Exploiting Logical Structures to Reduce Quorum Sizes of Replicated Databases

Parul Pandey and Dr. Maheshwari Tripathi

Jain University, Bangalore
IET, Lucknow

Abstract.

Voting is a traditional mechanism used for maintaining the consistency of replicated data in distributed systems. A significant problem in protocols that use voting is the size of the quorum needed on each access to the replicated data. In this paper, we propose replica control protocols where the synchronization cost is reduced by exploiting the structural information of the underlying system. We also propose a novel logical structure for managing replicated data, by imposing a logical wheel structure on the set of copies of an object. The logical structure ensures minimum read quorum size of one, by reading one copy of an object while guaranteeing fault-tolerance of write operations.

Key words:

Replica-control, distributed database, quorum

1 Introduction

Fault- tolerance in a distributed database system is achieved by replication. Replication masks and tolerates failures in the network gracefully and increases availability. In particular, the system remains operational and available to the users despite failures. A problem that must be solved when using replication is how to maintain the copies in a consistent state [8]when multiple accesses are being made. There must exist a control protocol responsible for synchronizing the access so that the logical data is kept consistent. Weighted voting [7] which is a generalization of the majority consensus method presented in [16], is a popular method for maintaining consistency of replicated data. In the quorum consensus [10](QC) algorithm, each copy is assigned a non-negative weight [6] . A Read and write threshold, RT and WT respectively for x, such that both 2WT and (RT + WT) are greater than the total weight of all copies of x. A read (or write) quorum of x is any set of copies of x with a weight of at least RT (or WT). For better performance, some logical structure is imposed on the network, and the quorums are chosen under the consideration of such structures. Such logical structures include the tree [4], diamond [5], ring [12], triangular mesh [3], and grid [14] structures. A geometric approach for dealing with logical structures is proposed in [17].

In this paper we proposed replica control protocols where the synchronization cost is reduced by exploiting the structural information of the underlying system. A new logical structure, which is called The Wheel Structure , for managing replicated data is also proposed. The sites in the network are logically organized into a wheel structure. This structure can be viewed as specialized version of ring and tree protocol.

DOI : 10.5121/acij.2012.3108

2 Logical Structure Based Solutions

In this section, we discuss spectrum of distributed mutual exclusion solutions based on quorums. Protocols presented here illustrate that by logically structuring copies of data the cost of mutual exclusion can be reduced considerably .

2.1 The Grid Protocol

Maekawa [11] proposed arranging copies in a logical grid . This proves to derive efficient O(sqrtN)solutions for mutual exclusion. The protocol also assumes that node failures are fail-stop. A read quorum group contains exactly one node from each column. A write quorum group consists of nodes in a read group and all nodes in a column of the grid. Nodes are arranged in grid topology only conceptually, which is used to describe the protocol. The availability of quorums in the grid protocol is sensitive to failure patterns. In particular, if one column is not accessible, neither read nor write quorum can be formed. If we consider a $\sqrt{N}$ x $\sqrt{N}$ grid, the size of read and write quorum is $\sqrt{N}$ and $2\sqrt{N}$ -1, respectively. In the worst case, read and write operations are resilient to ($\sqrt{(N)}$ - 1 failures. In the best case, read and write operations are resilient to N - $\sqrt{N}$ and N - $2\sqrt{N}$ + 1 failures. For a grid protocol, it is assumed that the grid structure is approximately a square, the read quorum size is approximated by $\sqrt{}$ (N), and the write quorum size is approximated ny $2\sqrt{}$ (N).

2.2 The Tree Protocol

Tree protocol [1], logically organizes the Y copies of an object to form a complete binary tree; i.e., if k is the level of the tree, then it has 2^{k+1} - 1 copies, where the root is at level 0. The standard tree terminology, i.e., root, child, parent, leaf, etc., is used. A path in the tree is defined to be a sequence of copies s_1 , s_2 , . . s_i , s_{i+1} , s_n such that s_{i+1} is a child of s_i . Informal description of the algorithm for constructing a quorum for a binary tree is as follows. A quorum is constructed by selecting any path starting from the root and ending with any of the leaves. If successful, this set of copies constitutes a quorum. If a path cannot be constructed due to the inaccessibility of a copy c , residing on a failed or inaccessible site (due to partitioning failures), then the algorithm must substitute for that copy with two paths, both of which start with the children of copy c_i and terminate with leaves. Note that each path must terminate with a leaf, hence if the last copy in the path is inaccessible, the operation must be aborted. It is shown that in the best case the size of the quorum is log(n). This case arises when there are no failures or under certain favorable failure distributions. In the worst case, the size of the quorum increases to $\lceil n/2 \rceil$. This situation occurs, for example, when all interior nodes of the tree are inaccessible.

A simple extension of this protocol for read and write access requires that both read and write oper- ations obtain quorums using the tree protocol. Thus, the sizes of the read and write quorums range from log(n) to $\lceil n \rceil$. However, this scheme has the undesirable property that there is unnecessary redundancy since two read operations always have a nonempty intersection. Thus, the efficiency of this solution is at the expense of unnecessary redundancy for read operations.

The availability of quorums in the tree protocol is also sensitive to failure patterns. In the worst case, the failure of log(n) sites, which form a path in the tree, prevent the formation of the quorum. On the other hand, if every site except the above log(n) sites fail, a quorum can still be constructed. Thus in the best case the protocol can tolerate n - log(n) failures.

2.3 The Hierarchical Protocol

The hierarchical quorum consensus protocol [9] logically organizes a set of copies of an

object in a database into a multilevel tree (of depth) with the root as level 0. Higher level nodes of the tree correspond to logical groups and leaves store physical copies of an object. A node at level i, where i varies from 0 to m-1 is viewed as a logical group which in turn consists of subgroups at level i+1. A quorum is associated with each level and to access a logical group at a certain level, a quorum consisting of its subgroups must be first assembled. A read (write) quorum at level i is defined as the number of subgroups of a level $i-1$ group l_{i+1} that must be locked by a read (write) operation to obtain read (write) access to the group. The read (write) quorum at level i is denoted by r_i (w_i) Note that this is a recursive definition. Therefore, each level i group must in turn assemble r_{i+1} of its subgroups at level $i + 1$, and so on. This would eventually translate into a quorum consisting of physical copies of the object. For $i =$ 1...m, r_i, w_i and l_i, must satisfy the following constraints:

$$r_i + w_i > l_i \qquad (1)$$

$$2 * w_i > l_i \qquad (2)$$

To perform read (write) operations on the replicated object, a read (write) quorum at level 0 must be obtained first. The quorum size of this protocol is $N^{0.63}$. Contrast this with the majority consensus method where the quorum size is (N+1)/2. It is evident that the quorum size in the hierarchical quorum consensus protocol will grow at a much slower rate with respect to N than in the majority voting method.

2.4 The Ring Protocol

In the ring protocol [12] copies are organized into a ring structure. It uses the adjacency property to reduce the read and write quorums. There are two protocols - The flat ring protocol and the hierarchical ring protocol. Flat ring arranges nodes in a single ring and achieves a read quorum of two copies (constant), and a write quorum equal to the majority of copies. The hierarchical ring protocol uses a multi-level ring structure and is a generalization of the flat ring protocol. Hierarchical quorum in general achieves smaller write quorum than the majority of the copies and read quorum is independent on the total number of copies. Both protocols are tolerant to multiple failures and do not need any special reconfiguration procedures when failure occur. For the special case taken in [12], best and worst quorum sizes are given by $q_r = n^{\log_d 2}$ and $q_r = (\lfloor \frac{d}{2} \rfloor + 1)\log_d n$.

2.5 Crumbling Walls

The elements are arranged in rows of varying widths, and a quorum is made by one full row and a single representative from every row below the full row. A crumbling wall [13] with a sequence of row widths $n = n_1, n_2, ..., n_d$ is denoted by CW(n).

A wall as a non - dominated coterie iff the first row is of width 1 and rows 2.., d are of width >=2. Best crumbling wall CWlog system, with quorum of size $\lg n = \lg \lg n$ is introduced. CWlog has high availability for small universe sizes as well; its availability is much better than the Grid and slightly better than tree.

2.6 The Triangular Mesh Protocol

Nodes are organized into a triangular mesh [3]. A k-triangular mesh consists of a vertex set V and an edge set E. V is defined as a set of (x,y) tuples where x,y are both integers, 0<= x <= k-1, 0 <= y <= k-1 and 0<=x+y<=k-1. The y-axis is slanted to the right 30 degree to accommodate the left-hand side of the right triangle. E is defined as a set of vertex pairs (v_1,v_2), where v_1 and v_2 are in V, $v_1 = (x_1, y_1)$, $v_2 = (x_2, y_2)$ and x_1,

$y1$, $x2$, $y2$ satisfy one of the following conditions:

$$x_1 - x_2 = y_2 - y_1 = 1 \quad (3)$$

$$x_2 - x_1 = y_1 - y_2 = 1 \quad (4)$$

$$|x_1 - x_2| + |y_1 - y_2| = 1 \quad (5)$$

Quorum in triangular mesh consists of a centre and subquorum I for i=0,1,2. Subquorum I consists of nodes of a subcolumn parallel with a side determined by the type of quorum, starting at the centre and ending at side i. The quorum size in our three triangular-mesh-based protocols is k, which is $\lceil\sqrt{(2N)}\rceil$. It can tolerate up to (k – 2) node failures.

2.7 Diamond Quorum Consensus

The sites in the network are logically organized into a 2- dimensional diamond structure [5]. Diamond is actually as a specialized version of grid protocol because it is a grid with holes. It also resembles the crumbling walls protocol. However crumbling walls doesn't consider read and write operations. Its superior to previous protocols of majority, tree , grid, hierarchical quorum : 1) It has the highest read capacity , 2) It has the smallest optimal read quorum of 2. To form a write quorum, we can choose all nodes of any one row plus an arbitrary node for each remaining rows. Read quorum can be formed by choosing any entire row of nodes, or by using an arbitrary node of each row. Minimum read quorum can be obtained by choosing the whole top and bottom row of nodes plus a node for each remaining row. This protocol can achieve high read capacity, low quorum size, and other desirable features for replicated data management. For diamond quorum consensus optimal read quorum size is 2 and is independent of the total number of sites. Worst case read quorum size is $\lceil\sqrt{(2N)}\rceil$. Optimal and worst quorum sizes for diamond write quorum are $\lceil\sqrt{(2N)}\rceil$ and $2\lceil\sqrt{(2N)}\rceil - 2$ respectively.

3 Wheel Structure Model

In this section we propose a new protocol for replica control. We name this protocol as The Wheel Protocol and describe its model here.

A distributed system consists of a set of distinct sites that communicate with each other by sending messages over a communication network. No assumptions are made regarding the speed, connectivity, or reliability of the network. We assume that sites are fail-stop [15] and communication links may fail to deliver messages.

Data is replicated by storing copies of the same logical data item at different nodes. Two operations, read and write, are allowed on replicated data. Before performing the operation a node must obtain permission from a number of copies (quorum) using a control protocol. The correctness criteria for replicated databases is one-copy serializability [2], which ensures one-copy equivalence and serializable execution of transactions. In order to ensure one-copy equivalence, a replicated object z may be read by reading a read quorum of copies, and it may be written by writing a write quorum of copies. The following restriction is placed on the choice of quorum assignments:

Quorum Intersection Property: For any two operations O[Z] and ó[z] on an data item x, where at least one of them is a write, the quorums must have a nonempty intersection.

Version numbers or timestamps are used to identify the current copy in a quorum. Each node is logically characterized by attributes like ID, Node Location, HUB, SUC, PRED . **ID** which is a unique sequential ID. In our discussion, IDs are numbered as 1, 2,

3. **Node Location** is the location where the node is physically residing. In other words this is the address of a node in the network. **HUB** contains the ID of the node in the wheel which is currently acting as hub. In our discussion, ID of the hub node is ZERO. **SUC** contains the ID of the successor w_{i+1} , which is the next node in the wheel. **PRED** contains the ID of the predecessor w_{i-1} , which is the previous node in the wheel.

ID	Node location	HUB	SUC	PRED

Let $W_n = w_1 , w_2 , w_3 ,...,w_n$ be the set of nodes that store copies of a replicated data item. **A wheel structure**, W_n is a logical structure with n nodes, formed by connecting a single node called HUB to all vertices of an (n-1) cycle. Vertices in the cycle are called as spokes. The numerical notation for wheels is used inconsistently in the literature: some authors instead use n to refer to the length of the cycle, so their W_n is the graph we would denote as W_{n+1} . All nodes in the cycle maintain adjacency relationship by maintaining ID's of their successor and predecessor. Each node is defined by attributes ID, Node Location, HUB, Suc, and Pred. Wheel structure is easily imposed on the set of nodes by selecting first node as HUB and adding other nodes as spokes in cycle by defining the successor(Suc(i))
, predecessor (Pred(i)) operations and by setting hub in each spoke.

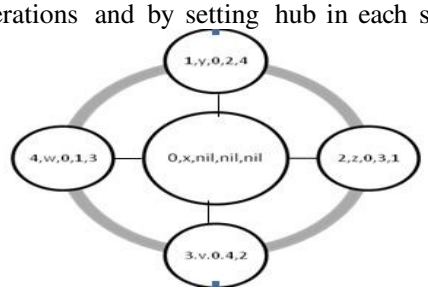

Fig. 1. Wheel Structure

Informally, read quorum can be obtained by reading only hub and write quorum by reading HUB plus alternate spokes . Read quorum size of one is the minimum among all other proposed protocols. In case of failure of hub, some reconfiguration algorithm can be used to elect a new hub. Thus making hub always available and keeping read quorum size minimum. Even in case of no reconfiguration, read quorum can be obtained by reading any two adjacent spokes in the wheel, which is also a smaller read

quorum size. In future, we plan to do detailed algorithmic evaluation of Wheel structure. Wheel structure looks very promising specially for the applications where number of read operations are more than write operations.All discussed protocols help to achieve high availability, but some of them have restrictions on N, the number of nodes in the system. Our proposed wheel structure has no such restriction and any number of nodes can be arranged in a wheel structure

An outstanding feature of Wheel quorum is its minimum read quorum size of one in failure free environment. If reconfiguration can be done to make hub always available then, this smallest read quorum size can be maintained. Even without it, read quorum size will be two with is obtained by reading adjacent
spokes. Write quorum size is $\lceil((n-1)/2\rceil + 1$.

4 Conclusion

In this paper, we propose implementing replica control protocols using different logical structures. In the original quorum protocol [7] , the sizes of both read and write quorum may vary depending on the relative availability requirements of read and write operations. In the logical structure protocols discussed in this paper the best case sizes of the quorums are less then the quorum protocol. A new logical wheel structural arrangement is also suggested. Its model and quorum construction method is defined. Wheel structure promises to give smaller read and better write quorum size.In particular, our protocol performs well in systems where read operations are requested more frequently than write ones.

References

1. D. Agrawal and A. El Abbadi. An Efficient. An efficient solution to the distributed mutual exclusion problem. In Proceedings of the Eighth ACM Symposium on Principles of Distributed Computing,, pages pages 193–200,, August 1989.

2. P. A. Bernstein and N. Goodman. A proof technique for concurrency control and recovery algorithms for replicated databases. Distributed Computing, Springer- Verlag, 2(1):32-44, January 1987.

3. Yao-Jen Chang. A triangular-mesh-based approach to fault-tolerant distributed mutual exclusion. Master's thesis, National Sun Yat-sen University, June, 1995.

4. Amr E. Abbadi Divyakant Agrawal. The tree quorum protocol: An efficient approach for managing replicated datain. Proceedings of the 16th International Conference on Very Large Data Bases (1990),, pages pp. 243– 254., 90:.

5. Ada Wai-Chee Fu, Yat Sheung Wong, and Man Hon Wong. Diamond quorum consensus for high capacity and efficiency in a replicated database system. Distrib. Parallel Databases, 8:471–492, October 2000.

6. Hector Garcia-Molina and Daniel Barbara. How to assign votes in a distributed system. J. ACM, 32:841–860, October 1985.

7. H. Gifford. Weighted voting for replicated data. in Proceedings of 7th Symposium on operating Systems, ,ACM, pages pp 150–162, 1979.

8. Jim Gray, Pat Helland, Patrick O'Neil, and Dennis Shasha. The dangers of replication and a solution. In SIGMOD '96: Proceedings of the 1996 ACM SIGMOD international conference on Management of data, volume 25, pages 173–182, New York, NY, USA, June 1996. ACM.

9. A. Kumar. Hierarchical quorum consensus: a new algorithm for managing replicated data. Computers, IEEE Transactions on, 40(9):996–1004, 1991.

10. M. L. Liu, D. Agrawal, and El A. Abbadi. Abbadi. On the implementation of the quorum concensus protocol. In In Proc. Parallel and Distributed Computing Systems, 1995.

11. Mamoru Maekawa. A sqrt(N) algorithm for mutual exclusion in decentralized systems. ACM Trans. Comput. Syst., 3(2):145–159, May 1985.

12. Nabor C. Mendona and Ricardo O. Anido. The hierarchical ring protocol: An efficient scheme for reading replicated data. Technical Report DCC-93-02, Department of Computer Science, University of Campinas, February 1993. In English, 30 pages.

13. David Peleg and Avishai Wool. Crumbling walls: a class of practical and efficient quorum systems. In Proceedings of the fourteenth annual ACM symposium on Principles of distributed computing, PODC '95, pages 120–129, New York, NY, USA, 1995. ACM.

14. M. H. Ammar S. Y. Cheung and M. Ahamad. The grid protocol: A high performance scheme for maintaining replicated data. IEEE Transactions on Knowledge and Data Engineering, Vol. 4, No. 6:pp. 582–592, Dec. 1992.

15. Richard D. Schlichting and Fred B. Schneider. Fail-Stop Processors: An Approach to Designing Fault-Tolerant Computing Systems. Computer Systems, 1(3):222–238, 1983.

16. Robert H. Thomas. A Majority consensus approach to concurrency control for multiple copy databases. ACM Trans. Database Syst., 4(2):180–209, June 1979.

17. Y.C.Kuo and S.T. Huang. A geometric approach for constructing coteries and k-coteries. IEEE Transaction Parallel and Distributed Systems, 8(4):402 411, April 1997.