A Dynamic Replica Creation and Eviction Mechanism based on Files Dependencies in a Federated Grid Environment

Musa Sule Argungu[1,2], Suki Arif[1] and Mohd Hasbullah Omar[1]

[1]*Universiti Utara Malaysia, Malaysia, {argungu@internetworks.my, suki1207@uum.edu.my, mhomar@uum.edu.my}*

[2]*Department of Computer Science, University of Science and Technology, Aliero, Kebbi State Nigeria*

ABSTRACT

For quite a while, data replication has attracted the attention of many researchers in the field of data grid system, with a few researchers focusing on data replication in a federated data grid environment. The central goal of data replication mechanisms is to improve data locality, by creating more copies of the most popular files, and evicting unpopular files. In the conventional data grid systems, a common practice was to evaluate popular files based on economic approach, which evicts unpopular files based on least frequently or least recently used factor, to create space for incoming file replicas. However, this method tends to ignore the possible logical dependencies amongst the replica files, which leads to the inappropriate evaluation of popular /unpopular files and consequently may result to the eviction of files with high logical dependencies on other files that may be needed later by other files or applications. As a resolution to this problem, a dynamic replica creation and eviction mechanism (DRCEM) is proposed by this research work, which considers files logical dependencies, while evaluating popular or unpopular files and before evicting files from the system.

Keywords: Data Grid, Data Grid Federation, Data Replication Mechanism, File Dependency.

I INTRODUCTION

This write-up seeks to develop a dynamic replica creation and eviction mechanism (DRCEM) for improving the performance of Data Grid Federation (DGF) systems, regarding jobs times, storage usage and bandwidth consumption. DGF belongs to grid computing paradigm, and it is formed by joining more than one data grid system or computing clusters together via a federation mechanism or software according to research by (Bihani, Oliver, & Liu, 2016). Furthermore, DGF is a large-scale resource management system consisting of data and compute resources, linked via Peer-to-Peer connections (Jagatheesan & Moore, 2004).

Thus, DGF provides a means for managing different types of data grid systems connected from different locations and computing background. As a point of difference between the two concepts; whereas the data grid provides the facility for naming, organizing, as well as management of data on diverse distributed computing resources, the DGF provides a way for naming, organizing, and managing data on multiple data grids according to (Jagatheesan & Moore, 2004). Figure 1 shows an abstract view of DGF system. It shows four regions connected via Peer-to-Peer connections over WAN networks.

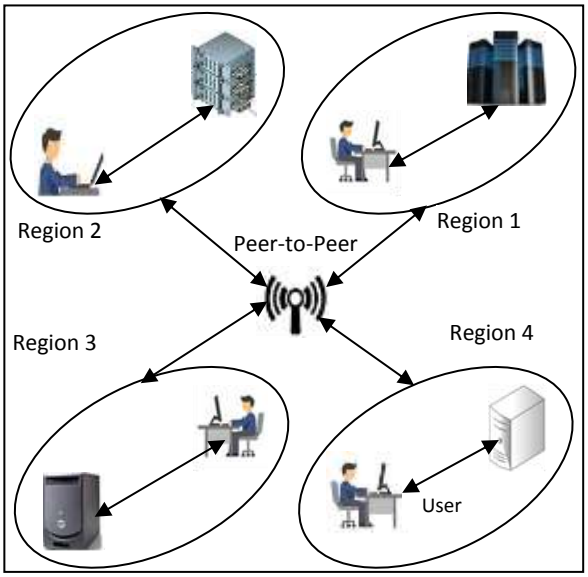

Figure 1. Data Grid Federation Architecture

Data replication aims to improve data locality, by creating more copies of the most popular files and evicting unpopular files. In the conventional data grid systems, a common practice was to evaluate and create more copies of popular files based on economic approach, which evicts unpopular files based on least frequently or least recently used factor to create space for incoming file replicas (Mansouri & Asadi, 2014). However, this method tends to ignore the possible logical dependencies amongst the replica files, which leads to the inappropriate evaluation of popular /unpopular files and consequently may result to the eviction of files with high logical dependencies on other files that may be needed later by other files or applications.

Existing researchers attempted to address the issue of data replication within both data grid platform and its federation. However, majority of the researchers focused on data replication within the conventional data grid systems (Amjad, Sher, & Daud, 2012), with a few researches targeted to DGF systems (Ciubăncan & Dulea, 2017).

In this study, a dynamic replica creation and eviction mechanism (DRCEM) is proposed for improving the performance of DGF systems with peer level connection abilities. The proposed DRCEM mechanism strikes a balance between two existing data replication mechanisms. The first is the enhanced least access lowest weight (ELALW) mechanism by the authors (Mansouri & Asadi 2014). The second is the data replica creation mechanism (DRCM) by (Madi, 2012). Both ELALW and DRCM were also based on the European DG infrastructural topology (Bonacorsi et al., 2015).

The issues addressed by this research include file inter-dependability, required data availability and storage resource requirement while deciding to create new replicas for data files. The proposed DRCEM mechanism incorporates all these factors for proper replica creation and eviction decision, which were overlooked by the existing researchers. Before describing the proposed mechanism, the next section discusses the related works reviewed by this research work.

II LITERATURE REVIEW

The fundamental difference between DG and DGF systems is the way the replication mechanisms handle replica management regarding replica creation and eviction decisions according to (Chang, Chang, & Wang 2008). Some mechanisms consider replica creation on the individual regions of the DGF systems, as reported by (Park, Kim, Ko, & Yoon, 2003), while others consider the entire federation system according to (Ranjan, Buyya, & Harwood, 2008). The proposed DRCEM considers the entire federation system for replica creation and eviction by taking account of the individual DGF regions in a stepwise progression. This section explores some the popular research works on data replication, based on which the proposed DRCEM mechanism was developed.

Research by (Park, Kim, Ko, & Yoon, 2003), proposed a data replication mechanism for addressing the problem of data availability within DGF environment based on files' access history within regions with high bandwidth concentration. The mechanism improves access latency and remote site access, by selecting a file for replication using access history. However, restricting data access to sites with high bandwidth may overburden the system, and hence results in reducing the overall system performance.

Data availability within DGF regions was considered by the researchers (Chang, Chang, & Wang 2008) aimed at improving data locality by proposing a mechanism called least access lowest weight (LALW). The LALW mechanism duplicates data objects based on access frequency with the aim of improving access time and transfer time. However, even though the mechanism identified the region within which to place file replicas; it falls short of explicitly determining the ideal locations within the regions, which are suitable for hosting the duplicated data objects. Also, the replicating mechanism failed to determine sites distances, which will have a drastic effect on jobs completion time. The other shortcoming is that LALW failed to consider logical dependencies amongst the replica files, which is a crucial parameter in file evaluation for determining file popularity and its worth in relation to users and other files. Also, the mechanism did not consider response time, which affects the decision for selecting file replicas, as well as the jobs completion time.

The work of (Madi, 2012), proposed a replica creation algorithm for DG systems (DRCM), by capitalizing on the shortfalls of the LALW mechanism. The replica selection decision was based on files with high weight or rate of growth based on the LALW mechanism proposed by (Chang, Chang, & Wang 2008). The mechanism aimed to minimize network bandwidth consumption and storage cost. However, it failed to find the availability of the site to hold file replica and did not consider response time as well. Another issue is that DRCM did not correctly address the issue of file dependability, as it only considers direct logical dependability (DLD), ignoring indirect logical dependability (ILD) between replica files.

Also, the research by (Almomani & Madi, 2014), proposed a GA-Based replica placement mechanism for DG systems to address the issue of management of large data files in DG environment that provides massive data resources across geographically distributed systems. The GA-Based mechanism aimed at improving data resource sharing capability via replication. The mechanism identifies a suitable location to place file replicas, using five design metrics (read cost, storage cost, sites' workload, and replication site). However, the mechanism failed to consider file inter-dependability amongst replica files, desired data availability, and storage constraints.

The research by (Mansouri & Asadi 2014), proposed an enhancement of the LALW mechanism referred to as the "Enhanced LALW" mechanism or ELALW. The mechanism create file replica by considering how many times the file wold be requested in the near future, which was based on the work of research work by (Chang, Chang, & Wang 2008). ELALW used economic approach, which evaluates popular files based on access frequencies. However, the approach ignores the possible logical dependability amongst the file replicas.

Thus, DRCEM resolves this problem by taking account of file logical dependencies, in addition to determining the required data availability within each region as well as the storage requirement to host the newly created file replicas. The proposed DRCEM mechanism was deployed and tested in a data grid simulation environment OptorSim and compared the results with two other mechanisms namely DRCM and ELALW on the number of jobs and jobs times. The next section explains the experimental settings and the algorithm for the proposed DCREM mechanism.

III SIMULATION ENVIRONMENT AND SETTINGS

As explained in the previous section, OptorSim simulator was used in this research to evaluate the performance of the proposed mechanism alongside the existing mechanisms. OptorSim is extensively used in the simulation of DG systems as explained severally by different researchers, such as (Manjaiah & Guroob, 2017) and (Chang & Chang, 2008) as well as in DGF related data optimisation problems according to (Mohamad, Ahmad, Rose, Mohamad, & Deris, 2016) and (Vashisht & Sharma, 2014). OptorSim is a tool designed to test numerous replication optimisation approaches in a simulated grid environment before they are deployed in the real grid system, especially simulating data access optimisations mechanisms. OptorSim uses discrete event simulation framework. The OptorSim architecture for Peer-to-Peer replication optimisation of data resources, which comprises of the access mediator, storage mediator, and the Peer-to-Peer mediator, is shown in Figure 2.

The basic settings and parameters used in this setup are shown in Table 1.

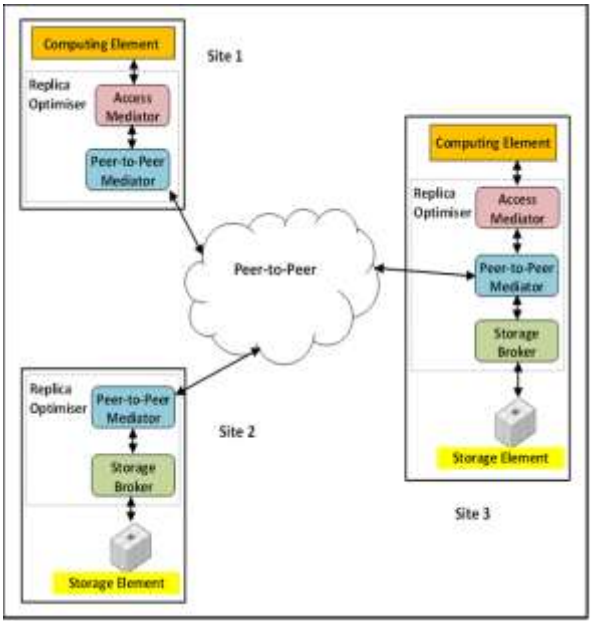

Figure 2. OptorSim Architecture

Table 1. Configuration parameters

Parameter	Value
Number of Jobs	50, 500, 1000, 2000, 3000, 4000, 5000
File Size	10GB, 5GB, 2.5GB
Scheduler	QAC scheduler

Table 1. Continued.

Site Policy	All Job Types
Access history length	1000000 ms
Storage metric (D)	0.67
Max. Queue Size	400
Job Delay	2500 ms

From the table, it could be seen that the number of jobs was varied from 50 to 5000, on the interval of 500, 1000, 2000, in that order. Also, the file size was varied from 2.5 GB, 5 GB to 10 GB. The same job scheduler was used throughout the simulation to maintain consistency in the obtained results.

The performance of the proposed mechanism as well the existing mechanisms were observed based on a number of jobs against job completion time, network, and storage usage. These are outputted in a graph format as shown in Figures 4(a-c).

The overall DRCEM logic is outlined in Figure 3. In addition, Figure 5 compares the effect of a number of jobs and amount of replica files created by each of the mechanisms for varying file sizes. However, due to space constraints, the only plot for 10 GB file size is shown in this write up for all the mechanisms.

Input: *Number of access for each file (NoA(file$_s$)), number of access intervals t, indirect logical dependency (IDL), file size, bandwidth between sites, number of existing replicas of each file (NOR(file));*
Output: *Creating copies of popular files replicas for placement to suitable locations,*
Procedure*:*

1. */* **Dynamic Replica Creation and Eviction Mechanism (DRCEM)** */*
2. *Use file access history workload data file;*
3. *for each file in the grid regions, given the number of access within the past time intervals (T1, T2…Tn);*
4. */* evaluate individual files to determine popular and unpopular files */*
5. *Find average increase/decrease rate for all files at specified intervals;*
6. *Calculate file lifetime for all files = file access frequency for the upcoming time intervals;*
7. *Compute Indirect logical dependence(ILD) for individual files*
8. *Compute file weight for all files;*
9. *Compute file vale for all files;*
10. *Compute required data availability = Projected no of replicas(PNoR);*
11. *Compare PNoR with zero;*
12. *if (PNoR > 0) then*
13. *Add file$_s$ to the Popularity_List File Records;*
14. *else if (PNoR < 0) then*
15. *Add file$_s$ to Unpopularity_List File Records //for eviction at a later time ;*
16. *else if ($PNoR$ = 0) then*
17. *Add file$_s$ to the Stability_List;*
18. */* **Dynamic Replica Eviction System (DRES)** */*
19. */*Invoked when there is not enough storage space for incoming files replicas*/*
20. *for each file$_i$ in the best_replica_site file records;*
21. *Calculate indirect logical dependencies for all files;*
22. *Check space availability, FV and ILDs for site$_i$, file$_i$;*
23. *Sort file$_i$ in asccending order of ILDs*
24. *if optimum CDS$_{si}$ < SR (file$_i$); /*highly loaded*/*
25. *if ILD (file$_i$) < Average ILD (file$_i$); /*less file dependability*/*
26. *Select files that their size >= SR;*
27. *Delete files with least ILDs;*
28. *Create Replica (site$_i$, file$_i$);*
29. *Repeat for all*
30. *Return*
31

Figure 3. Layout for DRCEM Mechanism

The algorithm encapsulates scheme for Dynamic Replica Eviction system (DRES) within the DRCEM mechanism. This is invoked after creating the required number of replicas, and there is not enough space to host them.

IV RESULTS AND DISCUSSION

The following Figures 4(a-c) indicate results for jobs completion times, effective network usage (ENU) and storage element usage (SEU) against a number of jobs for the three mechanisms evaluated using file dependency as a design metric. All the mechanisms showed similarities in the results patterns. From 50 to 3000 jobs, MJET keeps rising for all the mechanisms, with DRCEM recording less completion times, which makes it complete faster than all the other mechanisms. However, jobs completion time drops as the number of jobs increases due to access history. This indicates that the mechanism keeps track of visited sites, so that subsequently, little time is required for locating relevant data files. Similarly, the ENU drops as the number of jobs increases, while the SE usage increases with increasing number of jobs. Also, Replication increases with increase in lobs number as indicated on Figures 4(a-c).

From the Figures (4a-4c), DRCEM shows better performance regarding the measured metrics, which include jobs completion times, effective network usage (ENU) and storage element usage (SEU), indicating that it is a better alternative mechanism for replica creation and eviction compared to existing ones. The advantage of DRCEM over the existing mechanisms is that DRCEM evaluates popular files more accurately, and evicts files with caution, performs less number of replications (Figure 5) without compromising the job completion time. Les number of replica creation saves storage space. Also, less replica creation conserves network bandwidth usage, thereby improves on the job completion time.

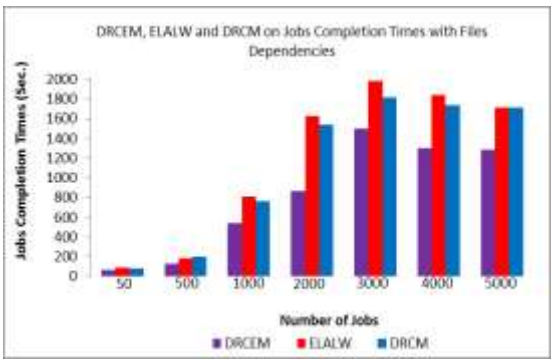

Figure 4a. Number of Jobs vs. Jobs Completion Times

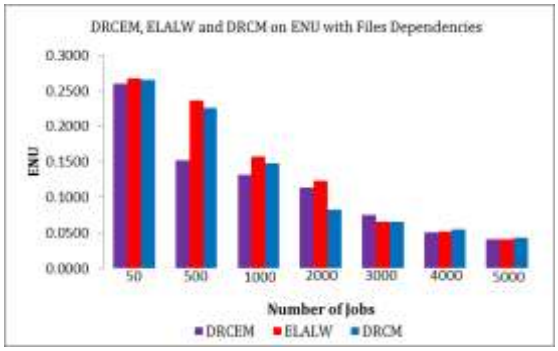

Figure 4b. Number of Jobs vs. Effective Network Usage (ENU)

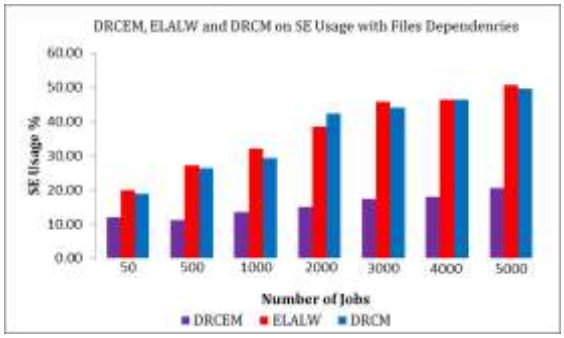

Figure 4c. Number of Jobs vs. Storage Element Usage (SEU)

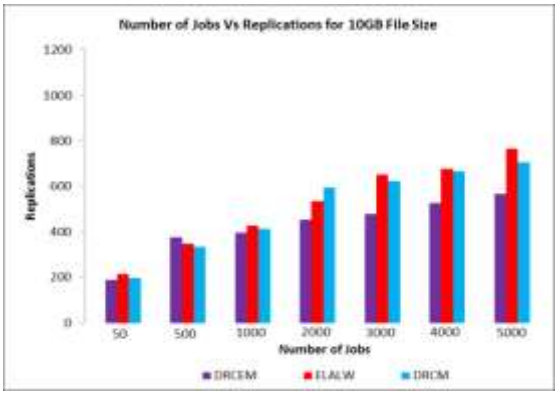

Figure 5. Number of Jobs vs. Number of Replications

V CONCLUSION AND FUTURE WORK

From the simulation results and the ongoing discussion, DRCEM proves more effective regarding storage management; network usage as well executes jobs in a faster period. In the future, the mechanism will be implemented in a federated cloud environment. Also, the mechanism will be evaluated against higher number of jobs to ascertain its scalability

ACKNOWLEDGEMENTS

The authors wish to thank the Ministry of Education, Malaysia for funding this study under the Long Term Research Grant Scheme (LRGS/bu/2012/UUM/Teknologi Komunikasi dan Infomasi).

REFERENCES

Bihani, B., Oliver, B. K., & Liu, C. (2016). *U.S. Patent Application No. 14/866,585*.

Moore, R. W., Jagatheesan, A., Rajasekar, A., Wan, M., & Schroeder, W. (2004, April). DATA GRID MANAGEMENT SYSTEMS. In *NASA/IEEE MSST 2004 Twelfth NASA Goddard Conference on Mass Storage Systems and Technologies* (p. 1).

Mansouri, N., & Asadi, A. (2014). Weighted data replication strategy for data grid considering the economic approach. *Int. J. Comput. Elect. Auto. Control Inf. Eng*, *8*, 1336-1345.

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.

Ciubăncan, M., & Dulea, M. (2017, September). Implementing advanced data flow and storage management solutions within a multi-VO grid site. In *Networking in Education and Research (RoEduNet), 2017 16th RoEduNet Conference* (pp. 1-4). IEEE.

Bonacorsi, D., Boccali, T., Giordano, D., Girone, M., Neri, M., Magini, N., ... & Wildish, T. (2015). Exploiting CMS data popularity to model the evolution of data management for Run-2 and beyond. In *Journal of Physics: Conference Series*(Vol. 664, No. 3, p. 032003). IOP Publishing.

Chang, R. S., Chang, H. P., & Wang, Y. T. (2008, March). A dynamic weighted data replication strategy in data grids. In *Computer Systems and Applications, 2008. AICCSA 2008. IEEE/ACS International Conference on* (pp. 414-421). IEEE.

Kim, J. S., Lee, W. J., & Jeon, C. H. (2008). A Dynamic Data Grid Replication Strategy Based on Internet Architecture. *Journal of the Institute of Electronics Engineers of Korea CI*, *45*(3), 1-6.

Ranjan, R., Harwood, A., & Buyya, R. (2008). A case for a cooperative and incentive-based federation of distributed clusters. *Future Generation Computer Systems*, *24*(4), 280-295.

Madi, M. K. (2012). *Replica Creation Algorithm for Data Grids* (Doctoral dissertation, Universiti Utara Malaysia).

Almomani, O., & Madi, M. (2014). A GA-Based Replica Placement Mechanism for Data Grid. *International Journal of Advanced Computer Science and Applications (IJACSA)*, *5*(10).

Manjaiah, D. H., & Guroob, A. H. (2017, February). Triple integration optimization techniques in data grid environment using OptorSim simulator. In *Data Management, Analytics and Innovation (ICDMAI), 2017 International Conference on* (pp. 138-144). IEEE.

Chang, R. S., & Chang, H. P. (2008). A dynamic data replication strategy using access-weights in data grids. *The Journal of Supercomputing*, *45*(3), 277-295.

Mohamad, Z., Ahmad, F., Rose, A. N. M., Mohamad, F. S., & Deris, M. M. (2016). Implementation of Sub-Grid-Federation Model for Performance Improvement in Federated Data Grid. *Malaysian Journal of Applied Sciences*, *1*(1), 55-67.

Vashisht, P., Kumar, R.. & Sharma, A. (2014) "Efficient dynamic replication algorithm using an agent for the data grid," *The Scientific World Journal*.