IJCSE International Journal of Computer Sciences and Engineering Open Access
Research Paper Vol.-6, Issue-11, Nov 2018 E-ISSN: 2347-2693

Database Transaction Processing with Effective Locking Mechanism for Consistency in Cloud Database

A. Aasha Begum[1*], K. Chitra[2]

[1]Department of Computer Science, Govt Arts College, Melur, Madurai, India
[2]Department of Computer Science, Govt Arts College, Melur, Madurai, India

[*]*Corresponding Author: a.aashabegum@gmail.com*

Available online at: www.ijcseonline.org

Accepted: 27/Nov/2018, Published: 30/Nov/2018

***Abstract*—** The cloud infrastructure provides Database Management System as a service with scalability and elasticity. Cloud DBMS is a less expensive platform for managing our data resources. Cloud data are replicated for high data availability. Concurrent transactions are done in the cloud database. Database users expect that data should be consistent when two users access to the same data at the same time and can see the same value. Due to replication of data in cloud environment, data inconsistency may occur between different replicated nodes. Hence Effective locking mechanisms are needed to handle such replicated cloud database. This paper proposes a novel algorithm for transaction processing to solve inconsistency and to control concurrency using lock managers and queue supervisor.

***Keywords*—** Transaction Processing, Consistency, Database Management System, Cloud computing.

I. INTRODUCTION

A Cloud DBMS is a distributed database that brings computing as a service instead of a product. It shares the resources, information and software between multiple devices over Internet. Nowadays, it is growing significantly. As a result, database management tasks are outsourced to third parties just like putting it into the cloud for much lower cost. The structure of cloud computing database and its functioning in collaboration with other nodes is observed under database as a service [1]. Many e-commerce companies are getting benefits from DB as a service. This paper proposes a novel algorithm for transaction processing to solve inconsistency and to control concurrency using lock managers [2].

The following sections are organized as follows, Section II discusses the related work Section III introduces the Transaction Processing for Consistency (TPC) Algorithm, Section IV evaluates the performance of the proposed algorithm and Section V concludes the work.

II. RELATED WORK

In this section, the previous research works are described. In cloud systems, three major questions of when to replicate, where to replicate, and what to replicate [5] rise. It must be handled in an economically feasible way [7]. [3] analyzes different cloud definitions suggested by different experts from different perspective. Some of them emphasize resource usage optimization and scalability as the vital elements of the cloud.

A transaction is a unit of work that is performed in a database [11]. Transactions are sequence of tasks done in an order. It is the propagation of one or more changes to the database. In order to ensure data integrity and to handle database errors, t is important to control transactions [12].

There are number of works in the literature that studies data replication in the cloud environment. Many of them focus on satisfying the availability [6]. In a cloud environment, frequent queries are placed on a large-scale data, the cloud clients expect low response time. However, performance guarantees are often not provided by cloud providers, e.g. response time. In order to provide response time guarantees, there are several works proposed [8] in the literature. For data replication, only a few studies are available to improve the response time [2] [4] [5]. In addition, very fewer studies are taking economics of the cloud into account [10].

III. TRANSACTION PROCESSING FOR CONSISTENCY (TPC) ALGORITHM

When more than one transaction tries to read or write the same data item, they are said to be concurrent transactions. Simultaneous execution of concurrent transactions will lead to inconsistent database [5]. Even in case of replicated database, more care must be taken to handle concurrent transactions. TPC algorithm is the solution to solve inconsistency. This algorithm is used to handle concurrency control in distributed database system with replicated database.

A. TPC Algorithm

In our proposed TPC system, there are clients, servers (nodes) and a Query Supervisor. Every node maintains lock-manager and replication list. The replication list is a tuple (A, Node), where A is the data item and Node is replicated site. Hence, all the nodes know where the data items are replicated. The lock-manager is responsible for lock and unlock processes during read or write transactions on the data items available in that particular site.

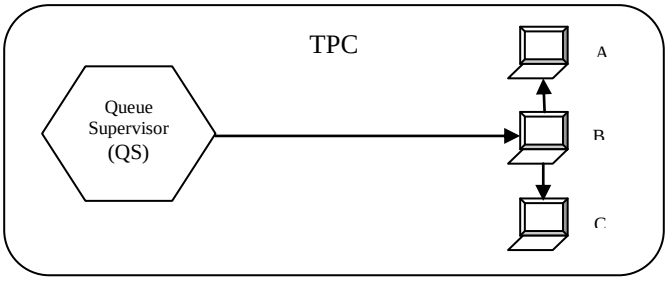

Figure 1. Structure of TPC

TPC consists of a Queue Supervisor (QS). QS maintains the Read List (R) and the Write list (W). When a client sends a request, the Queue Supervisor receives it and checks for the type of transaction. If it is a Read transaction, the Write List is checked. If there is an ongoing Write transaction on the same data item, the Read is made to wait in Read Transaction Wait Queue (RTWQ) till the Write transaction completes. If it is a Write transaction, the Read List is checked. If there is any Read Request on the same data item, the Read is made to wait in RTWQ. After the transaction is over, the read transaction from the RTWQ is processed. This waiting is done for maintaining consistency.

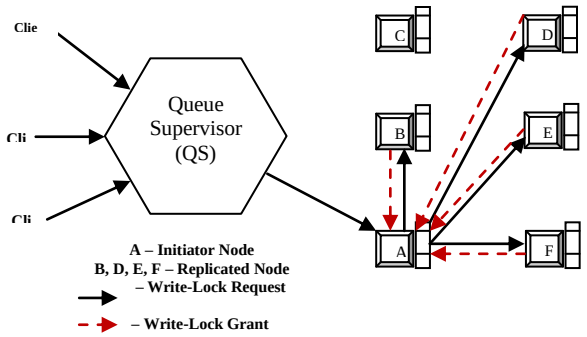

Figure 2. Write Transaction in TPC

The Lock Manager does its job as follows: If a data item L is replicated over n nodes, then a read lock request message must be sent to any one of the nodes and write lock request to all other nodes in which L is replicated. For example, the transaction is initiated at node A. The data item L is replicated in B, D, E, F. A sends read lock request message to B and write lock request to D, E, F. The read transaction continues with the node A. Once the transaction is over, the lock is released.

When a write transaction comes in, it reaches a node. The node checks the replicated list and then sends write lock request on L to all the L replicated nodes. Every replicated node must grant the write lock on L. After this grant, the transaction takes place at all the L replicated sites. If any one or more nodes cannot grant write lock, the write transaction will not be continued. For example, the transaction is initiated at node A. The data item L is replicated in B, D, E, F. A sends write lock request to B D, E, F. The write transaction occurs in these four nodes and then these nodes send acknowledgement message to A. Once the node A receives acknowledgement from all the four nodes, it sends COMMIT message to B, D, E, F. If at least one node did not send acknowledgement, the write will not be committed. The initiated node A waits until the threshold time then it send ROLLBACK message to all nodes B, D, E, F.

If the transaction is successfully committed, the write lock is released at all nodes B, D, E, F. The lock-managers of all the nodes in which the data item L is replicated are responsible for handling lock and unlock requests locally and individually.

```
Let the nodes be A, B, C, D, E, F
Replicate the data items over 'n' sites
Let the data item L be replicated in B, D, E, F
Client sends Read Request on L
Queue Supervisor checks for Read or Write
If (Read)
        Checks the Write List W
        If (L Є W)
        {
                Read L goes to RTWQ
         }
        Else
        {
        LABEL:
                QS sends RREQ to A
A sends R-Lock to B
                //Send W-Lock to A, B, C, D
                Read L
                Release R-Lock
        }
Else
{
        Checks the Read List R
        If (L Є R)
        {
                Read L goes to RTWQ
        }
        Send WREQ to A
        A sends W-Lock to B, D, E, F
        Write L at B, D, E, F
        Send ACK to A
        If (ACK < n)
        {
                ROLLBACK write in B, D, E, F
                Release W-Lock
        }
        Else
        {
                COMMIT
                Release W-Lock
                If (L Є RTWQ)
```



```
                    {
                        GO TO LABEL
                    }
                }
}
```

IV. PERFORMANCE EVALUATION

TPC algorithm is simulated using CloudSIM with varying number of replicated nodes 5, 10, 15, 20, 25 and the number of successful read transactions increases while increasing the number of replicated nodes.

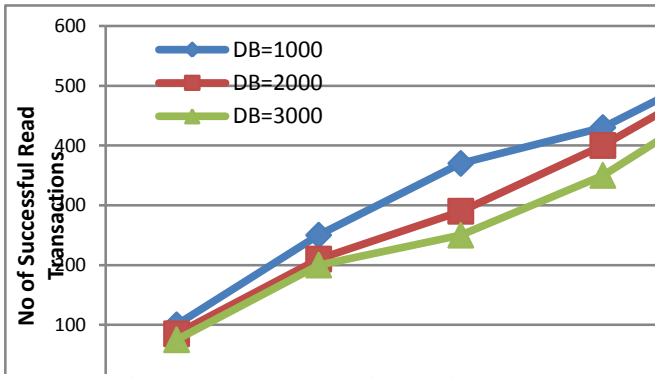

Figure 3. No of Replicated Nodes Vs No of Successful Read Transactions

It is also checked for the Write transactions. The number of successful write transactions increases while increasing the number of replicated nodes. But it is less than the number of successful write transactions. Hence compared to the read transaction the write transaction consumes some more time. and provides result as shown in the figure 4.

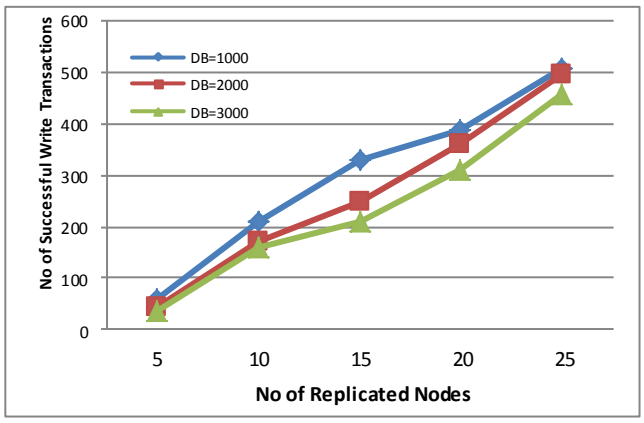

Figure 4. No of Successful Write Transactions Vs No of Replicated Nodes

Figure 5 shows the response time during write transactions while increasing the number of replicated nodes. The write involves the lock request, lock grant, update, acknowledgement and commit. It shows that the response time increases while increasing the number of replicated nodes because of the time involved in sending lock request, receiving lock grant, updating in all replicated nodes and receiving acknowledgement and committing the latest update.

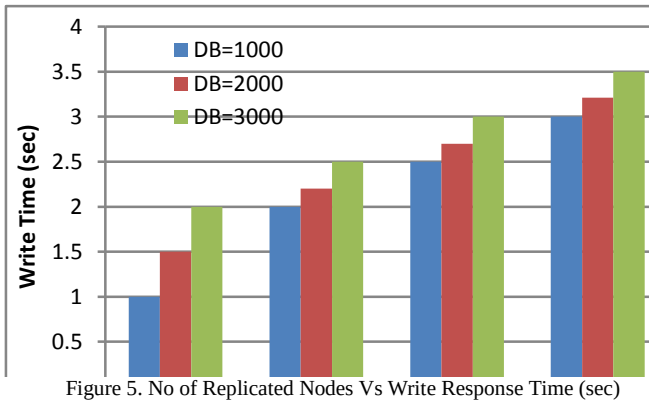

Figure 5. No of Replicated Nodes Vs Write Response Time (sec)

V. CONCLUSION

Database Management Systems as a cloud service are engineered to run as a scalable, elastic service available on a cloud infrastructure. CloudDBMSs will have an impact for vendors desiring a less expensive platform for development. In this paper, we presented the idea of DBMS in the cloud, the possibilities to be offered as one of the services offered by promising capability of cloud computing, that is to be a DBMS as a Service. In this paper we proposed TPC algorithm of DBMS in the cloud. In order to maintain database in the cloud, the concurrent transactions are managed to serve consistent data to the clients. These transactions are handled with effective locks. TPC is a novel algorithm for transaction processing to solve inconsistency and to control concurrency using lock managers.

REFERENCES

[1] D. Abadi, "Consistency tradeoffs in modern distributed database system design: Cap is only part of the story," Computer, Vol No. 2, Feb. 2012.

[2] D. J. Abadi, "Data management in the cloud: Limitations and opportunities," IEEE Data Eng. Bull., vol. 32, no. 1, pp. 3–12, 2009.

[3] P. S. Almeida, A. Shoker, and C. Baquero, "Efficient State-based Credits by Delta Mutation" in Proceedings of third International Conference Networked Systems (NETYS), 2015.

[4] P. Alvaro, P. Bailis, N. Conway, and J. M. Hellerstein, "Consistency without borders" in Proceedings of the 4th Annual Symposium on Cloud Computing, ser. SOCC'13. New York, NY, USA: ACM, 2013, pp. 23:1–23:10.

[5] R Anandhi, K Chitra, "A challenge in improving the consistency of transactions in cloud databases-scalability", International Journal of Computer Applications 52 (2), 12-14, 2012.

[6] E. Anderson, X. Li, M. A. Shah, J. Tucek, and J. J. Wylie, "What consistency does your key-value store actually provide?" in Proceedings of the Sixth international conference on Hot topics in system dependability, ser. HotDep'10. Berkeley, CA, USA: USENIX Association, 2010, pp. 1–16.

[7] B Jeevarani, K Chitra, "Improved consistency model in cloud computing databases", 2014 IEEE Computational Intelligence and Computing Research (ICCIC), 2014.

[8] O. Khan, R. C. Burns, J. S. Plank, W. Pierce, and C. Huang, "Rethinking erasure codes for cloud file systems: minimizing I/O for recovery and degraded reads," in Proceedings of the 10th USENIX conference on File and Storage Technologies, FAST, San Jose, CA, USA, February 14-17, 2012, p. 20.

[9] A. Khanafer, M. Kodialam, and K. Puttaswamy, "The constrained ski-rental problem and its application to online cloud cost optimization," in Proceedings of the IEEE INFOCOM, April 2013, pp. 1492–1500., 2014

[10] C. Li, D. Porto, A. Clement, J. Gehrke, N. Preguic¸a, and R. Rodrigues, "Making geo-replicated systems fast as possible, consistent when necessary," in Proceedings of the 10th USENIX Conference on Operating Systems Design and Implementation, ser. OSDI'12. Berkeley, CA, USA: USENIX Association, 2012, pp. 265–278.

[11] S.L. Mewada, "A Proposed New Approach for Cloud Environment using Cryptic Techniques", In the Proceedings of the 2016 International Conference on Computer Science and Engineering, India, pp.542-545, 2016.

[12] S. Tamilarasan, P.K. Sharma, "A Survey on Dynamic Resource Allocation in MIMO Heterogeneous Cognitive Radio Networks based on Priority Scheduling", International Journal of Computer Sciences and Engineering, Vol.5, No.1, pp.53-59, 2017.

Authors Profile

Dr.K.Chitra is having 17 years of academic experience and 14 years of research experience. She has published more than 100 research papers in reputed International Journals. She is guiding research scholars in the area of Cloud Computing and Big Data Analytics. She is ORACLE and IBM certified academician.

A.Aasha Begum is having 10 years of academic experience and 8 years of research experience.She has presented 3 papers in conference andpublished 2 papers in International journals.Her research area is Big Data Analytics.

