 16, no. 1, pp. 181–205, 1991.

[15] N. F. Tzeng and G. L. Feng, "Resource allocation in cube network systems based on the covering radius", Parallel and Distributed Systems, IEEE Transactions on, vol. 7, no. 4, pp. 328–342, 1996.

[16] V. R. Sonigo, "Optimal replica placement in tree networks with qos and bandwidth constraints and the closest allocation policy", preprint arXiv:0706.3350, 2007.

[17] M. Tu, P. Li, L. Xiao, I.-L. Yen, and F. B. Bastani, "Replica placement algorithms for mobile transaction systems", IEEE Transactions on Knowledge and Data Engineering, vol. 18, no. 7, pp. 954 – 970, 2006.

[18] S. M. Park, J. H. Kim, Y. B. Ko, and W. S. Yoon, "Dynamic data grid replication strategy based on Internet hierarchy", in *Grid and Cooperative Computing*, pp. 838–846, 2004.

[19] M. Tang, B.-S. Lee, C.-K. Yeo, and X. Tang, "Dynamic replication algorithms for the multi-tier Data Grid", in *Future Generation Computer Systems*, vol. 21, no. 5, pp. 775–790, 2005.

[20] R. S. Chang and H. P. Chang, "A dynamic data replication strategy using access-weights in data grids", in *The Journal of Supercomputing*, vol. 45, no. 3, pp. 277–295, 2008.

[21] A. Horri, R. Sepahvand, and G. Dastghaibyfard, "A hierarchical scheduling and replication strategy", International Journal of Computer Science and Network Security- IJCSNS, vol. 8, no. 8, pp. 30–35, 2008.

[22] M. Shorfuzzaman, P. Graham, and R. Eskicioglu, "Adaptive popularity-driven replica placement in hierarchical data grids", in *The Journal of Supercomputing*, vol. 51, no. 3, pp. 374–392, 2010.

[23] N. Mansouri, "An Effective Weighted Data Replication Strategy for Data Grid", in *Australian Journal of Basic and Applied Sciences*, vol. 6, no. 10, pp. 336–346, 2012.

[24] S. Venugopal, R. Buyya, "An scope-based heuristic approach for scheduling distributed data-intensive applications on global grids", in*Journal of Parallel and Distributed Computing*, pp. 471–487, 2008.

[25] A. Chakrabarti, S. Sengupta, "Scalable and distributed mechanisms for integrated scheduling and replication in data grids", in *Proceedings of 10th International Conference on Distributed Computing and Networking (ICDCN)*, 2008.

[26] A. Chervenak, E. Deelman, M. Livny, M. Su, R. Schuler, S. Bharathi, G. Mehta, K. Vahi,"Data placement for scientific applications in distributed environments", in *Proceedings of IEEE/ ACM international conference on grid computing*, 2007.

[27] N. N. Dang, S. B. Lim, "Combination of replication and scheduling in data grids". In *International Journal of Computer Science and Network Security*,pp. 304–308, 2007.

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

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

[30] D. G. Cameron, R. Carvajal-Schiaffino, P. Millar, C. Nicholson, K. Stockinger, F. Zini, "Evaluating scheduling and replica optimization strategies in OptorSim", in *Forth international workshop on grid computing*, Phoenix, USA, 2003.

[31] W. H. Bell, D. G. Cameron, R. Carvajal-Schiaffino, P. Millar, C. Nicholson, K. Stockinger, F. Zini,"OptorSim v1.0 installation and user guide", 2004.