DOI: http://dx.doi.org/10.26483/ijarcs.v8i7.4460

ISSN No. 0976-5697

Volume 8, No. 7, July – August 2017

International Journal of Advanced Research in Computer Science

RESEARCH PAPER

Available Online at www.ijarcs.info

AN EFFICIENT DEPTH 1 FIXED TREE CONSISTENCY (D1FTC) METHOD FOR DISTRIBUTED DATA TRANSACTIONS IN CLOUD ENVIRONMENT

J.Antony John Prabu
Research Scholar and Assistant Professor
Department of Computer Science,
St. Joseph's College
Trichy, Tamilnadu, India- 620 002

Dr.S Britto Ramesh Kumar
Assistant Professor
Department of Computer Science,
St. Joseph's College,
Trichy, Tamilnadu, India- 620 002

Abstract: Cloud provides everything as a service in the IT market. Industries are started to move their services to cloud environment. Managing data transaction is one of the main issues in cloud and it needs efficient architecture to handle the transactional data. This paper discusses existing consistency methods, two-phase commit protocol and proposes a new consistency method to maintain ACID properties in cloud environment. It elaborates how the proposed Depth 1 Fixed Tree Consistency (D1FTC) method works in cloud environment. This work has done to maintain ACID and minimize the transactional issues in distributed cloud environment.

Keywords: Cloud DTM, 2-Phase Commit Protocol, Cloud Databases, Database Consistency

I. INTRODUCTION

The distributed data transaction system needs to maintain ACID guarantees to commit successful transactions and it is very hard in cloud environment [1]. Cloud is easy and good for analytical data but it has some complexity for transactional data like banking sectors, online reservation and shopping cart etc. Business people are very eager to switchover to cloud services to provide more reliability for their customers.

Cloud expects a better consistency method in transaction management [2, 24]. Multiple users can access databases and other services simultaneously from remote area. So cloud needs to ensure the reliability and assure the guarantee for every transaction committed successfully. Distributed two-phase commit protocol is suitable for cloud data transactions [19, 22]. Because, a number of virtual servers involved in provide transaction services through cloud. The goal is to ensure the ACID properties of a transaction that accesses multiple resource managers. Two-phase commit protocol has two phases, one is transaction coordinator and other one is more than one transaction participants are involved for each transactions. In distributed storage systems replication can be used to increase durability and availability of data as well as to enable fault tolerance and low latencies for distributed clients [13, 23]. Finally in this paper proposed a Depth1Fixed Tree Consistency (D1FTC) method to maintain ACID guarantee in cloud environment and it fully supports to implement the Two-phase commit protocol and replicate/ update data in cloud environment.

II. REVIEW OF LITERATURE

This section deals with efficient methods to handle the secured data transaction management in cloud. Md. Ashfakul Islam, et al proposed a Tree-Based Consistency Approach for Cloud Databases , This paper defined how tree based consistency approach reduces interdependency among replica servers by introducing partially consistent and fully consistent state of cloud databases[7,8]. Tim Kraska, et al proposed a paper for multi-data centre consistency. This paper described how MDCC can commit transactions in a single round-trip across data centers in the normal operational case and also propose a new programming model which empowers the application developer to handle longer and unpredictable latencies caused by inter-data center communication [9]. Peter Bailis, et al proposed a paper for bolt-on causal consistency. This paper developed a shim layer that upgrades eventually consistent stores to provide convergent causal consistency and describe algorithms and shim implementations those are suitable for a large class of application-level causality relationships [10]. David Bermbach, et al proposed a paper ttowards comprehensive measurement of consistency guarantees for cloud-hhosted data storage services. This paper motivates further research on building a standard comprehensive bench-mark for quantifying the consistency guarantees of eventually consistent storage systems [11]. Katarina Grolinger et al proposed a paper for Data management in cloud environments: NoSQL and NewSQL data stores. This paper reviewed that NoSQL and NewSQL solutions with the objective of providing a perspective in the field, providing guidance to practitioners and researchers to choose the appropriate data store, and identifying challenges and opportunities in the field [12]. Sebastiano Peluso, et al proposed a paper when scalability meets consistency: Genuine Multiversion update-Serializable Partial Data Replication. This paper introduce GMU, a genuine partial replication protocol for transactional systems. This consistency criterion is particularly attractive as it is strong to ensure correctness even for very demanding applications [13]. Alexander Thomson, et al proposed a paper for Calvin: Fast Distributed Transactions for Partitioned Database Systems. This paper described how Calvin supports disk-based storage, scales near-linearly on a cluster of commodity machines, and has no single point of failure [14]. Rohan G. Tiwari, et al proposed a consistency model for data stored in cloud environment and developed a framework towards the goal of deploying database as a service over cloud. This paper also proposed an effective algorithm to ensure the distributed consistency of data without compromising the availability of the replicated data [15]. R. Anandhi, et al proposed a paper for improving the consistency of transactions

in cloud databases scalability. This paper explains the types of scalability, choosing the correct scalability and other issues. It also shows the way to improve cloud scalability [16].

III. EXISTING CONSISTENCY METHODS

The word consistency is derived from Latin word consistere which means “standing together" or also “stopping together". So consistency normally describes relationships between items that are someway connected [4]. The consistency of data is a consistent state which is need that all relationships between data items and replicas. It focuses on the correctness and it can be seen in both the database as well as the distributed systems community.

a) Client-centric Consistency:

The distributed storage system has two perspectives on consistency the provider views the internal state of the system. His focus is on the synchronization processes among replicas and the ordering of operations [3, 17]. Hence, this perspective is called data-centric. The other perspective is the one of a client of the storage system. Here, a client refers to the process that interacts with the storage system which can be any kind of application, middleware or even software running on the end user's machine or mobile device. Both perspectives have advantages and limitations for the analysis of consistency guarantees - depending on the issue of interest [6].

a.1) Monotonic Read Consistency (MRC):

It is helpful as from an application perspective data visibility might not be instantaneous but versions at least become visible in chronological order [17].

a.2) Read Your Writes Consistency (RYWC):

It helps as for example, to avoid user irritation when person ‘A’ checks his bank account statement, does not see the transaction and consequently wires the same amount of money again.

a.3) Monotonic Writes Consistency (MWC):

It is useful to avoid apparently lost updates when an application writes and then updates a datum but the update is executed before the initial write and is, thus, overwritten.

a.4) Write Follows Read Consistency (WFRC):

This model essentially extends MWC guarantees to updates by other clients that have at least been seen.

b) Data-centric Consistency:

In this section, we will present data-centric consistency models ordered by the strictness of their guarantees and talk about for each model how it can be translated into a client-centric consistency model [17].

b.1) Weak Consistency:

As Weak Consistency does not provide any ordering guarantees at all, there is no relation to client-centric consistency models [5,18].

b.2) Eventual Consistency (EC):

It is a little stricter. It requires convergence of replicas in the absence of updates and failures the system unites towards a consistent state. Updates may be reordered in any way possible and a consistent state is simply maintain as all replicas being identical EC is very hazy in terms of real guarantees but is very popular for web-based services. Most of the NoSQL systems implemented EC [19].

Currently, there are three foremost strategies or approaches to protect the consistency of high-distributed environments, such as: *Classic*, *Quorum*, and *Tree-based*

a) Classic Approach:

The basic concept of *Classic strategy* is to preserve consistency through a simple, synchronous replication of distributed environment. In this strategy, each operation of writing is participate in all nodes or DBMS replica servers of environment, therefore this strategy does not show a sufficient performance for the transaction intensive data processing [20].

b) Quorum Approach:

The Quorum strategy is more advanced and frequently exploited strategy in cloud environments to replicating highly-distributed DBMS database. The same is based on the so-called quorum voting of replica nodes - clustered servers by some basis - cooperating during execution of each operation on transactional DBMS database. These quorum servers confirm each operation that is voted by a majority number of members of a quorum. Even though this strategy should show improved performance than the Classic, in practice this is not so good because its execution on database exhibit a slowdown because it is just for the sake of the voting quorum. This is still the most commonly used strategy because it gives a high guarantee on the consistency of cloud database [20].

c) Tree-Based Approach:

This is one of the advanced strategies of management data consistency and integrity for highly-distributed DBMS environments those are actually based on complex tree structures. Hence, one of these strategies is Tree-Based Consistency (TBC) that promotes varying degrees of data consistency in relation to the level of tree replica-server which can be accessed in a single operation. This strategy introduces component and the dynamic of variability of cloud environment with distributed database across a large number of nodes and replicas [7, 8]. Thus the highest levels of TBC tree composed the various replica nodes provide the strongest guarantees of consistency while the same decreases moving throughout the tree - towards the very leaves of the tree. At single leaves it can be found the weak guarantees for data consistency in some segments of cloud database.

Advantages and Limitations of existing consistency approaches:

TABLE- 1

Consistency Approaches	Advantages	Limitations
Classic Approach	• Reliable approach for read transactions • Simple structure for implementation	It expects response from all nodes
Quorum	• It expects response from	It needs more time for data replication

Approach	only selected nodes • Reduce execution time compare with classic approach	between all nodes [20].
Tree-Based Approach	• Reduce the execution time compare with classic and quorum approaches • Cheep node can be used for the replica servers [20].	• Very complex structure • Performance differs in sparse, medium, and dense tree. • Not reliable for small transactions • Decrease the response time when increase density of the tree [20].

The above analysis of existing consistency approaches have many issues in their performances. So that the data transactions need a better consistency method or approach with minimum execution time and also strictly maintain ACID properties.

IV. DISTRIBUTED TWO-PHASE COMMIT PROTOCOL

Two-phase commit protocol has two phases, one is transaction coordinator and another one is more than one transaction participants are involved for each transaction. The following table illustrates the overall process and circumstances of the distributed two-phase commit protocol [20].

Phase 1:

The coordinator sends a request-to-prepare message to each participant involved in the transaction. The coordinator waits for all participants to vote. Each participant may vote Prepared message if it's ready to commit or may vote No message for any reason or may delay voting indefinitely.

TABLE- 2

S. No	**Transaction Coordinator (TC)**	**Transaction Participants (TP)**	**Status / Decision**	
			Commit	Abort
Phase – I				
1	Request-to-Prepare [Send to all Participants]	Preparing	____	____
2	Wait for Response [From all Participants]	,,	____	____
3	,,	Send Prepared Message OR NO Message [send to coordinator]	____	____
Phase – II				
4	If(get Prepared Message From all Participants)	Prepared	**Commit**	____
5	Otherwise	No		**Abort**
6	Wait for acknowledge	acknowledge	Done	Done

Phase 2:

If coordinator receives Prepared message from all participants, it comes to a commit state. Otherwise, it comes to an abort state. The coordinator throws its decision to all participants (i.e. Commit or Abort). Participants send acknowledgement of Commit or Abort by replying Done to the coordinator [21].

V. PROPOSED DEPTH 1 FIXED TREE CONSISTENCY (D1FTC) METHOD

The proposed D1FTC method is very comfort with data transactions in cloud. It efficiently supports 2 phase commit protocol to execute each transaction without affect ACID properties even in the critical situations. The preliminary setup and methodology of the proposed D1FTC method is elaborately explained as follows.

Preliminary setup (Implementation):

Step 1: Create an undirected graph for available nodes

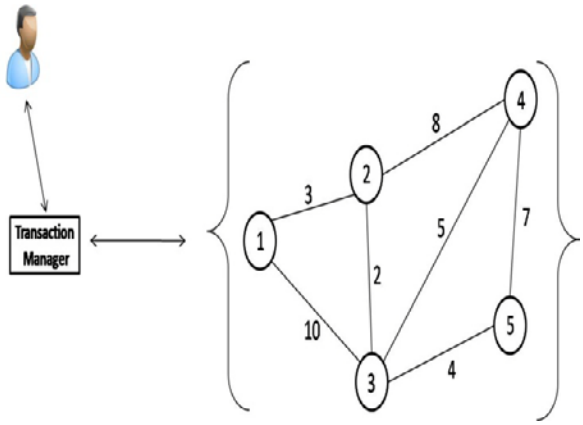

Fig. 1. undirected graph for cloud virtual machines

Step 2: Refer Adjacency List to create possible Depth 1 Fixed Trees (D1FTC)

This Adjacency List used to find possible depth 1 fixed trees for undirected graph consider as cloud virtual machines.

The adjacency list for the graph is as follows:

Node	Possible Linked Nodes
1	2 3
2	1 3 4
3	1 2 4 5
4	2 3 5
5	3 4

Fig. 2. Adjacency List

It specifies number of nodes and possible for linked nodes to each node. It is used to create possible depth 1 fixed trees for the graph as follows:

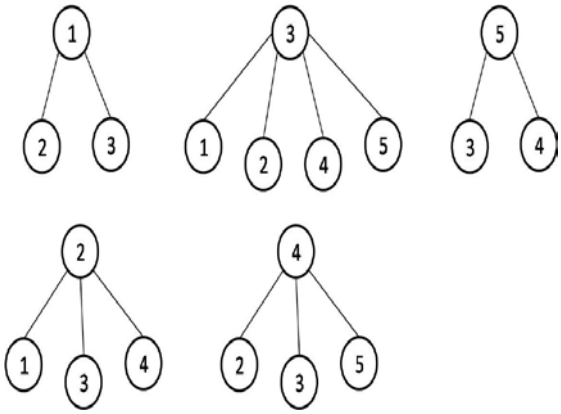

Fig. 3. Possible Fixed Trees

Step 3: Refer Adjacency Matrix to calculate the distance between each nodes in D1FTC

The adjacency matrix is used to find number of nodes for each tree and sum of distances between nodes in the tree. The adjacency matrix for the above mentioned undirected graph is as follows:

	1	2	3	4	5	No. Of Nodes	Distance between Nodes
1	α	3	10	α	α	3	13
2	3	α	2	8	α	4	13
3	10	2	α	5	4	5	21
4	α	8	5	α	7	4	20
5	α	α	4	7	α	3	11

Fig. 4. Adjacency Matrix

Step 4: Refer Dijkstra's Algorithm to find the shortest path from all nodes

The Dijkstra's Algorithm is used to find the shortest path from a given node to all other reachable nodes. In the following graph Fig (a), desire to find shortest path from node 1. Edge values in the graph are weights and Node values in the tree are total weights. The transaction manager fixes one nearest node for starting position to find shortest path to connect all nodes. Fig (b) is the shortest path from node 1 and Fig (c) is created as fixed tree for shortest path to update between all cloud servers after successful commit of the cloud transaction. So after every transaction is committed the transaction manager is replicated the data with the help of shortest path of fixed tree.

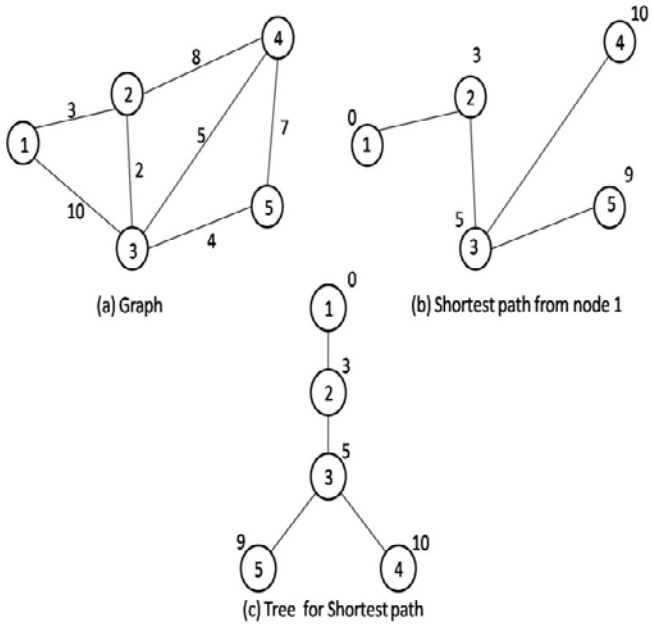

Fig. 5. Find shortest path

D1FTC Method:

After successful implementation of D1FTC method among the cloud virtual machines it is ready to execute the cloud transactions. The components of D1FTC are as follows:

User
Transaction Manager
Fixed Trees
Transaction Coordinator
Transaction Participants
Two-Phase commit protocol
Shortest path Trees

a) User:

The user can interact with the transaction manager to submit the transaction request and get response from it without loss of details.

b) Transaction Manager:

Transaction manager is responsible to maintain all the transactions held in a cloud system. It analyzes whether a particular transaction is search or update operation, if it is for the update operation it will fix the fixed tree with transaction coordinator and transaction participants.

c) Fixed Trees:

The structures of these possible fixed trees are reliable to implement two-phase commit protocol, because it has one to many relationships that is one transaction coordinator and many transaction participants for all update transactions. A participant need not affect the other participants and it communicates only with the coordinator. So the transaction manager can choose any one of the node for transaction coordinator and linked nodes under the transaction coordinator are chosen as transaction participants.

c.1) Adjacency Matrix:

The transaction manager fixes any one of the tree that depends on the required nodes needed to execute a transaction. So the adjacency matrix calculates the number of nodes and the distance between nodes that is to simplify the work of transaction manager.

d) Transaction Coordinator:

Transaction Coordinator is responsible for the given transaction and it maintains all the participants in the selected fixed tree to commit a transaction. The two-phase commit protocol implements in the transaction coordinator.

e) Transaction Participants:

The transaction divided into small no of process and it sends to transaction participants in the fixed tree. All participants are under supervised by the transaction coordinator.

f) Two-Phase commit protocol:

Two-Phase commit protocol is one of the efficient ways to execute data transactions in the distributed system. It can help successfully and execute the transactions with ACID guaranties in cloud environment. It is also very reliable for the proposed D1FTC method.

g) Shortest path Trees:

In cloud, data replicates from large geographic distance for every transaction and data may be loss during the replication process. The work of the finding shortest path is to connect all virtual machines. It is to avoid the inconsistency database and minimize the replication time in cloud environment.

The methodology of D1FTC is as follows:

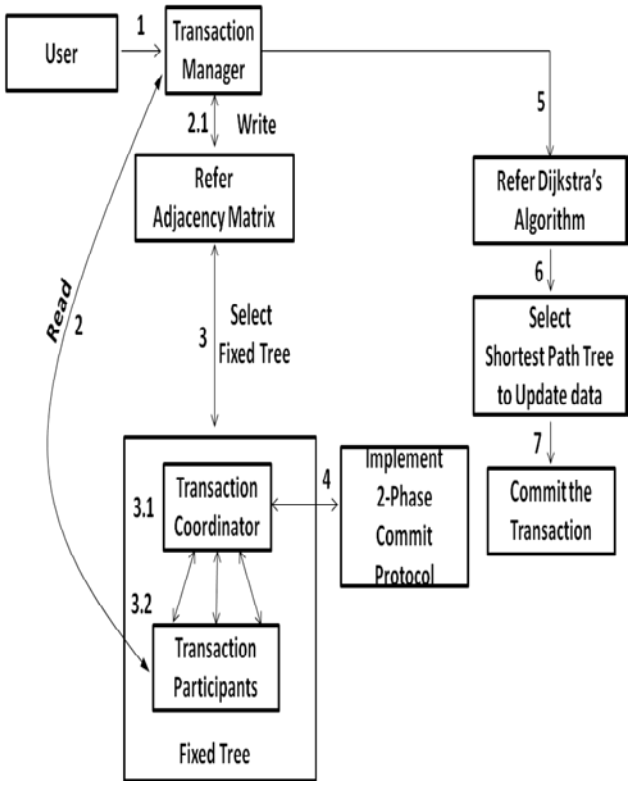

Fig. 6. D1FTC methodology

Algorithm for Depth 1 Fixed Tree Consistency (D1FTC) Method :

Step 1 : User send request to transaction manager

Step 2 : If (Read operation)

{

Access any node and get updated data.

}

Else

Step 2.1: Refer Adjacency Matrix

Step 3 : Select the Fixed Tree [According to the transaction querry,]

Step 3.1: Fix the root node as transaction coordinator(TC) and Access Transaction Coordinator through Transaction Manager(TM)

Step 3.2: Fix the child nodes are transaction participants

Step 4 : Implement 2-phase commit protocol in the TC.

TC divide the transaction and send it to Transaction Participants (TP) and execute the transaction

Step 5 : Refer Dijkstra's Algorithm

Step 6 : Select shortest path tree and Replicate/update data.

Step 7 : Successfully commit the transaction

VI. FEATURES OF THE PROPOSED D1FTC METHOD

Easy implementation:

The implementation of the proposed method is simple among cloud virtual machines. Adjacency list, adjacency matrix and Dijkstra's Algorithm are efficiently used for better reliability for cloud transactions.

Ensure ACID guaranties:

The proposed D1FTC method maintains *Consistency* for data transactions and after execution data are updated with the shortest path tree to maintain the *Durability* of successfully committed transactions. The 2-phase commit protocol implemented in each fixed trees to maintain the *Atomicity* and *Isolation* properties for the data transactions. So finally the proposed D1FTC method ensures ACID guaranties in cloud environment

Minimize the response time:

The adjacency matrix identifies the suitable fixed tree for every submitted transaction. It calculates a number of nodes and sum of distance between involved nodes in each tree. And it chooses apt fixed tree for transactions, so it can be able to locate fixed tree for transactions according to the weight of submitted transactions. This efficient methodology may minimize the response time to commit transactions in cloud environment.

Reliable structure for 2-phase commit protocol:

The proposed D1FTC method contains a number of depth one trees has one -to- many relationships structure. The root node consists of transaction coordinator (TC) and the child nodes are transaction participants (TP). Most of the messages passed between one TC to many TP vice versa, so the two phases of protocol execute the functionality very reliable in the proposed D1FTC method.

Efficient data replication:

After successful execution of each transaction needs to update data in all servers in the cloud. In cloud, virtual machines are scatter in distances. The proposed method uses Dijkstra's Algorithm to find the shortest path tree from every node to connect all nodes. So it helps the transaction manager can select any nearest node with shortest path tree to update/ replicate data efficiently with in the short span of time when compare with other approaches.

VII. CONCLUSION AND FEATURE WORK

Data transactions are prospective candidates in cloud environment. Execution of a transaction with strict ACID properties is not easy in cloud. It needs a better consistency method to improve transaction speed and also ensure ACID guaranties. In this paper, a new efficient D1FTC method is proposed for improving data transactions in cloud databases. The components of proposed method like Transaction Manager, Fixed Trees, Transaction Coordinator, Transaction Participants, Two-Phase commit protocol and Shortest path Trees are strengthened cloud data transactions without loss of data and transaction properties. It has easy implementation procedure and surely minimizes the execution time in cloud. The future work is to implement the proposed D1FTC method in number of cloud virtual machines and analyze comparative study with existing consistency methods to prove D1FTC is an efficient method for cloud data transactions.

VIII. REFERENCES

[1] Anand Tripathi and Gowtham Rajappan, "Scalable Transaction Management for Partially Replicated Data in Cloud Computing Environments" , IEEE 9th International Conference on Cloud Computing, 2016

[2] Min Shen, Ajay D. Kshemkalyani, and Ta-yuan Hsu, "Causal Consistency for Geo-Replicated Cloud Storage under Partial Replication", IEEE International Parallel and Distributed Processing Symposium Workshops, /15 $31.00 © 2015 IEEE, DOI 10.1109/IPDPSW.2015.68

[3] Vinit Padhye, Anand Tripathi, "Scalable Transaction Management with Snapshot Isolation on Cloud Data Management Systems ", IEEE Fifth International Conference on Cloud Computing, 2012.

[4] Indu Arora, Dr. Anu Gupta, "Developing and Validating Virtualized Transactional Application of Educational Institutes using InFraMegh" , Intl. Conference on Advances in Computing, Communications and Informatics (ICACCI), Sept. 21-24, Jaipur, India, 2016.

[5] Marian K. Iskander Dave W. Wilkinson Adam J. Lee Panos K. Chrysanthis, "Enforcing Policy and Data Consistency of Cloud Transactions", 31st International Conference on Distributed Computing Systems Workshops, 2011.

[6] Amir Mohamed Talib, Rodziah Atan, Rusli Abdullah & Masrah Azrifah ,Azmi Murad , "Security Framework of Cloud Data Storage Based on Multi Agent System Architecture - A Pilot Study", 978-1-4673-1090-1/12/$31.00 ©2012 IEEE.

[7] Md. Ashfakul Islam1, Susan V. Vrbsky2 and Mohammad A. Hoque3, "Performance Analysis of a Tree-Based Consistency Approach for Cloud Databases", International Conference on Computing, Networking and Communications, Cloud Computing and Networking Symposium, 978-1-4673-0009-4/12/$26.00 ©2012 IEEE

[8] Md. Ashfakul Islam and Susan V. Vrbsky, "Tree-Based Consistency Approach for Cloud Databases", 2nd IEEE International Conference on Cloud Computing Technology and Science, 978-0-7695-4302-4/10 © 2010 IEEE, DOI 10.1109/CloudCom.2010.87

[9] Tim Kraska Gene Pang Michael J. Franklin Samuel Madden, "MDCC: MultiData Center Consistency", rearXiv: 1203.6049v1 [cs.DB] 27 Mar 2012

[10] Peter Bailis†, Ali Ghodsi†,‡, Joseph M. Hellerstein†, Ion Stoica†, "Bolt-on Causal Consistency ", SIGMOD'13, June 22–27, 2013, New York, New York, USA., Copyright 2013 ACM 978-1-4503-2037-5/13/06

[11] David Bermbach1, Liang Zhao2, and Sherif Sakr2, "Towards Comprehensive Measurement of Consistency Guarantees for Cloud-Hosted Data Storage Services", Researchgate, Conference Paper · August 2013, DOI: 10.1007/978-3-319-04936-6_3

[12] Katarina Grolinger, Wilson A Higashino, Abhinav Tiwari, Miriam AM Capretz, "Data management in cloud environments: NoSQL and NewSQL data stores", Journal of Cloud Computing: Advances, Systems and Applications 2013,2:22 doi:10.1186/2192-113X-2-22

[13] Sebastiano Peluso, Pedro Ruivo, Paolo Romano, "When Scalability Meets Consistency: Genuine Multiversion Update-Serializable Partial Data Replication", inescid lisboa, November 2011.

[14] Alexander Thomson, Thaddeus Diamond, Shu-Chun Weng, "Calvin: Fast Distributed Transactions for Partitioned Database Systems", SIGMOD '12, May 20–24, 2012, Scottsdale, Arizona, USA.Copyright 2012 ACM 978-1-4503-1247-9/12/05

[15] Rohan G. Tiwari, Shamkant B. Navathe, "TOWARDS TRANSACTIONAL DATA MANAGEMENT OVER THE CLOUD", Second International Symposium on Data, Privacy, and E-Commerce,2010

[16] R. ANANDHI, K. CHITRA, " A Challenge in Improving the Consistency of Transactions in Cloud Databases - Scalability" , International Journal of Computer Applications (0975 – 8887) Volume 52– No.2, August 2012

[17] David Bermbach1, Liang Zhao2, and Sherif Sakr2, "Towards Comprehensive Measurement of Consistency Guarantees for Cloud-Hosted Data Storage Services", Researchgate, Conference Paper · August 2013, DOI: 10.1007/978-3-319-04936-6_3

[18] Adewole Ogunyadeka, Muhammad Younas , Hong Zhu, Arantza Aldea, "A Multi-Key Transactions Model for NoSQL Cloud Database Systems", IEEE Second International Conference on Big Data Computing Service and Applications, 2016

[19] B.Jeevarani, Dr.K.Chitra, "Improved Consistency Model in Cloud Computing Databases ", 2014 IEEE International Conference on Computational Intelligence and Computing Research, 978-1-4799-3975-6/14/$31.00 ©2014 IEEE

[20] Jasmina Dizdarevic, Zikrija Avdagic, "The aspects of consistency management of highly-distributed transactional database in a hybrid cloud environment for the energy sector", XI International Symposium on Telecommunications (BIHTEL) ©2016 IEEE

[21] Zhou Wei, Guillaume Pierre, Chi-Hung Chi, "Scalable Join Queries in Cloud Data Stores", 12th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing, 2012.

[22] Zhou Wei, Guillaume Pierre, Chi-Hung Chi, "CloudTPS: Scalable Transactions for Web Applications in the Cloud" , Ieee Transactions on Services Computing, Special Issue On Cloud Computing, 2011, 1939-1374/11/$26.00 © 2011 IEEE

[23] Ms. Shalini Ramanathan, Dr.Savita Goel, Mr.Subramanian Alagumalai,"Comparison of Cloud Database: Amazon's SimpleDB and Google's Bigtable", 2011 International Conference on Recent Trends in Information Systems.

[24] Donald Kossmann, Tim Kraska, Simon Loesing, "An Evaluation of Alternative Architectures for Transaction Processing in the Cloud", SIGMOD'10,June 6–11, 2010, Indianapolis, Indiana, USA.Copyright 2010 ACM 978-1-4503-0032-2/10/06"

[25] Tharam Dillon ,"Cloud Computing: Issues and Challenges", International Conference on Advanced Information Networking and Applications, 2010 24th IEEE

BIOGRAPHIES

Prof. J. Antony John Prabu is working as Assistant Professor and pursuing doctor of philosophy in Department of Computer Science, St. Joseph's College, (Autonomous), Tiruchirappalli, Tamil Nadu, India. He received his M.Phil degree from Jamal Mohamed College, Tiruchirappalli. He received his MCA degree from St. Joseph's College, Tiruchirappalli. His areas of interest are Cloud Data Transactions and Distributed Technologies.

Dr. S. Britto Ramesh Kumar is working as Assistant Professor in the Department of Computer Science, St. Joseph's College (Autonomous), Tiruchirappalli, Tamil Nadu, India. He has published many research articles in the National/International conferences and journals. His research interests are Cloud Computing, Data Mining, Web Mining, and Mobile Networks.