Managing Data Using Neighbor Replication on Triangular-Grid Structure

Ali Mamat[1], M. Mat Deris[2], J.H. Abawajy[3], and Suhaila Ismail[1]

[1] Faculty of Computer Science and Information technology,
University Putra Malaysia,
43400 Serdang, Selangor, Malaysia.
`ali@fsktm.upm.edu.my`
[2] University College of Tun Hussein Onn,
Faculty of Information technology and Multimedia,
P.O.Box 101, 86400 Parit Raja, Batu Pahat, Johor.
`mmustafa@kuittho.edu.my`
[3] Deakin University, School of Information Technology, Geelong, VIC, Australia
`jemal@deakin.edu.au`

Abstract. Data is one of the domains in grid research that deals with the storage, replication, and management of large data sets in a distributed environment. The all-data-to-all sites replication scheme such as read-one write-all and tree grid structure (TGS) are the popular techniques being used for replication and management of data in this domain. However, these techniques have its weaknesses in terms of data storage capacity and also data access times due to some number of sites must 'agree' in common to execute certain transactions. In this paper, we propose the all-data-to-some-sites scheme called the neighbor replication on triangular grid (NRTG) technique by considering only neighbors have the replicated data, and thus, minimizes the storage capacity as well as high update availability. Also, the technique tolerates failures such as server failures, site failure or even network partitioning using remote procedure call (RPC).

1 Introduction

With the proliferation of computer networks, PCs and workstations, new models for workplaces are emerging [1]. In particular, organizations need to provide current data to users who may be geographically remote and to handle a volume of requests of data distributed around multiple sites in distributed database environment. Therefore, the storage, availability, accessibility and consistency are important issues to be addressed in order to allow distributed users efficiently and safely access data from many different sites. One way to provide access to such data is through replication. In particular, there is a need for minimal replication in order to provide an efficient way to manage the data and minimize the storage capacity.

One of the solutions is based on synchronous replication [2,3]. It is based on quorum to execute the operations with high degree of consistency [4] and also to ensure serializability. Synchronous replication can be categorized into several schemes, i.e., all data to all sites (full replication), all data to some sites and some data to all sites.

However, full replication causes high update propagation and also needs high storage capacity [4,5,6]. Several studies model partial replication in a way that each data object is replicated to some of the sites. However, it is still undefined which copies are placed on which site, such that different degrees of quality of a replication scheme can be modeled. A few studies have been done on partial replication techniques based on some data items to all sites using tree structure technique [7,8]. This technique will cause high update propagation overhead. Thus, some-data-items-to-all-sites scheme is not realistic. The European DataGrid Project [9,10] implemented this model to manage the file-based replica. It is based on the sites that have previously been registered for replication. This will cause the inconsistence number of replication occurs in the model. Also, the data availability has very high overhead as all registered replicas must be updated simultaneously. Thus, an optimum number of sites to replicate the data are required with non-tolerated availability of the system.

The focus of this paper is on modeling a technique based on synchronous solution to minimize the storage and optimize the system availability of the replicated data in a data grid. We describe the neighbor replication on triangular grid (NRTG) technique by considering only neighbors have the replicated data. This assignment provides a higher availability of executing write operations in replicated database due to the minimum number of quorum size required.

This paper is organized as follows: In Section 2 the model and the technique of the NRTG is presented. In Section 3, the replica management is discussed. In Section 4, the performance of the proposed technique is analyzed in terms of availability, and a comparison with tree grid structure technique is given.

2 Neighbor Replication on Triangular Grid (NRTG) Technique

2.1 Model

Briefly, a site *i* initiates a NRTG transaction to update its data object. For all accessible data objects, a NRTG transaction attempts to access a NRTG quorum. If a NRTG transaction gets a NRTG write quorum without non-empty intersection, it is accepted for execution and completion, otherwise it is rejected. We assume for the read quorum, if two transactions attempt to read a common data object, read operations do not change the values of the data object. Since read and write quorums must intersect and any two NRTG quorums must also intersect, then all transaction executions are one-copy serializable.

2.2 The NRTG Technique

In NRTG, all sites are logically organized in the form of triangular grid structure. It is inherited from binary-tree structure with the inner leaves of the tree linked together as shown in Fig. 1.

Each site has a master data file. In the remainder of this paper, we assume that replica copies are data files. A site is either operational or failed and the state (operational or failed) of each site is statistically independent to the others. When a site is operational, the copy at the site is available; otherwise it is unavailable.

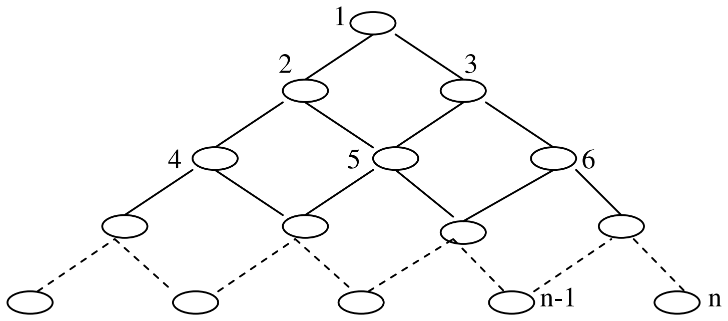

Fig. 1. Triangular-neighbor organization of n sites

For example, from Fig. 1, data from site 1 will replicate to site 2 and site 3 which are neighbors to it. Site 5 has four neighbors, which are sites 2, 3, 8, and 9. As such, site 5 has five replicas. From this property, it is clear that each site required with the maximum number of replications is 5, thus, minimizing the storage capacity as compared with full replication scheme.

3 Managing Replication Under NRTG

The quorum for an operation is defined as a set of sites (called *replicas*) whose number is sufficient to execute that operation. In other words, it is modeled as a set of replicas: $C = \{C_1, C_2, \ldots, C_n\}$, where $i=1,2,\ldots,n$, are called the sequence numbers of these replicas. Each replica, C_i manage a set of data: $D_i= \{d_1^i, d_2^i, \ldots, d_m^i\}$. The consistency constraint requires that $D_i \equiv D_j$, for $i,j = 1,2,\ldots,n$. The replication management of NRTG is analogous with the one that proposed in [11].

Each replica in the system provides a number of remote procedures that can be called by users for processing the data managed by the replica.

3.1 The Coordinating Algorithm for the Primary Replica

When a primary replica receives an NRTG-Procedure-Call from a transaction manager, it uses the coordinating algorithm to maintain the consistency of all replicas in terms of NRTG-Procedure-Call. This section describes the coordinating algorithm.

In the coordinating algorithm, the primary replica uses the 2PC protocol to ensure the replication consistency. In the first phase, the primary replica asks the NRTG quorum whether it can be formed or not. If the NRTG quorum can be formed (replicas under NRTG quorum return a Successful (1) for such execution), the primary replica returns a SUC to the transaction manager. If the transaction manager requests a commit, then in the second phase the primary replica asks all replicas to commit the NRTG-Procedure-Call execution. If NRTG quorum cannot be formed, then the primary replica returns a (0) to the transaction manager and asks all replicas to abort the operation in the second phase. If operation returns a Partial-Commit (-1) (the primary replica returns a PC and other non-primary replicas may return a -1), the primary replica returns a -1 to the transaction manager. The primary replica also records the number of replicas that return a -1. This number will be used in conflict resolution during the recovery process.

3.2 The Cooperating Algorithm for NRTG Replicas

When a NRTG replica receives a request from a primary replica, it checks whether the request can proceed or not and acts accordingly.

- If the request can be performed, the NRTG replica locks the required data and returns 1. Later if the primary replica asks to commit the operation, the neighbor performs the operation and releases the lock. If the neighbor asks to abort the operation, it will release the lock.
- If the neighbor finds that the operation cannot be executed (the required data is not free), it then returns an 0.
- If the NRTG replica finds that the data is in a partially committed state, then it returns a PC to the primary replica. The primary replica will then partially commits the operation and record the event when the primary replica asks to partially commit the operation. If the primary replica asks to abort the operation, then the non-primary replica aborts the operation.

The 2PC protocol used by NRTG-Procedure-Call transaction manager guarantees that the replicas will be in a consistent state if a transaction returns a 1 or 0. However, to guarantee all replicas do the same thing to the transactions with -1 pending, a majority consensus should be reached among all replicas.

The order of each partially committed NRTG-Procedure-Call over a data is also important when a re-join of network partitions carries out of these NRTG-Procedure-Calls.

If the network is partitioned into two disconnecting parts, they will eventually be re-united again. In this case, replicas in both partitions have partially committed updates. The conflict resolution algorithms during recovery process are responsible to make replicas in these two parts consistent. When recovering from a network partition, replicas of each part of the partition have to send a ‘re-uniting’ message to replicas of the other part.

The NRTG-RPC returns -1 only when the network is partitioned. If the network is partitioned into two disconnecting parts, the two parts will eventually be re-united again. In this case, replicas in both partitions may have some partially committed updates. The conflict resolution algorithm is responsible to make replicas in these two parts consistent.

4 Performance Analysis and Comparisons

4.1 Availability Analysis

In estimating the availability, all copies are assumed to have the same availability *p*. $C_{X,Y}$ denotes the communication cost with X technique for Y operation, which is R(read) or W(write).

4.1.1 TGS Technique

Let h denotes the height of the tree, D is the degree of the copies in the tree, and $M = \lceil(D+1)/2\rceil$ is the majority of the degree of copies. The availability of the read and write operations in the TGS can be estimated by using recurrence equations based on the tree height h. Let AR_{h+1} and AW_{h+1} be the availability of the read and the write operations with a tree of height h respectively. Then the availability of a read operation for a tree of height h+1 can be represented as:

$$AR_{h+1} = p + (1-p) \sum_{i=M}^{D} \binom{D}{i} AR_h^i \left(1 - AR_h\right)^{D-i}$$

and the availability of a write operation for a tree of height h+1 is given as:

$$AW_{h+1} = p \sum_{i=M}^{D} \binom{D}{i} AW_h^i (1 - AW_h)^{D-i} \qquad (1)$$

where p is the probability that a copy is available, and $AR_0 = AW_0 = p$.

Thus system availability for TGS is

$$\text{Av(TGS,r,w)} = f AR_{h+1} + (1\text{-}f) AW_{h+1} \qquad \ldots(2)$$

where f and (1-f) are the probability that an arriving operation of read and write for data file x, respectively.

4.1.2 NRTG Technique

Let p_i denote the availability of site i. Read operations on the replicated data are executed by acquiring a read quorum and write operations are executed by acquiring a write quorum. For simplicity, we choose the read quorum equals to the write quorum. Thus, the quorum size for read and write operations equals to $\lfloor L_{Bx}/2 \rfloor$, that is,

$A_{NRTG,R} = A_{NRTG,W} = \lfloor L_{Bx}/2 \rfloor$. For example, if the primary site has four neighbors, each of which has vote one, then $C_{NRTG,R} = C_{NRTG,W} == \lfloor 5/2 \rfloor = 3$.

For any assignment B and quorum q for the data file x, define $\varphi(B_x,q)$ to be the probability that at least q sites in $S(B_x)$ are available, then

$\varphi(B_x,q) = \Pr\{\text{at least q sites in } S(B_x) \text{ are available}\}$

$$= \sum_{G\in Q(B_x,q)} \left(\prod_{j\in G} p_j \prod_{j\in S(B_x)-G} (1-p_j) \right) \qquad \ldots (3)$$

Thus, the availability of read and write operations for the data file x, are $\varphi(B_x,r)$ and $\varphi(B_x,w)$, respectively. Let $Av(B_x,r,w)$ denote the system availability corresponding to the assignment B_x, read quorum r and write quorum w. If the probability that an arriving operation of read and write for data file x are f and (1-f), respectively, then

$$Av(B_x,r,w) = f\,\varphi(B_x,r) + (1\text{-}f)\,\varphi(B_x,w).$$

Since in this paper, we consider the read and write operations are equal, then the system availability,

$$Av(B_x,r,w) = \varphi(B_x,r) = \varphi(B_x,w). \qquad \ldots (4)$$

4.2 Performance Comparisons

4.2.1 Comparisons of Write Availabilities

In this section, we will compare the performance on the write availability and system availability of the TGS technique based on equations (1) and (2), and our NRTG technique based on equations (3) and (4) for the case of n=13, 40 and 121. In estimating the availability of operations, all copies are assumed to have the same availability.

Fig. 2 and Table 1, show that the NRTG technique outperform the TGS technique. When an individual copy has availability 88%, write availability in the NRTG is approximately 99% whereas write availability in the TGS is approximately 82% for n=13. Moreover, write availability in the TGS decreases as n increases. For example, when an individual copy has availability 86%, write availability is approximately 78% for n=13 whereas write availability is approximately 73% for n = 121.

Table 1. Comparison of the write availability between TGS and NRTG under the different set of copies p=0.8,…,0.98

Techniques	Write Availability								
	0.82	0.84	0.86	0.88	0.9	0.92	0.94	0.96	0.98
NRTG(R/W)	.956	.968	.978	.986	.991	.996	.998	.999	1.00
TGS(W),n=13	.692	.738	.782	.823	.861	.896	.927	.955	.979
TGS(W),n=40	.634	.697	.755	.807	.853	.892	.926	.954	.979
TGS(W),n=121	.571	.656	.731	.794	.847	.890	.925	.954	.979

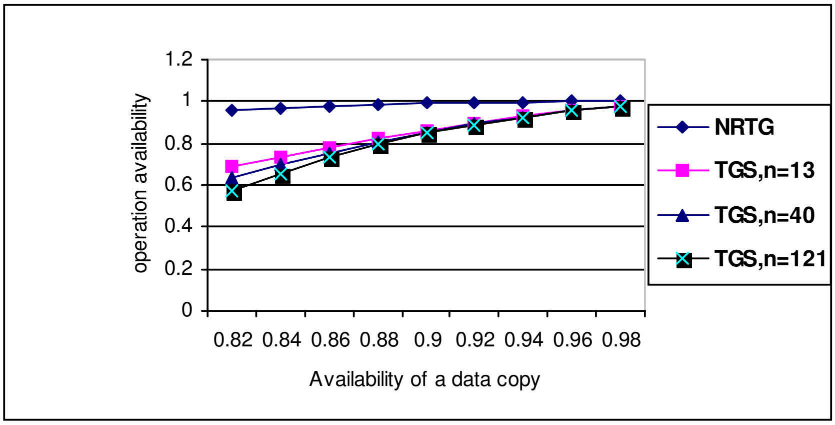

Fig. 2. Comparison of the write availability between TGS and NRTG

5 Conclusions

In this paper, a new technique, called neighbor replication on grid (NRTG) structure has been proposed to manage the data in distributed database environment. It showed that, the NRTG technique provides a convenient approach to minimize the storage, high availability for update-frequent operations by imposing a neighbor binary vote assignment to the logical grid structure on data copies, and also data access time. This is due to the minimum number of replication sites required. The fault-tolerant programs developed in the environment are able to tolerate failures such as server failure, a site failure or even a network partitioning. In comparison to the TGS structure, NRTG provides higher system availability, which is preferred for large systems.

References

1. O. Wolfson, S. Jajodia, and Y. Huang,"An Adaptive Data Replication Algorithm,"*ACM Transactions on Database Systems*, vol. 22, no 2 (1997), pp. 255-314.
2. J. Holliday, R. Steinke, D. Agrawal, and A. El-Abbadi,"Apidemic Algorithms for Replicated Databases", *IEEE Trans. On Know. and Data Engineering*, vol.15, No.3 (2003), pp.1-21.
3. H. Stockinger,"Distributed Database Management Systems and The Data Grid",IEEE-NASA Symposium, April 2001, pp. 1-12.
4. Budiarto, S. Noshio, M. Tsukamoto,"Data Management Issues in Mobile and Peer-to-Peer Environment", *Data and Knowledge Engineering, Elsevier*, 41 (2002), pp. 183-204.
5. L.Gao, M. Dahlin, A. Nayate, J. Zheng, and A. Lyengar,"Improving Availability and Performance with Application-Specific Data Replication", *IEEE Trans. On Knowledge and Data Engineering*, vol.17, no. 1, (2005), pp 106-120.
6. M.Mat Deris, J.H. Abawajy, H.M. Suzuri,"An Efficient Replicated Data Access Approach for Large Scale Distributed Systems", *IEEE/ACM Conf. On Cluster Computing and Grid (CCGRID2004)*, April 2004, Chicago, USA.
7. M. Nicola and M. Jarke," Performance Modeling of Distributed and Replicated Databases", *IEEE Trans. On Knowledge and Data Engineering*, vol.12, no. 4, (2000), pp 645-671.
8. J.Huang, Qingfeng Fan, Qiongli Wu, and YangXiang He,"Improved Grid Information Service Using The Idea of File-Parted Replication", *Lecture Notes in Computer Science Springer*, vol. 3584 (2005), pp. 704-711.
9. P. Kunszt, Erwin Laure, Heinz Stockinger, Kurt Stockinger,"File-based Replica Management", *International Journal of Future Generation of Computer Systems, Elsevier,* 21 (2005), pp. 115-123.
10. H. Stockinger, F Donno, E. Laure, S. Muzaffar, P. Kunszt, "Grid Data Management in Action: Experience in Running and Supporting Data Management Services in the EU Data Grid Project", Int'l. Conf. On Computing in High Energy and Nuclear Physics, March 2003, La Jolla, California. http://gridpp.ac.uk/papers/chap3-TUAT007.pdf
11. W. Zhou and A. Goscinski,"Managing Replicated Remote Procedure Call Transactions", *The Computer Journal*, Vol. 42, No. 7, pp 592-608, 1999.