International Journal of Engineering Science Invention Research & Development; Vol. I Issue III September 2014
www.ijesird.com e-ISSN: 2349-6185

Cost Reduction of Replicated Data in Distributed Database System

[1]Divya Bhaskar, [2]Meenu
Department of computer science and engineering
Madan Mohan Malviya University of Technology
Gorakhpur 273010, India
divyabh70@gmail.com, Myself_meenu@yahoo.co.in

***Abstract*—In this paper, we are proposing a new replica control algorithm NCP (node child protocol) for the management of replicated data in distributed database system. The algorithms impose logical tree structure on a set of copies of an object. The proposed protocols reduce the quorum size of the read and write quorum. With this algorithm read operation is executed by reading one copy in a failure free environment and if the failure occur then also it read single copy .The less number data copies required for the write operation and it provide low write operation cost then the other protocols.**

Keywords—Replication, message cost, quorum protocol,

I. INTRODUCTION

A distributed database system is consist of a set of computers (called site) which is distributed in several location. In a distributed system processors are loosely coupled site that share no physical components.

Replication is a useful technique for distributed database system where reliability is important, such as banking system, airline reservation system etc. replication is process to maintain multiple copies of the data at different site. In a distributed system data is replicated to achieve fault-toltrence . One of the most important advantage of replication is that it mask and tolerate failure in a network gracefully. If failure occur even then system remain operational and available to the user. However there are complex and expensive algorithms are built to maintain the replicas. And also these algorithm help to reduce the cost of executing operation and also maintaining the availability of the replicated data .

In a replicated database, copies of an object may be stored at several sites in the network. Multiple copies may appear as a single logical object to the transaction. This is termed as one-copy equivalence and is enforced by the replica control protocol. The correctness criteria for replicated database is one-copy serializability , which ensure one-copy equivalence and serializability execution of transaction.

The replication can increase the availability of data item in a distributed database system it means if one of the site fail then transaction may continue by accessing data from another site also. In this way it achieve fault-tolerence.

A. Type of replication

The replication tools may be selected based on type of replication it supports. The capabilities and performance characteristics varies from one type of replication to another. A replication strategy may be selected based on two basic characteristics: *Where* and *When*.

When the data is updated at one site, the updates have to be propagated to the respective replicas. When the updates can be propagated can be achieved by Synchronous (eager) and Asynchronous (lazy) methods and where the updates can take place can be achieved by update everywhere and primary copy (master-slave) methods.

Synchronous replication (Master-Slave replication) works on the principle of Two-Phase commit protocol. In a two-phase commit protocol, when an update to the master database is requested, the master system connects to all other systems (slave databases), locks those databases at the record level and then updates them simultaneously. If one of the slaves is not available, the data may not be updated. The consistency of data is preserved; however it requires availability of all sites at the time of propagation of updates.

There exists two variations of *Asynchronous replication* (Store and Forward replication) i.e. Periodic and Aperiodic. In Periodic replication, the updates to data items are done at specific intervals and in aperiodic replication the updates are propagated only when necessary (usually based on firing of event in a trigger). The time at which the copies are inconsistent is an adjustable parameter which is application dependent In U*pdate anywhere* method, the update propagation can be initiated by any of the sites. All sites are allowed to update the copy of the datum whereas in a *Primary Copy* method there is only one copy (primary copy or master) which can be updated and all other (secondary or slave) copies are updated reflecting the changes to the master. Various forms of replication strategies are as follows:

***Snapshot Replication*:** In snapshot replication, a snapshot or copy of data is taken from one server and moved to another server or to another database on the same server. After the initial synchronization, snapshot replication can refresh data in published tables periodically. Though snapshot replication is easiest form of replication, it requires copying all data items each time a table is refreshed.

***Transactional Replication*:** In transactional replication, the replication agent monitors the server for changes to the database and transmits those changes to the other backup

servers . This transmission can take place immediately or on periodic basis. Transactional Replication is used for server-server scenarios.

***Merge Replication*:** Merge replication allows the replicas to work independently . Both entities can work offline. When they are connected, the merge replication agent checks for changes on both sets of data and modifies each database accordingly. If transaction conflict occurs, it uses a predefined conflict resolution algorithm to achieve consistency. Merge replication is used mostly in wireless environments.

***Statement based replication*:** The statement based replication intercepts every SQL query and sends it to different replicas. Each replica (server) operates independently. To resolve conflicts , Read-Write queries are sent to all servers where as read only queries can be sent to only one server. This enables the read workload to be distributed. Statement based replication is applicable for optimistic approaches where each cache maintains the same replica.

II. RELATED WORK

A. Read one write all (ROWA):

The simplest protocol for management of replicated data in distributed database system is read one write all (ROWA), in this read operation are allowed to read one copy in a failure free environment, and in write operation it allow to write all copy of the data item. The read one write all protocol provides read operation with a high degree of availability at very low cost: a read operation accesses a single copy. On the other hand, this protocol severely restricts the availability of write operation since they cannot be executed after the failure of the copy. Thus if there are N copies in the system then the cost of read operation is 1 and the cost of write operation is N in a failure-free environment, this protocol restrict the availability of write operation since they cannot be executed after the failure of any copy.

MESSAGE COST ANALYSIS: In ROWA, a read operation read only one copy and a write operation is required to write all the copies of a data item. Thus if there are N copies in the system, $C_{rowar} = 1$ and $C_{rowaw} = N$.

B Voting protocol:

In a this protocol every replicated file assigned some number of votes. Each read operation collects a read quorum of r votes to read a file, and write quorum of w votes to write a file, such that r + w is greater than the total number of votes assigned to the file. This ensure that there is a non-null intersection between every read and write quorum. An appropriate r , w are choose to control reliability and performance characteristic of a replicated file.

MESSAGE COST ANALYSIS: In voting, if the majority consensus method is used, the quorum size for both read and write operation can be the same, i.e., the majority of the total number of votes assigned to copies. Thus the cost C_{VTR} and C_{VRW} are the same as [(N + 1) /2] , where N is the total number of votes assigned to copies.

C Tree Quorum Protocol:

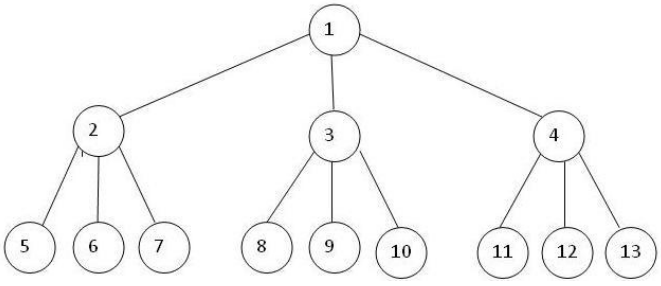
In tree quorum protocol, replica control protocol is used to reduce the cost of execution of the read and write operation without the need for reconfiguration. This is achieve by imposing logical tree structure on the set of copies of each object. We describe a protocol that operates by reading one copy of an object while guaranteeing fault tolerance of write operations and still does not require any reconfiguration on account of a failure and subsequent recovery. The protocol provides a comparable degree of data availability as other replica control protocols at substantially lower costs. Furthermore our approach is fault-tolerant, and exhibits the property of graceful degradation . In a failure free environment, the communication costs are minimal and as failures occur the cost of replica control may increase. However, when failures are repaired the protocol reverts to its original mode without undergoing any reconfiguration.

MESSAGE COST ANALYSIS: In TQ to estimate the cost of operation, it take h as a height of the tree, D is the degree of node in the tree, and M is the majority of D i.e., M=(D+1)/2. In tree quorum protocol, read quorum is 1 if root is accessible and if root fails then read quorum is the majority of its children, thus quorum size is M. so the range of read operation is from 1 to $(d+1)^h$:$1 \leq C_{rQW} \leq M_h$.

In TQ, all the write quorum have the same size: the root is selected and the majority of the children of each node selected are included recursively. Thus the cost of write operation C_{rQW} can be represented as $C_{rQW} = [(d+1)^{h+1}-1]/d$

III. PROPOSED WORK

In this Section, we present a new protocol for the management of replicated data item in distributed database system. In this protocol we impose the data items on a tree structure with a well defined root. The tree contain 13 nodes in it with degree 3 and height 3 In this approach, quorums are constructed by selecting node and its single child in logical tree structure of data copies.



A. *Construction of read quorum*

For a read quorum, the recursive function ReadQuorum is called with the root of the tree. The read quorum can be formed by selecting root node if root node is accessible if the

root node is not accessible then the any single child of the root is selected and the child is also not accessible then the any one child of the selected child is selected. It 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.

B. *Construction of write quorum*

For a write quorum, the recursive function WriteQuorum is called with the root of the tree. The write quorum can be constructed by selecting root and any single child of the root and any single child of the selected child and the same process continues depending on the height of the tree.

Figure 2. Algorithm for read and write quorum

```
FUNCTION ReadQuorum(Tree : TREE) : QUORUM;
VAR
MajorityQuorum, Majority: QUORUM;
BEGIN
IF Empty(Tree) THEN
RETURN({});
ELSE IF Tree T Boot is read accessible THEN
RETURN(TTree T .Root);
ELSE
(* Collect majority of subtrees to substitute for the root of the subtree *)
MajorityQuorum = UiEMajority ReadQuorum(Tsee T .SubTree[zj);s
IF Unable to collect a majority THEN
RETURN({));
ELSE
RETURN(MajorityQuorum);
END; (* IF l )
END; (* IF ‘)
END ReadQuorum
```

```
FUNCTION WriteQuorum(Tree : TREE) : QUORUM:
VAR
SubTrees, Majority: QUORUM;
BEGIN
IF Empty(Tree) THEN
RETURN({));
ELSE IF Tree 1 Boot is write accessible THEN
(* Collect majority of subtrees to be included with the root of the subtree *)
SubTree = sU I Majority WriteQuorum(Tree 1 .SubTree[i]);
IF Unable to collect a majority THEN
RETURN({));
ELSE
RETURN(Tree 1 .Root u SubTrees);
END; (* IF l )
ELSE
RETURN({});
END; (’ IF l )
END WriteQuorum;
```

C. *An Example of read and write quorum*

For the tree in figure 1, a read quorum can be formed by selecting root in the best case. If the root node is not accessible, then the any one child of the root is selected and if that child is also not accessible then the any one child of the selected child is taken , so the possible read quorums may be {1,2,5} or {1,3,8} or {1,3,10} etc.

Write quorum is formed by taking root and any single child of the root and any single child of the selected child. So some possible write quorums are {1,4,12} , {1,2,6}, {1,2,5} etc. We can see that there is always a non-empty intersection between read and write quorum of the given tree .

IV. PERFORMANCE ANALYSIS

In this section, we estimate the message cost and the availability of read and write operations in the proposed node child protocol (NCP) and compare them with the read-one write-all (ROWA) , voting (VOTE) protocols and the tree protocol.

In this protocol we try to improve the read and write cost to some extent with compare to the other existing protocol, in this protocol we mainly focus on the cost.

Availability analysis is done to show that the availability of read and write operations is not decrease in our protocol.

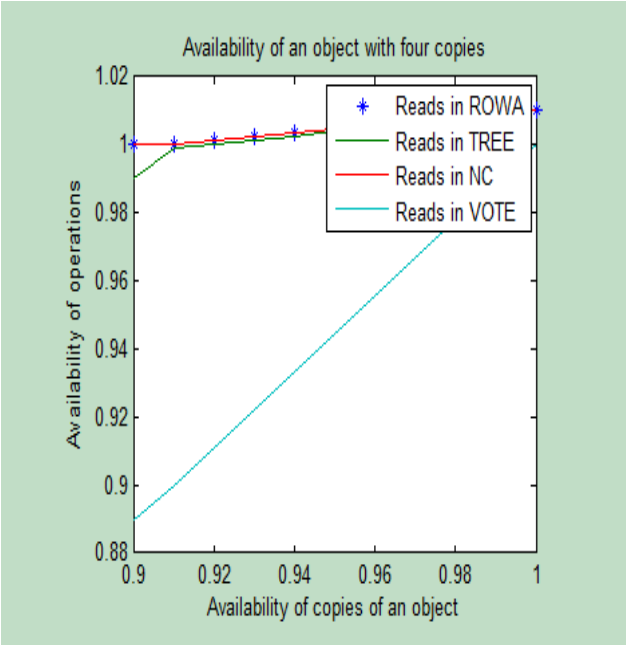


Fig. 1 Comparison of read Availability

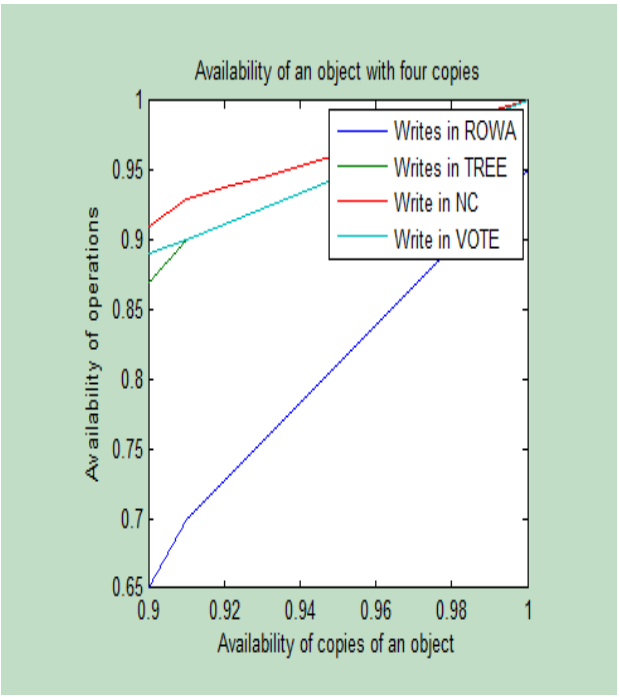


Figure 2. Comparison of write Availability

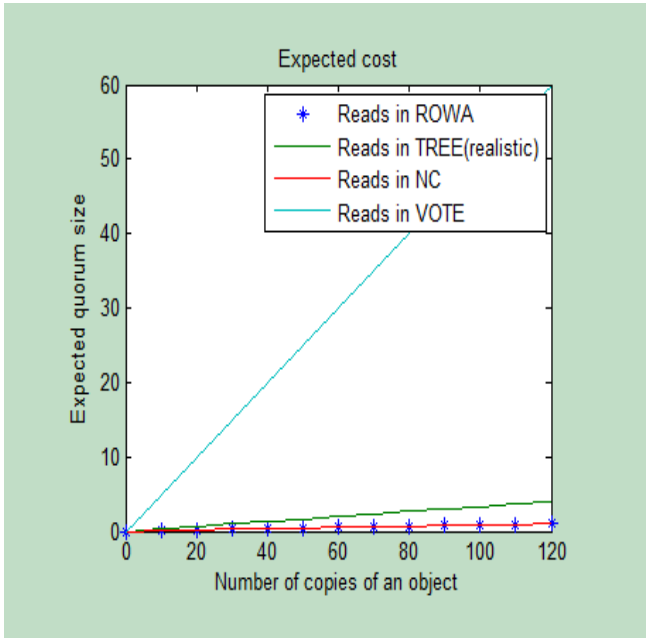


Fig. 3 Comparison of read cost

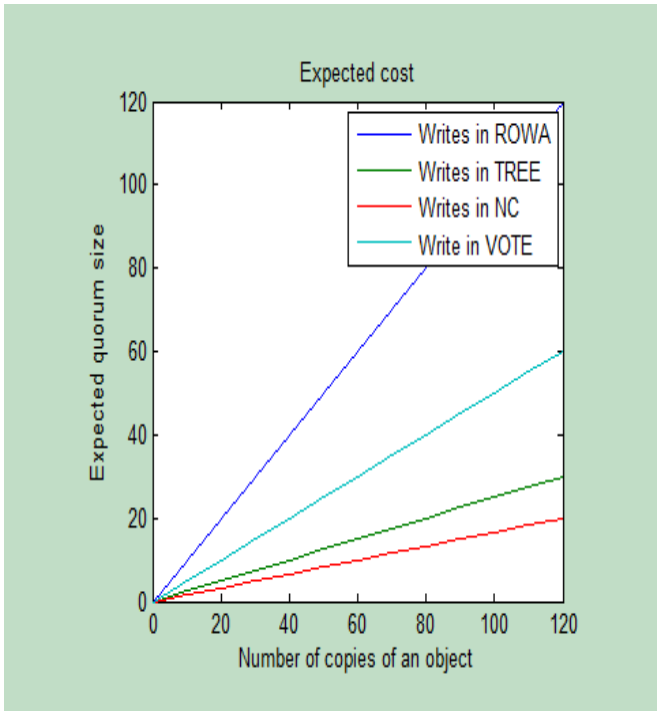


Figure 4. Comparison of write Cost

V. CONCLUSION

In this paper, we have proposed Node child protocol (NCP) for the management of replicated data in distributed systems. A logical tree structure is imposed on this protocol to decrease operation cost . With NCP 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 root and single child of the root. So, the write operation cost of NCP is lower than all the three mentioned protocols. In both NCP 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 NCP in case of site failure and subsequent recovery. The logical structure of tree will 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 NCP gives very good performance in failure free environment as well as in failure environment.

REFERENCES

[1] D. Agrawal and A. EI Abbani "*The Tree Quorum Protocol: An Efficient Approach for Managing Replicated data*", Proc VLDB 1990, pp.243-254.

[2] D.Agrawal and A.EI Abbani, "*Efficient Techniques for Replicated Data Management*," proc. Workshop on management of replicated data,1990 pp.48-52

[3] D.K. Gifford, "*Weighted Voting for Replicated Data*." Proc symp on operating system principles, 1979, pp.150-162

[4] R. H. Thomas, " *A Majority Consensus Approach to Concurrency control for Multiple copy Database*," ACM Trans. on database system, Vol.4, No.2. 1979, pp.180-209.

[5] M. Chung t and Cailing Cao "*Multiple Tree Quorum Algorithm for Replica in Distributed Database System*." 1992 IEEE

[6] Philip A. Bernstein and Nathan Goodman, "*An Algorithm for Concurrency Control and Recovery in Replicated Distributed Database*" ACM Transaction on Database System, Vol. 9 No. 4, December 1984.

[7] M. Ahamad and M.H. Ammar , "*Performance Characterization of Quorum-Consensus Algorithms for Replicated Data*." 1989 IEEE.

[8] D. Agrawal and A.E. Abbani, "*An Efficient solution to the distributed mutual exclusion problem*." ACM syms. On principle of Distributed Computing, pages 193-200, august 198 9.

[9] 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.