Parent Siblings Oriented Tree Quorum Protocol

Anurag Singh, Ashish Kumar Rai, Anup Kumar Jayswal and Meenu

Department of Computer Science and Engineering
Madan Mohan Malaviya Engineering College
Gorakhpur, India

***Abstract*- In this paper, we are proposing a new replica control algorithm Parent Siblings Oriented Tree Quorum Protocol (PSTQ) for the management of replicated data in distributed database system. This algorithm imposes a logical structure of tree on the set of copies of an object. The proposed protocol provides a small read quorum as well as a small write quorum while guaranteeing fault-tolerance of write operations. With this algorithm read operation is executed by reading one copy in failure-free environment. In case of failure of sites, number of data copies required for read operation increases but remains constant for subsequent failure of the sites. The less number of data copies required for write operation provide low write operation cost and high write availability.**

I. INTRODUCTION

Replication is the technique of maintaining multiple copies of the data items at different sites. Replication increases the data availability. It means that we can access the data from any accessible site. It provides fault-tolerance so that after failure of some sites, transaction can continue using different copy of data item. By imposing logical tree structure on the set of data copies there is no need of the reconfiguration. So failure and subsequent recovery of the sites do not cause any reconfiguration.

In replication multiple copies of a data is stored at different sites. These multiple copies of a data item must appear as a single logical data copy to the transactions. This is called as one copy equivalence [4]. The replica control protocol ensures this equivalence.

The quorum is the set of the minimum number of data copies required for the successful execution of an operation. So, to execute a read and write operation, read and write quorum must be constructed respectively. The quorums must be constructed in such a way that quorums follow the quorum intersection property. Quorum intersection property states that for any two operations op1(x) and op2(x) on a data item x, where at least one of them is write, the quorums must have a non-empty intersection. We note that tree structure is logical and does not have to correspond to actual physical structure of the network connecting the sites storing the copies.

II. RELATED WORK

There are various existing protocols for managing replicated data. In read one write all (ROWA) protocol read operation reads any one copy whereas all data copies are require for write operation. In ROWA, read cost is small and write cost is very high. Write operation cannot tolerate failure of any site in ROWA.

In order to increase the fault tolerance of write operations in ROWA, voting protocol is proposed where write operations are not required to write all copies. Voting approach [3] is proposed to increase the fault tolerance of ROWA. Majority of votes of sites are required to make quorums. In this protocol, write operation need not to write all copies but the read operation reads several copies that increases the read cost. There are two versions of voting protocol namely static and dynamic voting protocol. In static protocol the size of quorums are predefined and fixed whereas in static voting the size of quorum vary according to the situation.

To overcome the problem of expensive red operation in voting protocols, several protocols has been proposed that use the network configuration information. As a result, read operation requires only a single copy.

The tree quorum protocol tries to achieve the advantages of reconfiguration protocol i.e., that is low cost operation execution while maintaining availability. In tree quorum protocol [1], a logical tree structure is imposed on data copies. A read operation reads a single copy like ROWA in the failure free environment. Read operation require more copies in case of failure. Write operation tolerates failure and no reconfiguration protocol is used. Here, a write operation is required to write a majority of copies at all levels of the tree. Read operation can be executed by reading a majority of copies at any single level of the tree.

III. PARENT SIBLINGS ORIENTED TREE QUORUM PROTOCOL (PSTQ)

In this Section, we present a new protocol for the management of replicated data item in distributed database system. We assume that the tree has a well defined root. In this approach, quorums are constructed by using relationship of parent and siblings with node in logical tree structure of data copies. For each node, we have defined node-parent-siblings (NPS) group. NPS group for a node consists the node, its parent and all of its siblings. We are describing a protocol that works by reading one copy of an object while guaranteeing fault-tolerance of write operations and still does not require any reconfiguration in case of a failure and subsequent recovery. This protocol provides a comparable degree of data availability too. Fig. 1 shows a tree of degree 3 and of height 3 having 13 nodes.

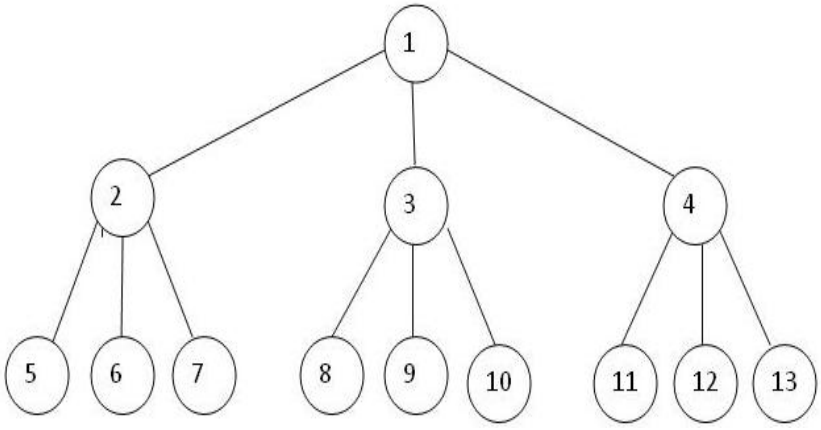

Figure 1. Tree Structure Imposed on Data Copies

A. Construction of Read and Write Quorum

For a read operation, the recursive function ReadQuorum is called with the root of tree as parameter. A read quorum can be formed by all the nodes of NPS group of any node. NPS group corresponding to root consist only root node because root does not has parent or siblings. If the root node is accessible, a read quorum can be formed by root node only. Fig. 2 and Fig.3 shows the algorithms for read and write quorum construction.

A transaction attempting to construct a write quorum calls the recursive function WriteQuorum with the root of the tree as parameter. By taking exactly one node from NPS group of each node, read quorum may be formed. Since NPS group of root has only root node, write quorums must have the root. We start from root and move downwards recursively.

```
Function ReadQuorum(trees): QUORUM;
var NPSQuorum, children: QUORUM;
var node: NODE;
begin
  node= root of the tree;
   if Empty(trees)  then
      return ({ });
   else if  all nodes of NPS(node) are accessible  then
      return (each node of NPS(node));
    else
      begin
         children=  children of node;
             for each node ε children
    NPSQuorum = ReadQuorum(child Subtrees);
   if all nodes of NPS(node) are inaccessible then
               return ({ });
            else
      return (NPSQuorum);
         end
end;
```

Figure 2. Algorithm for Read Quorum Construction

```
Function WriteQuorum(Trees): QUORUM;
var NPSQuorum,  children: QUORUM;
var node: NODE
begin
node= root of the tree;
if Empty(Trees)  then
       return  ({ });
else if any node of NPS(node) is already taken then
        for each child  ε children
NPSQuorum  = NPS Quorum  U
                  WriteQuorum (child subtree);
else  if  each node in NPS(node)  is  accessible  then
      begin
           NPSQuorum =  NPSQuorum U
                    (any node among node and its siblings);
            children=  children of node;
           for each child ε children
           NPSQuorum  = NPSQuorum  U
                    WriteQuorum (child subtree);
           if unable to take any node from NPS(node)
                    return ({ });
           else
                    return  (NPSQuorum);
      end
else
           return  ({ });
end;
```

Figure 3. Algorithm for Write Quorum Construction

Before selecting one node from a NPS group, check all the nodes. If any node is already taken by our algorithm, there is no need to take a node from that NPS group. Otherwise, take only one node among current node and its siblings from NPS group.

B. An Example

For the tree in figure 1, a read quorum can be formed by {1} (NPS group of root) in the best case. If the root node is not accessible, other possible read quorums may be {2,5,6,7}, {3,8,9,10} or {4,11,12,13} which is NPS group of node 5, 8 and 11 respectively.

Write quorum is formed by taking at least one node from each distinct NPS group. For the given tree, NPS(1) consist only node1. Write quorum take node1. Since for the NPS(2), NPS(3), NPS(4) node 1 is already taken, there is no need to take any node. For NPS(5), we take any node out of (5,6,7). Again, there is no need of consideration of NPS(6) and NPS(7). Similarly for NPS(8) and NPS(11) take one node out of (8,9,10) and (11,12,13) respectively. So, some possible write quorums are {1,5,8,11}, {1,6,10,13},{1,7,9,12} etc. We can see that there is always a non-empty intersection between read and write quorum of the tree given in the figure 1.

C. Correctness of the Parent Siblings Oriented Tree Quorum Algorithm

The following theorem establishes the correctness of the Parent-Sibling oriented tree quorum protocol. We demonstrate that the read and write quorums constructed by the proposed algorithm will always have a non-empty intersection.

Theorem- The PSTQ ensures the intersection of the read and write quorums.

Proof- The proof is by induction on the height of the tree.

Basis-The theorem holds for a tree of height zero since there is only one copy in the tree .So read quorum and write quorum both will contain this copy. So both quorums have a non–empty intersection.

Induction Hypothesis- Assume that the theorem holds for the tree of height h.

Induction Step- Consider a tree of height h+1. The read and write quorums constructed for the tree has the following form:-

1. Read Quorum: {root} OR {all nodes of a NPS group}

2. Write Quorum :{ root} and {one node of all distinct NPS groups}.

If read quorum contain the root of the tree, it is sure to have a non empty intersection with any write quorum because in any write quorum presence of the root is must. On other hand, if read quorum consist all nodes of any one NPS group the nodes of read quorum will be common with at least one node of the write quorum of the sub tree of height h. Since the sub trees are of the height h, the induction hypothesis guarantees that the read and write quorums will have a non-empty interaction.

IV. PERFORMANCE ANALYSIS AND COMPARISON

In this section, we estimate the message cost and the availability of read and write operations in the proposed parent siblings oriented tree quorum protocol and compare them with the read-one write-all (ROWA), voting (VOTE) protocols and the tree protocol. Availability analysis is done to show that the availability of read and write operations is not degraded in our protocol.

A. Message Cost Analysis

The message cost of an operation is directly proportional to the quorum size required to execute the operation. Therefore, we represent the message cost in terms of the quorum size of the operation. In the read-one write-all approach, read operations costs one whereas write operations costs n (n is the total

number of data copies). In the voting protocol, the quorum size consists a majority (n + 1)/2 votes. Thus, read and write operations have a cost of (n + 1)/2 [6]. In the case of the tree quorum protocol (TREE), the size of read quorums vary from 1 to M^h. On the other hand, the cost of write operations is $\sum_{i=0}^{h} M^i$. M is equal to (d+1)/2.

Suppose tree is of degree d and of height h. In our protocol, size of read quorum is one or d+1 where d represents the degree of tree. If root is accessible, read quorum is one. Otherwise read quorum size is d+1 (all nodes of a NPS group). In spite of failure of nodes, maximum read quorum size remains d+1. On the other hand, the write quorum size is $(d^{h+1}-1)/(d^2-1)$ when h is odd. In case of even value of h, write quorum size is $1+[(d^{h+1}-d)/(d^2-1)]$.

In figure 4 and 5, we compare the read cost and write cost of our protocol with three previously mentioned protocols. In figure 4, we plot the maximum read cost of each protocol to compare them in worst case. We took entries for 4, 13 and 40 total number of data copies.

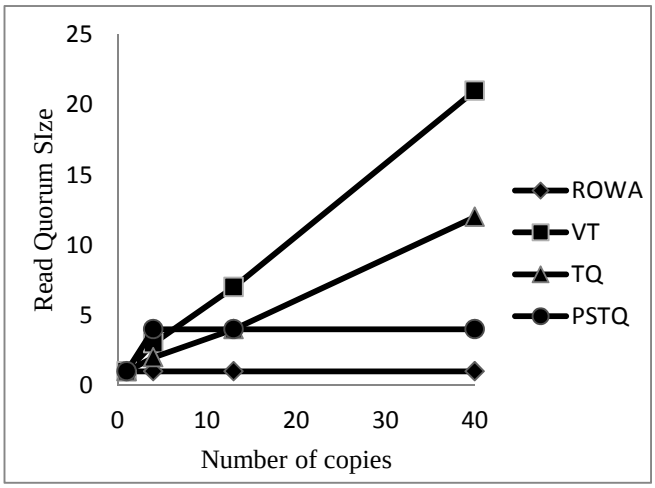

Figure 4. Comparison of the Read Cost

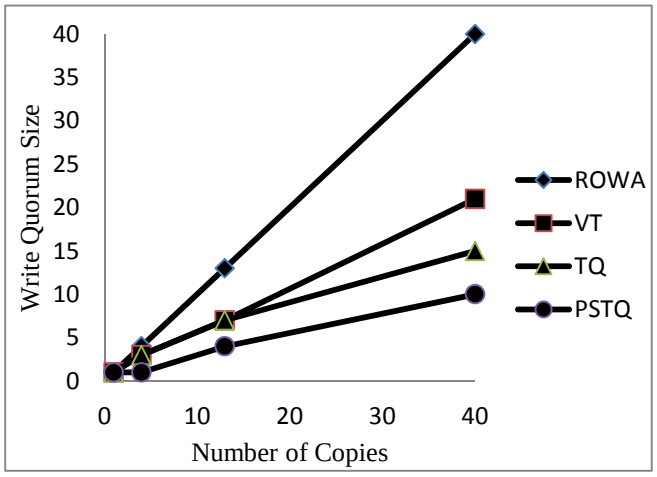

Figure 5. Comparison of the Write Cost

For ROWA, read cost is just one. In voting approach, read quorum consists the majority of data copies. In the best case, both TQ and PSTQ are equivalent to ROWA. But in worst case, TQ requires M^h node whereas PSTQ requires just d+1 nodes.

In figure 5 for write operation, ROWA has highest cost. Write cost of voting protocols is same as read cost of it. In TQ, write cost is less than (n/2) where n is the total number of nodes. For the proposed protocol PSTQ, write cost is less than that of tree quorum protocol which can be seen in the figure 5. In PSTQ, write quorum consist just one node from each node-parent-siblings (NPS) group.

B. Availability Analysis

In this section, PSTQ is compared with other protocols on the basis of operation availability. Operation availability is the probability of making the required quorum for that operation. It is assumed that all copies of data are of equal availability p. Availabilty of read and write operations for ROWA, voting protocol and tree protocol can be calculated in terms of p [5].

The availability of the read and the write operations for PSTQ can be derived by using recurrence equations based on the tree height h. Let R_i and W_i be the availability of the read and the write operations with a tree of height i and of degree d. The availability of read and write operation for a tree of height i can be derived as:

$$R_i = \begin{cases} p+(1-p)[1-(1-R_{i-1})^d] & \text{when } i=h \\ p^{d+1}+(1-p^{d+1})[1-(1-R_{i-1})^d] & \text{when } i<h \end{cases}$$

where $R_0=p^{d+1}$

$$W_i = \begin{cases} p(W_{i-2})^{d*d} & \text{when } i=h \\ [1-(1-p)^d][W_{i-1}]^d & \text{when } i<h \end{cases}$$

where $W_0=1-(1-p)^d$

In the figure 6 and figure 7, availability analysis of operations for PSTQ is compared with three other protocols namely ROWA VT and TQ.

We assume that all data copies have the same availability. Here we take total number of copies(n) =40. In the figure 6, we can see that the availability of read for PSTQ is higher than VT and comparable to TQ. ROWA has very good availability for read operation. In case of write operation availability, PSTQ is better than ROWA and VT. PSTQ gives better write availability than TQ when p>0.6.

From the point of view of operation cost, PSTQ is better than ROWA, VT and TQ in the most cases. Considering write operation availabilities, PSTQ is comparable with TQ and better than ROWA and VT.

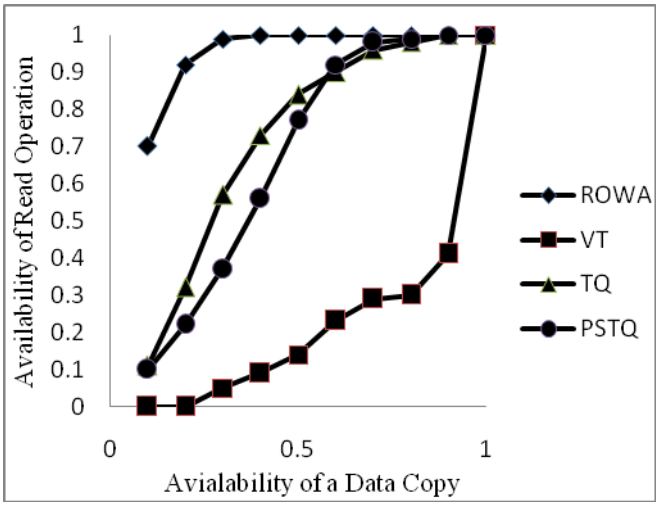

Figure 6. Comparison of the Read Availabilty

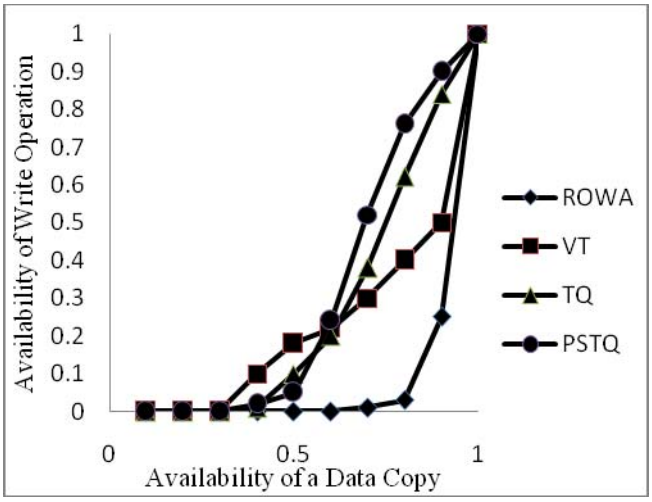

Figure 7. Comparison of the Write Availabilty

V. CONCLUSIONS

In this paper, we have proposed Parent Siblings Oriented Tree Quorum Protocol for the management of replicated data in distributed systems. A logical tree structure is imposed by this protocol to increase operation availability and decrease operation cost. This protocol focuses on the parent and siblings relationships with the node. With PSTQ a read operation can be carried out by only a single root copy. Read quorum size does not increase with the increment in the number of site failures. Write operation requires one copy from NPS group corresponding to each node in the tree. So, the write operation cost of PSTQ is lower than all the three mentioned protocols. In both PSTQ and TQ, root node must be included in write quorum. So, the root node acts as a bottleneck for write operations. There is no need of reconfiguration for PSTQ in case of site failure and subsequent recovery. The logical structure of tree wiil be particularly beneficial if it is organized such that most reliable site is chosen as the root and the least reliable sites as the leaves. In this situation, the PSTQ gives very good performance in failure free environment as well as in failure environment.

VI. REFERENCES

[1] Agrawal and A. El Abbadi, "The tree quorum protocol: An efficient approach for managing replicated data," Roc. VLDB, 1990, pp.243-254.

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

[3] K.Gifford, "Weighted voting for replicated data," hoc. Symp. on Operating System Principles, 1979, pp. 150-162.

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

[5] Soon M.Chung and Cailin Cao, "Multiple tree quorum algorithm for replica control in Distributed Database System" , IEEE, 1992, pp.282-286

[6] R. H. Thomas, "A majority consensus approach to concurrency control for multiple copy databases," ACM Trans. on Database Systems, Vol. 4, No. 2, 1979, pp.180-209.