Journal of Computer Science 4 (2): 103-110, 2008
ISSN 1549-3636
© 2008 Science Publications

Access Weight Replica Consistency Protocol for Large Scale Data Grid

[1]Mohammed Radi, [1]Ali Mamat, [2]M. Mat Deris,
[1]Hamidah Ibrahim and [1]Subramaniam Shamala
[1]Faculty of Computer Science and Information Technology,
Universiti Putra Malaysia, 43400 Serdang, Selangor, Malaysia
[2]Faculty of Information Technology and Multimedia,
University of Tun Hussein Onn, 86400 Parit Raja, Batu Pahat, Johor, Malaysia

Abstract: Replication is a well known technique to improve reliability and performance for a Data Grid. Keeping consistent content at all distributed replica is an important subject in Data Grid. Replica consistency protocol using classical propagation method called the radial method suffers from high overhead at the master replica site, while line method suffers from high delay time. In Data Grid not all replicas can be dealt with in the same way, since some will be in greater demand than others. Updating first replica having most demand, a greater number of clients would access the updated content in a shorter period of time. In this study, based on asynchronous aggressive update propagation technique, a scalable replica consistency protocol is developed to maintain replica consistency in data grid which aimed to reaching delay reduction and load balancing, such that the high access weight replicas updated faster than the others. The simulation results shows that the proposed protocol is capable of sending the updates to high access replicas in a short period while reducing the total update propagation repose time and reached load balancing.

Key words: Data grid, replication, update propagation, access weight, aggressive copy

INTRODUCTION

Large-scale scientific applications such as high energy physics, data mining, molecular modeling, earth sciences and large scale simulation produce large amount of datasets[1,2]. Data Grid is one of the applications in Grid computing which is developed to manage the large scale Data applications. Data Grid has been using replication to reduce access latency, improve data locality, increase robustness, scalability and performance for distributed applications. Several data replication techniques[3,4] have been developed to support high-performance data access, improving data availability and load balancing to remotely produce scientific data. Most of those techniques do not provide the replica consistency in case of updates. Grid data management middleware usually assumes that (1) whole files are the replication unit and (2) replicated files are read only. However there are requirements for mechanisms that maintain consistency for a modifiable data. Shared data sets are updated by a master site and other sites might read old replicas that are not the latest version. Some applications do not require such dirty read to all replicas. For these applications, in order to keep the consistency among replicas, data updates occurring on a master site should be immediately propagated to other site holding its replicas. In general, the data consistency problem deals with keeping two or more data items consistent, but in data grid consistency is keeping replicas up to date[5]. In distributed database as well as in distributed file systems, content distribution networks web applications, several solution of replica synchronization already exist by using optimistic consistency protocols[6]. The current algorithm for replica synchronization can be classified into two categories, synchronous and asynchronous replication. Synchronous replication is expensive and not scalable on the network, which are suitable for a small system. Asynchronous replication algorithms are less expensive and more scalable. In asynchrony replication the update can be either pull based or push based or hybrid. In a pull based the secondary replica site ask the master replica site for some new updates, while in a push based the master replica site push the updates to all secondary replica sites. Push based update propagation is more suitable for the application that need the updates to immediately reach the secondary replica site.

Corresponding Author: Mohammed Radi, Faculty of Computer Science and Information Technology, Universiti Putra Malaysia, 43400 Serdang, Selangor, Malaysia

Many researchers studied replica consistency in data grid environment. Those researchers make effort in consistency maintenance. Early work propose a grid consistency service was in[5] it gives five models for different levels of consistency provided to grid user and also discusses how they can include into replica consistency service for Data Grid. In[7] stated the basic service interface and isolate the functionalities of the process of replica update synchronization. The basic components required by the consistency service were also defined. A few studies have been done to maintain replica consistency in Data Grid and propose a consistency models.

The literature discusses several replication and data consistency solutions. Including lazy replication[8,9], aggressive replication[8-11] and hybrid replica consistency services[12-16]. They make effort in performance, availability and consistency, but not study the efficiency, scalability of the protocol itself. Since they use classical update propagation technique in which the updates take a long time to reach all replica sites and high loads at the master site.

In most data grid applications, some replicas tend to have more access demands than the other replicas due to the geographical distributions, the number of clients and the number of request arising from more instance work among clients. Replica access weight must be taken into account when designing algorithm for maintaining replica consistency in Data Grid. Access weight has been utilized in[12,16] and gives better performance. But none of literature gives propriety to the high accesses weight replica. A replica consistency protocol giving priority to the delivery of updates to replica with higher access weight will be able to satisfy requests from more clients with up-to-date content in less time.

In this study a replication protocol is proposed, in which the updates reach other replicas using a propagation technique based on nodes organized into a logical structure network that enables the technique to scale well for thousands of replicas while minimizing update response time and reach load balance. On the other hand, it also gives priority propagating the updates to the high access weight replica. Moreover the efficiency of the replica consistency protocol, emphasis is also given on considering the high access weight replica first. The updates information is expected to be sent to the high access weight replica faster than other replicas, without affecting the performance of the consistency.

SYSTEM MODEL

In the model a data sharing model is considered in a federated organization in which replica consistency catalog has information about the sites that replicate the same data. Furthermore it is assumed that a large number N of grid sites communicates through message passing, a full replication environment is assumed. No assumptions are given to control the network environment and topology. As shown in Fig. 1, the grid environment concerned consists of N grid sites connected through the internet and there is a communication system that allows any node to communicate with each other. The data grid sites communicate by exchanging messages through internet, there is no global shared memory and an active grid site is able to communicate with any other online site. The underlying communication network may lose or duplicate messages and may not guarantee any order of delivery.

The Data Grid clients make request which can be a job request or an update request to its local site. When the client invokes an update request to the server, this update must be propagating to all replicas by sending an update message, the update message carries the update information. In this the access weight of the replica site is measured as the number of service requests by their client per time unit for a given replica. The access weight will be different from site to site. Fig. 2 shows the access weight for a group of replica

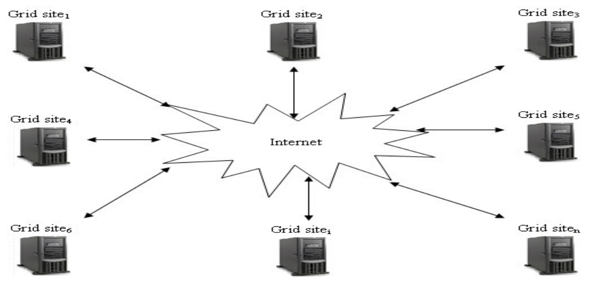

Fig. 1: The network model

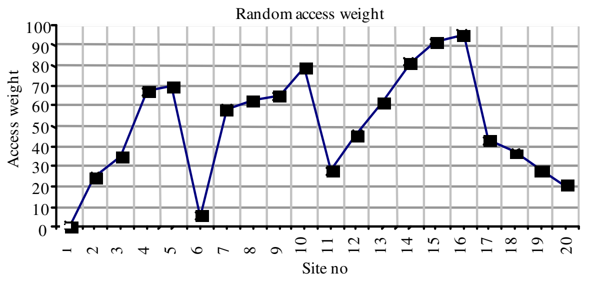

Fig. 2: The sites access weight for Fi data set

sites for a given replica. There is a group of replicas in which the Y axis corresponds to their access weight as the y value tends to decrease to zero, the access weight increase. As such the replicas with greatest y value is the replicas with smaller access weight.

MATERIALS AND METHODS

In our model each data file has Logical File Name (LFN) to denote a unique logical identifier for desired data content and each physical copy is specified by a unique Physical File Name (PHN), that specifies its location on a storage system. Therefore each data file has only one LFN and many PFNs.

In the consistency service it is important to distinguish between the following types of replica since they have different semantics and access permissions for the clients:

Master replica: The replica that can be updated by the clients of the Data Grid master replica is always up-to-date. Usually it is located at the data owner site.

Secondary replica: The replica that the clients have a read only access to it and to be updated by the replica consistency service.

The basic idea of a single master replication is that for each data item or file there exist one master copy and all other replicas are secondary replicas. The user updates only pass to the master replica, which executes the updates and then propagates the changes in the form of update information to all secondary replicas. The secondary sites receive the update information and apply the modification to its home replica.

Replica consistency architecture: In order to provide the required functionalities of the single master replication, Replica Consistency Service (RCS) architecture is proposed. Local Replica Consistency Service (LRCS) and Replica Consistency Catalogue (RCC) are the main components of the architecture:

- Local Replica Consistency Service (LRCS): Two types of LRCS are proposed, the first one is master replica which is the main entry of the update requests from users. The second one is the secondary replica and it is responsible for updating its local replica and relay the update propagation process if necessary
- Replica Consistency Catalogue (RCC) is used to store the metadata, this metadata will used by the RCS, moreover the RCC will manage the DUPG. The Data Grid sites and the replica catalogue server are connected through the network
- Resources Monitoring Service (RMS): This service is responsible for monitoring resource availability at a given grid node as well as collecting statistics about the resource usage and data access request

The interactions between the above components are shown in Fig. 3, this interaction can be explained through a simple case user wishing to update a master file Fi. In a basic scenario, a user passes the update request and the target LFN to the master replica site LRCS. The LRCS updates the local replica and reflect a consistent view to its user and then start the propagation process. In the propagation process master replica site LRCS inquires the RCC about the PFNs of those replicas. RCC gets information from RMS and sends back all information to LRCS. After that LRCS starts the update propagation technique. When the secondary replica site receives the updated information, it runs the update propagation algorithm and applies the updates locally, then reflects the updates to its users.

Replica consistency technique: All replica sites are logically organized in the form of two-dimensional grid structure. For example as shown in Fig. 4A is UPG consisting of 16 replica sites. The sites are organized in the form of 4X4 grids. Each site has one master replica and the other sites are secondary replicas. We calls each site as UPG (I, J), where I refers to the rows and J refer to the columns.

Update propagation process: When the master site client initiates an update to master copy, the update propagation process started, the update propagation process is shown in Fig. 4. As shown in Fig. 4A the round 0 of the update propagation in which the master replica site sends the update information to the first site in each column. The first site in each column

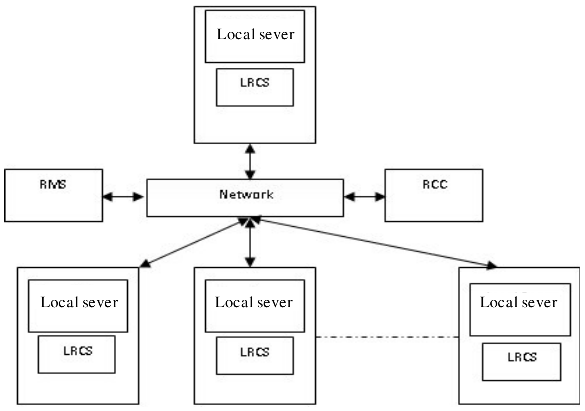

Fig. 3: Replica consistency architecture

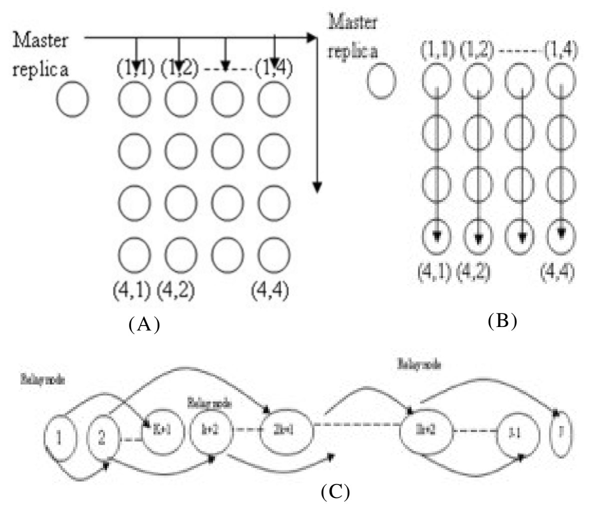

Fig. 4: The push phase of the update propagation process, (A): Round 0 of the push phase, (B): Other Round of the push phase, (C): Detail of the other rounds of the push phase for the column i

propagates the update information in parallel with other first sites, as shown in Fig. 4B where the first site at each column send the update information to k sites and instruct one of this sites to relay the update information and so on, this phase is completed when the update information reach all replica sites at each column as shown in Fig. 4C. While UPG give better performance than the radial method the update message reach the replica sites in random order. In our algorithm the updates reach the replica sites with heist access weight faster by selecting the k sites orderly by the access weight instead of random order.

Update propagation algorithm: The proposed algorithm for update propagation is called Access weight Update Propagation Grid (AUPG) and it is described in this section, first the explanation of the new data structure of the two dimensional Grid site. Each node in the grid has the following data structure:

Site id	PFN	Timestamp	Access_weight

Where:

Site id	=	Id of the site in the Grid
PFN	=	The physical file name of the updated file
Timestamp	=	Counter starting with 0 and increases by one for each update
Access_weight	=	The access weight of the replica at a given time

```
AccessUpdateUPG

Input: site S(I, J) update file Fi,  inconsistent replicas state

Output: a faire update propagation to all replica sites.

  AccessUpdateUPGMaster( M, Fi)

  1   $F_{iTimestamp} \leftarrow F_{iTimestamp} +1$.
  2   Sort the sites at each column in a increasing order
      by the access weight
  3   For each column J
  4   Prepare the Update_Information_Message
      <S[1,j],S[master], $F_i$ ,M,
  5   Timestamp($F_i$),Rlay_list(j)>
  6   Send the prepared update information message to
      all columns.
  7   Set a timeout to retransmit the message if the
      acknowledgments are not received.

  UpdateUPGSecondary(Update_Information_Message)

  8   Counter = 0
  9   While relay_list<> nil&& Q ≤ k
  10  Enqueue(S(Q+1,J) Relay_list(j))
  11  Prepare an update information message
  12  Send the message to S[Q,J]
  13  Set a timeout to retransmit the message if the
      acknowledgments are not received.
  14    Apply the updates M to the local replica
  15  $F_{iTimestamp}p \leftarrow F_{iTimestamp} +1$.
  16  Send a message to RCC to change the sitecase state
      to consistent,
```

Fig. 5: Update UPG algorithm

When the master site client initiates an update operation and after the master site processes the updates locally, the update propagation process is started. UpdateUPG algorithm propagates the update information from a single source (master site) to all replica sites in a UPG (I, J). Suppose that the owner S (master) updates the master replica of a file Fi. At this time it sends an invalidation message to RCC and gets the Fi UPG information from RCC, then initiate the UpdateUPG algorithm shown in Fig. 5. UpdateUPG algorithm has two parts, the first part is UpdateUPGMaster at the master site and the second part is UpdateUPGSecondary at the secondary replica site.

The UpdateUPG algorithm works as follows, Line 1 in UpdateUPGMaster performs the increment of the timestamp for the file Fi. Line 2 is to sort the site at each column in ascending order according to access weight. The algorithm maintains each column of the F_iUPG at line 4-6, line 4 to prepare the update message containing the receiver site, the sender site, the file, the update information, the relay list which contains the entire replica site in the column j. Line 5 is to send the update information message to the target site. Line 6 is to set an acknowledgement timeout for retransmit of the message if not received by the target site.

The lines 7-16 are start at site S(I, J) when the site S(I, J) receive the update message. Line 7 is to initialize the counter value. The while loop in lines 9-12 iterates as long as the Relay_list not empty or the counter not reaching k. The algorithm maintains the first k sites from the relay list at line 9-12, line 9 is to enqueue the site from the Relay_list, line 10 is to prepare the update message containing the receiver site, the sender site, the file, the update information and the relay list.

Validation: The goal of the proposed protocol is to reduce the update propagation response time and reach load balance. It also gives priority to deliver the updates to the replica with higher demand which will satisfy requests from more clients with up-to-date data in less time. Analysis the delay time of the update information to each site will then be done. The delay is defined as the total hop count from the master copy to each site that receives the update information. And the average load balance is defined as the average number of sites to which each site propagates the update information.

Analytical model: An analytical model is developed in this section, started from a completely consistent state and analyses a single update request. Delay time required to reach a consistent state is then evaluated. For the purpose of analysis it is assumed that the replica sites in UPG will form a two dimensional Grid. Table 1 shows some parameters and measurement matrices used in the analysis.

In the AUPG the delay time in a virtual network is of concern, so it is considered that the time of communication between two grid sites equal to one time unit as a constant model T = 1. The delay time of an update information to a given grid site can be defined as the number of hop count from the master replica to that grid site. To analyze the UPG method for N nodes it will be assumed that the nodes formulate a rectangular grid UPG (I, J), where N = I×J. In the first round the master site send the update information to I nodes. The delay time for the I nodes is 1.

Table 1: Analytical model parameters

Notation	Description
N	Number of replica sites
I	Number of columns
J	Number of sites in each column (number of rows)
K	Number of relay nodes
Round (I,I)	The round in which the site (I,J) will receive the update information
Delay (I,J)	The update propagation delay (in hop)

When any sites receive the update information in second round the delay is equal to 2 and the site receiving the updates in the third round the delay time will be 3. In general, if the sites receive the update information in the mth round the delay time will be m. The equation that give the number of rounds in which the site (I, J) will receive the update information is defined as:

$$\text{Round (I,I)} = ((\text{I}-1) \text{ div k})+1 \tag{1}$$

This implies that the site UPG(I,J) will receive the update information with delay time equal to:

$$\text{Delay (I,J)} = \text{Round (I,I)} \tag{2}$$

The second step is to compute the average delay time for each replica that is in the same access weight range. The replica site is divided according to access weight into 10 groups. The first group contains the replicas with access weight between 0 and 10, the second group contains replicas with access weight between 11 and 20 and the tenth groups contain the replicas with access weight between 91 and 100. The average delay time for each group is then computed:

$$\text{Avgdelay(min..max)} = \frac{\sum_{i=0}^{n\backslash j} \text{G(I,J).delay} \forall \text{G(I,J)}, \text{G(I,J).access weight} \in [\text{min} - \text{max}]}{\text{number of site in the group}} \tag{3}$$

Simulation model: A discrete event simulator has been developed in a C++ language based on the analytical model in section 6.1. The goal of the simulation is to study the effectiveness of the proposed protocol.

In the assumed simulation model, the number of replica sites in the enter system is n, where n = 100×h (h = 1, 2…10). The n sites construct UPG where the number of columns is set to be 5 columns. And the number of relay sites K is chosen to be 5 in the first experiment and then changed in experiment 2 from 1...10. Table 1 summarized the parameters and their value used in the simulation experiments. Each experiment was repeated 10,000 times.

RESULTS AND DISSECTION

Two experimental studies has been conducted, the first experiment provides a general update propagation response time balance analysis comparing the proposed protocol to the radial and line methods. The second experiment evaluates the effectiveness of using the access weight approach.

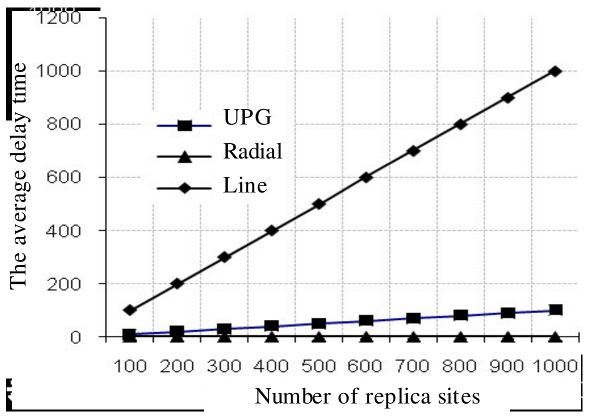

Fig. 6: Update propagation response time and the number of replica sites

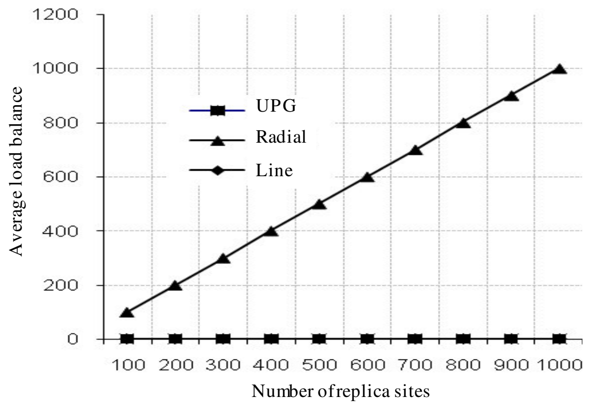

Fig. 7: Average load balance and the number of replica sites

Experiment 1: The experiments provide first, update propagation response time and average load balance in order to give a feeling of the effect of the proposed protocol. Furthermore the experiments compare the proposed protocol with the radial and the line methods. The experiments were run many times and varying the number of N, where N = 100×h (h = 1, 2…10).

Figure 6 shows the update propagation response time and Fig. 7 the average load balance for the number of replica sites. The horizontal axis in Fig. 6 and 7 shows the number of sites in the networks. The vertical axis indicates the update propagation response time in Fig. 6 and average load balance in Fig. 7.

For the proposed protocol and the line technique the update propagation response time increased with the number of replica site. On the other hand the update propagation response time maintained constant and equal to one in the redial technique because the master site sends the updates to all replica sites since the number of hubs is equal to one. These results are obvious considering the characteristics of the two methods

The average load balance for the line propagation technique is always 1 because each site propagates the update information to another one replica site. The average load of the radial propagation is always equal to the number of replica sites. The results considered the characteristics of the two methods. On the other hand the average load balance of UPG method is at most 5, because in UPG method each site propagates the update information to not less than 2 sites.

Comparing the average load balance in the three techniques, the line technique can distribute the loads into all sites and reach the load balance. On the other hand the proposed protocol keeps the average load balance less than 5.

Experiment 2: Now focus on the effectiveness of using the access weight approach. The protocol without considering the access weight called UPG and after considering the access weight as AUPG. The number of replica sites in the enter system is n where n is 500 and 1000 sites constructing a two-dimensional grid where the number of column is set to be 5 columns. And the number of replay sites K is chosen to be 5 Table 2 summarized the parameters and their value used in the simulation experiments. The parameters are basically fixed to constant value, but n changed from 500 to 1000. Simulations were carried out with 500, 1000 grid sites and experiments were repeated 10,000 times. The results can bee shown in Fig. 8 and 9.

Table 2 simulation parameters

Parameter	Value
N	100×h (h = 1, 2…10)
I	5
J	N/J
K	5

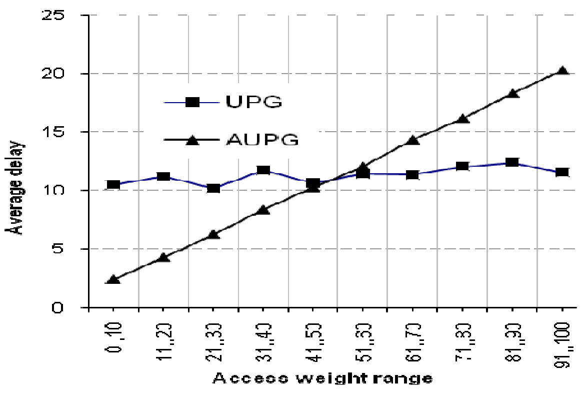

Fig. 8: Average delay time with 500 sites

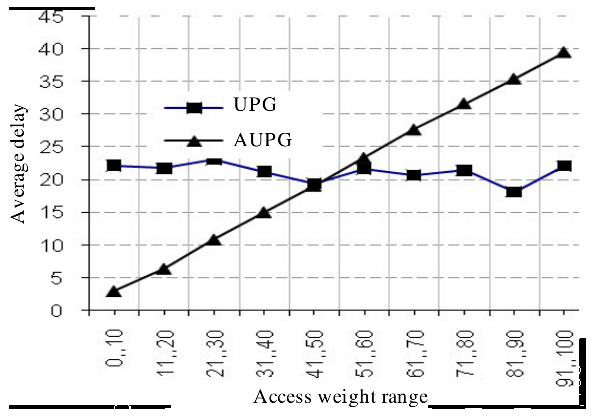

Fig. 9: Average delay time with 1000 site s

In both Fig. 8 and 9 the horizontal axis indicates the access weight range and the vertical axis indicates average delay time. In both case with 500 and 1000 replicas, it was observed that the updates arrive quickly to the grid sites with most demand, causing these replicas to reach the state of consistency faster. The time it takes for the update message to reach grid site is the propagation delay associated to that site, the comparison in Fig. 8 and 9 is expressed in average number of hubs needed for the updates to reach a grid site with a given range of access weight.

Comparing the group average delay time in the access weight based technique and the original technique, the access weight based technique sends the updates to the high access demand sites faster than the original technique because it gives a high priority to the high access weight sites. However the updates will reach the low access demand sites slower.

It can be concluded that the enhanced protocol (AUPG) caused the high access weight site to reach the state of consistency on average of 2, in the case of 500 replica sites and also on average 2 in the case of 1000 replica site and still give better average delay time for the sites with high access weight until 41-50.

The idea of showing the result of the proposed algorithm with different number of replica sites is to analyze the scalability of the proposed algorithm. As can be seen in the Fig. 8 and 9 the algorithm scaled well even by increasing the number of replica sites to 1000 sites.

The experiments have provided performance analysis of the replication protocol. To summarize the results, it can be clearly seen that the protocol reduces the average load compared with the radial propagation method and reduced the update propagation response time compared to the line propagation method. Besides it was able to send the updates to the high access replica faster. Thus it can be confirmed that the UPG achieved both load balance and delay reduction for update propagation in Data Grid network and gave priority to the high access weight replicas.

CONCLUSION

In this study we proposed an access weight based replica consistency protocol for a large scale Data Grid, in which the updates reach the replicas with high access weight (greatest demand) at a short time. The other replicas will also be updated but at a normal time. The protocol requires managing access weight for reaching replica sites by the RMS. The research has provided a quantitative performance analysis of the proposed replication protocol. The results showed that the proposed update propagation technique can reach both the load balancing and minimize the update propagation response time compared with the radial and line technique, while it is scalable for high number of replica sites. Moreover it is also capable of sending the updates to the high access demand replica sites faster.

REFERENCES

1. Chervenak, I. Foster, C. Kesselman, C. Salisbury and S. Tuecke, 2001.The data grid: Towards architecture for the distributed management and analysis of large scientific datasets. J. Network Comput. Appli., 23:187-200.
2. Allcock, J. Bester, J. Bresnahan, A.L. Chervenak, I. Foster, C. Kesselman, S. Meder, V. Nefedova, D. Quesnal and S. Tuecke, 2002. Data management and transfer in high performance computational grid environments. Parallel Comput. J., 28: 749-771,
3. Lamehamedi, H. and B. Szymanski, 2007. Decentralized data management framework for data grids. Future Generat. Comput. Syst., 23: 109-115.
4. M.Mat Deris, J.H. Abawajy, Ali Mamat.2008, An Efficient Replicated Data Access Approach for Large-Scale Distributed Systems. Journal of FGCS, Elsevier, vol. 24, no. 1, pp 1-9. doi:10.1016/j.future.2007.04.010.
5. Dullmann, W. Hoschek, J. Jaen-Martinez, B. Segal, H. Stockinger, K. Stockinger and A. Samar, 2001. Models for replica synchronisation and consistency in a data grid. In: 10th IEEE International Symposium on High Performance Distributed Computing (HPDC-10

'01), August 7-9, IEEE Computer Society Press, San Francisco, CA, USA, pp: 0067.

6. Saito, Y. and M. Shapiro, 2005. Optimistic Replication. ACM Computing Surveys (CSUR), pp: 42-81.
7. Domenici, A. *et al*., 2004. Replica Cpnsistency in Data Grid. Nyclear Istruments and Methods in Physics Reserch Sechtion, pp: 24-28.
8. Sun, Y. and Z. Xu, 2004. Grid replication coherent protocol. In: 18th International Parallel and Distributed Procssing Symposium (IPDPS'04), IEEE, Beijing, pp: 232-239.
9. Jaechun, No, Chang, Park and Sung-Soon, Park. 2006. Data Replication Techniques for Intensive Applications. International Conference on Computational Science Reading, UK, May 28-31, 2006 . Springer Berlin / Heidelberg, pp.1063-170. doi:10.1007/11758549_141
10. R. Wang, S. Wu, R. Chang. , 2007 A Novel Data Grid Coherence Protocol Using Pipeline-Based Aggressive Copy Method.. Second International Conference, GPC 2007. Paris, France, May 2-4, 2007: Springer Berlin / Heidelberg. pp. 484-495. 10.1007/978-3-540-72360-8_41
11. Domenic, A. *et al*., 2006. Relaxed data consistency with CONStanza. In: Sixth IEEE International Symposium on Cluster Computing and the Grid (CCGRID'06), IEEE Computer Society, New York, USA, pp: 425-429.
12. Chang, R. and J. Chang, 2006. Adaptable replica consistency service for data grids. In: Proceedings of the 3rd International Conference on Information Technology: New Generations (ITNG'06).
13. Hu, J. *et al*., 2005. An asynchronous replica consistency model in data grid. In: Workshop 7th International Workshop on Service Grid Computing and Applications (SGCA 2005), Springer Berlin/Heidelberg, pp: 475-484.
14. Belalemi, G. and Y. Slimani. 2007.Consistency management for data grid in optorsim simulator. Int. J. Multimedia Ubiquitous Eng., pp: 103-118.
15. Huang, C., F. Xu and X. Hu. 2006. Massive data oriented replication algorithms for consistency maintenance in data grids. In: 6th International Conference Computational Science-ICCS 2006, Springer Berlin/Heidelberg, UK, pp: 838-841.
16. Yang, C.T. *et al*., 2007. A one-way file replica consistency model in data grids. In: IEEE Asia Pacific Services Computing Conference, Tsukuba Science City, IEEE Computer Society, Japan, pp: 364, 373.