Study of Concurrency Control Techniques in Distributed DBMS

Article · December 2018

DOI: 10.30991/IJMLNCE.2018v02i04.005

3 authors, including:

Manoj Kumar Gupta
Trinity Institute of Innovations in Professional Studies
33 PUBLICATIONS **81** CITATIONS
SEE PROFILE

Rakesh Kr Arora
Krishna Engineering College
18 PUBLICATIONS **56** CITATIONS
SEE PROFILE

IJMLNCE JOURNAL

International Journal of Machine Learning and Networked Collaborative Engineering

Journal Homepage: http://www.mlnce.net/home/index.html

DOI : https://doi.org/10.30991/IJMLNCE.2018v02i04.005

Study of Concurrency Control Techniques in Distributed DBMS

[a]Manoj K. Gupta[*], [b]Rakesh K. Arora, [c]Bhoopesh Singh Bhati

[a]Professor, Rukmini Devi Institute of Advanced Studies, Delhi, India.

manojkgupta5@gmail.com, http://orcid.org/0000-0002-4481-8432

[b]Asso. Professor, Krishna Engineering College, Ghaziabad, UP, India.

rka1211@gmail.com , https://orcid.org/0000-0002-8601-2816

[c]Asstt. Professor, Ambedkar Institute of Advanced Communication Technologies and Research, Delhi, India.

bhoopesh.cse@gmail.com , https://orcid.org/0000-0001-8476-2798

Abstract

Concurrency control is one of the important task of any database management system. Without the proper concurrency control technique it is infeasible to maintain the integrity of the database system in concurrent environment. The concurrency control algorithms focus on maintaining consistency and integrity of databases through synchronized access. In centralized environment it is simple to synchronize among the various concurrent transactions. But, it becomes very complex as compared to centralized framework when the concurrency control algorithms are implemented for distributed framework because of the requirements of consistency and integrity within the multiple fragments / copies of the database, synchronization among the various distributed concurrent transactions, and isolation of the complexities of the algorithms / operations. A variety of concurrency control methods have been proposed by the many researchers so far for the centralized framework as well as for the distributed framework. The concurrency control methods are broadly classified as locking based methods, timestamp ordering based methods and optimistic methods. Two-phase locking based methods are the most popular and widely used methods used in both centralized and distributed framework of the database systems. A number of algorithms based on two-phase locking have been proposed by the many researchers for Distributed DBMS. This paper consolidates and discusses various lock based concurrency control techniques for Distributed DBMS. This paper also presents a comparative study of various two phase locking based concurrency control techniques.

Keywords

Database,

Distributed Database,

Distributed DBMS,

Concurrency Control Techniques,

Two Phase Locking,

1. Introduction

Database system is the backbone of the many organizations as is stores and manages the operational data of the organization. Most of the applications of the organizations are dependent on database systems. The database systems can be either implemented based on centralized approach or distributed approach. The

[*] Corresponding author

Manoj K. Gupta

Email: manojkgupta5@gmail.com

database system which implemented based on distributed approach is called as Distributed Database System. A database system (DBS) is a combination of Database Management System (DBMS) and Databases of the organization. The shift from centralized to distributed architectures over the years attributes to the demand for higher performance and availability. Distributed database system (DDBS) technology in the field of database systems is also the result of the same. This technology may be viewed as combination of database system, computer network technologies and the concept of distributed computing (or distributed processing) [3, 12, 14, 17].

"*A distributed database (DDB) involves a collection of number of logically related databases spread over a computer network*" [9]. Özsu [9] defined *distributed database management system* (DDBMS) as "*the software system concerned with the management of the distributed database and makes the distribution transparent to the users.*"

The combination of DDB and DDBMS is referred to as "distributed database system" (DDBS).

Internally, DBMSs perform several functions in order to manage and manipulate the data properly such as transaction management, concurrency control, recovery, security, etc. The concurrency control ensures the consistency and integrity of the databases in the concurrent execution environment. As we know that the DBMSs support database sharing among various transactions. The uncontrolled concurrent execution of the transactions may lead to the database systems into an inconsistent state. Therefore, there is requirement of controlling the concurrent execution of the transactions so that the consistency and integrity of the database systems can be ensured. The concurrency control is the mechanism through which DBMS can ensure the consistency and integrity of the databases even in the case of concurrent execution of the transactions without affecting the degree of concurrency.

2. Concurrency Control

In DDBS, at the same time, multiple users can access the database concurrently where each user thinks that he/she is working alone on dedicated system but this is not the case. In case of concurrent execution of transactions, the consistency of the database can ensured with the help of serializable schedules because the result obtained will be equivalent to one of the serial execution of the transactions. The use of only serial schedules limits the degree of concurrency hereby affecting performance. Therefore, concurrency control techniques need to be applied to guarantee that the schedules generated by concurrent execution of transactions are serializable [1].

Concurrency control involves the coordination among concurrent accesses to maintain consistency and integrity of database [1, 24]. The major problem in attaining this objective is to make ensure that database updates performed by one the transaction should not affect the updates and retrievals of the another transaction [9, 12, 13, 16, 25]. The problems regarding concurrency control is more difficult in a DDBMS due to the following reasons:

- Data may be accessed by the multiple users at number of distant sites,
- Database is fragmented and/or replicated across multiple sites,
- Complexity during the synchronization among concurrent transactions executed at multiple remote sites by multiple users, and
- Concurrency control techniques implemented at one location must ensure the consistency of the database at all other sites.

3. Concurrency Control Techniques

A number of concurrency control mechanisms have been proposed by the leading researchers so far. These concurrency control mechanisms broadly can be classified in the categories such as (a) Locking Based Protocols, (b) Timestamp-Ordering Based Protocols and (c) Optimistic Protocols. The first two categories are based on the pessimistic approach. The pessimistic protocols are used when there is high activity on the databases whereas the optimistic protocols are used in case of low activity on the databases. Implementation of these concurrency protocols / algorithms for the centralized environment is simpler than that of distributed environment.

The number of concurrency control techniques implemented for centralized environment can easily be extended to handle the problem in distributed databases, but there are some concurrency control techniques that are not suitable for a distributed environment [1, 16, 25]. The detailed classification of concurrency control techniques is as follows:

3.1. Pessimistic Techniques

These techniques involve the synchronization among the transactions during initial phase of their execution life cycle as these consider the high conflicts among the concurrent transactions. The pessimistic techniques are mainly suitable when there is high activity in the database system. In other words, if a large number of conflicting transactions are executed concurrently on the database systems frequently then it is preferred to use pessimistic techniques which reduce the wastage of resources if conflicts identified at later stage. In addition, if particular application involves activity that is beyond the rolling of database capability like printing, than such activity do not take place whenever conflict is generated, hereby removing the responsibility of performing undo operation. The pessimistic techniques further can be classified as:

3.1.1. Two Phase Locking (2 PL) Based techniques

Two phase locking technique is based on the locking technique which ensures the serializability of the concurrent transactions in order to maintain the consistency and integrity of the database system. In this technique, each data object of the database is associated with a shared variable called as lock which stores the state of the data object in order to control the shared access of the data object by the mutually exclusive transactions [1, 9, 19, 21, 23]. In two-phase locking, there are two phase of operations. The first phase is called as growing phase in which the transactions can acquire or upgrade the locks and in the second phase the transactions can release or degrade the locks only. To implement two phase locking the following rules need to be followed (a) Conflicting locks should not exist in two transactions. (b) Unlock operation cannot be performed before lock operation in any of the transaction. (c) Until and unless all locks are obtained, no data are affected in any of the transaction. This approach may results in deadlocks and starvation.

3.1.2. Timestamp-Ordering Based techniques

In this technique, the transactions are ordered based on some value called as timestamp. The timestamp can be either a value generated by a global incremental counter variable or current timestamp of the master clock which ensures the uniqueness of the timestamp. The transaction which is created earlier is assigned a lower timestamp as compared to the transaction which is created later. The transaction with lesser timestamp is older than the transaction with higher timestamp. Therefore, the ordering of transaction can be done based on the assigned timestamp and hence called as timestamp-based ordering technique which avoids the deadlock situation [1, 19, 20, 22, 23]. In this technique, each data item X is stored as a triplet X = {*x*, *write-timestamp*, *read-timestamp*} where *x* is the latest value of X, *write-timestamp* is the timestamp of the youngest transaction who has written the latest value of *X*, and *read-timestamp* is the timestamp of the youngest transaction who has read the value of *X*. There are two-approaches based on this technique (a) wait-die and (b) wound-wait. This technique has the advantage of eliminating deadlocks as the transactions need not to wait. This technique results in cascading rollbacks. Starvation can also occur if same transaction is aborted and restarted again and again.

3.1.3. Integrated

Both two-phase locking technique and timestamp-based ordering techniques are having their relative pros and cons. In order to exploit the pros of both the techniques, some DDBMS uses the integrated approach in which combination of the above two techniques are implemented at the same time. This approach proves to be advantageous when heterogeneous databases are connected together [1].

3.2. Optimistic techniques

In these techniques, synchronization of concurrent execution of transactions is delayed until their termination [19, 20, 21]. These techniques allow the transactions to perform their operations as they desire except write-phase in order to maximize the degree of concurrency. In this approach, the operations of the transactions are logically divided in to three phases (a) read phase, (b) validation phase, and (c) write phase. In the read phase, all the transactions are freely allowed to perform their operations in the local memory. Before writing the changes to the database the transaction has to pass the validation phase in which DDBMS

check whether the proposed write phase of the transaction will lead the database in to consistent state or not. If validation phase indicates the consistency of the database then the write-phase will be performed by the transaction otherwise the write-phase of the transaction is denied. Optimistic approach can be implemented either using 2PL or timestamp-based ordering techniques. No chance of cascading rollback is there because the actual write operation occurs only when the transaction initiating the write operation has committed.

The comparison between pessimistic and optimistic approach is presented in Table 1.

Table 1: Comparison between Pessimistic Approach and Optimistic Approach

Basis	**Pessimistic Techniques**	**Optimistic Techniques**
Synchronization	The synchronization of transactions occurs during initial phase of their execution life cycle.	The synchronization of concurrent execution of transactions is delayed till their termination.
Conflict	Large number of transactions will conflict with each other.	Relatively less number of transactions is in conflict with each other.
Phases of Transaction Execution	Validate, Read, Compute, Write	Read, Compute, Validate, Write
Strengths	• Simple • Suitable where transactions conflicts are more. • Lower Storage Cost	• Submitted operations never delayed. • Higher degree of concurrency
Limitations	• Lower degree of concurrency • May lead to deadlocks	• Suitable for applications where transactions conflicts are very rare. • Wastage of Resources (in case of failure of Validation phase) • Higher Storage Cost for intermediate results • Originally based only on timestamp ordering. • Concentrates on centralized DBMS.

4. Two Phase Locking Based Techniques

Locking-based techniques are widely used in centralized database systems in which logical and physical locks on data items are used for synchronization among transactions. If any transaction wants to read and/or write any granule of database, it has to first acquire the lock on that granule and after the completion of operation the locks has to be released. The locking and unlocking on data items is handled by lock manager on behalf of transactions. There are four major categories for implementing lock based algorithms:

4.1. Centralized two phase locking algorithm

It is a variation of 2 PL in which a single lock table is maintained at any one designated site for the whole distributed database. All the requests for locking and unlocking are sent to that site only [1, 26]. Only the designated site decides the grant of requisite lock to the requested transactions based on the compatability of requests. This technique is useful in the case of both replicated and fragmented distributed databases. The load on the single site will increase as all the rquests for locking and unlocking are made at single site only. If the designated site fails, then the operation of the DDBMS will fail.

4.2. Primary Copy two phase locking algorithm

This approach is useful when multiple copies of same data item are stored at different locations (replication). One of the copies at any particular site is designated as a primary copy. All the requests for locking and unlocking are sent to the designated site only with respect to the specific data items. [1, 14, 15, 18, 20]. This approach is suitable for replicated or mixed databases. Effect of the failure of one site is lesser than that of centralised two phase locking as each site is designated as primary copy for specific data items only.

4.3. Distributed two phase locking algorithm

Locking of data items are done at all sites where transaction accesses these data items. Once locking of all data items have been completed by the transaction, only than unlocking begins [24]. The trasactions are required to submit the locking and unlocking requests to all sites. The lock request will be granted to the transactions when it is allowed by all the sites. This approach is more complex then centralized two phase locking and primary copy two phase locking. In this approach, the network traffic will increase. Lot of communication overhead is involved and handling of deadlocks becomes more complex. This technique is more reliable as failure of centralized site will affect less number of transactions.

4.4. Majority Consensus 2 PL Algorithm

This technique is the variation of distributed two phase locking algorithm. Unlike, distribute two phase locking, it requires locking on majority of copies, i.e. at least (n+1) / 2 copies, instead of all copies. If the data item is to be updated, the transaction would have to send updated value to all sites where data item is stored [1, 18]. Less communication ovehead as compared to distributed two phase locking technique as less number of requests have to be made for locking and unlocking data items.This algorithm is more efficient than the distributed two phase locking algorithm.

The comparison among above two-phase locking based techniques is presented in Table 2.

Table 2: Comparison among Two Phase Locking (2 PL) Based Techniques

Basis	**Centralized 2PL**	**Primary Copy 2 PL**	**Majority Consensus 2 PL**	**Distributed 2 PL**
Control	The sites designated as primary site has the responsibility of granting locks on all data items to the transactions	For each data item , one of the site (in case of replication) is designated as the primary copy that has to be locked by the transactions	Each site has the responsibility of granting locks on its all local lock units to the transactions. A majority of copies are required to be locked instead of all copies	Each site has the responsibility of granting locks on its all local data items to the transactions
Lock Management	Centralized (only at one site)	Mixed (Centralized for each primary copy and Distributed for several primary copies)	Distributed (at each site)	Distributed Lock Management (at each site)
Number of Lock Managers	Only one (at primary site)	Equals to number of distinct lock units (for each primary copy)	Equivalent to the number of sites	Equivalent to the number of sites
Suitable for Data Distribution	Fragmented, Replicated or Mixed	Replicated or Mixed	Replicated or Mixed	Fragmented, Replicated or Mixed
Strengths	• Easy to implement	• Improved	• Number of messages	• Increased

	• Less costly	performance • Reduces the load on central site • Communication cost is less	for locking and unlocking is less • Number of deadlocks is relatively less.	performance for local transactions • Reduces the load of central site
Limitations	• Less reliable as the failure of the site would result in major system failures [9] • Performance degraded at high loads due increase in amount of work. • Bottleneck as all requests for locking and unlocking are made at same site.	• High Complexity • Demands a more sophisticated directory at each site	• High Complexity • Large communication overhead as compared to Centralized 2PL	• High Complexity • Large communication overhead • May cause the distributed deadlock • Demands a more sophisticated directory at each site

5. Conclusion

Several categories of concurrency control methods have been proposed and implemented for Distributed DBMS. The pessimistic methods are still widely used but are cause of concern as they may lead to number of problems like lowering the degree of concurrency, deadlocks, starvation and cascading rollbacks. The optimistic methods can be used to increase the degree of concurrency but the current works on optimistic methods concentrates mainly on centralized DBMS in contrast to distributed DBMS. The pessimistic approach is mostly suitable for the distributed database systems with high activity ratio whereas optimistic approach is mostly suitable for the distributed database systems with low activity ratio.

References

[1]. Bernstein, P. A. and Goodman N., "Concurrency Control in Distributed Database Systems". ACM Computing Surveys, 13(2), June 1981 (DOI: 10.1145/356842.356846).

[2]. Bernstein, P. A. and Newcomer, E., "Principles of Transaction Processing for the Systems Professional", Morgan Kaufmann, 1997.

[3]. Ceri, S. and Pelagatti, G., "Distributed Databases: Principles & Systems", McGraw-Hill.

[4]. Desai, B. C., "An Introduction to Database Systems", Galgotia, 2000.

[5]. G. Schlageter, "Problems of Optimistic Concurrency Control in Distributed Database Systems", ACM SIGMOD Record, 13(3):62–66, April 1982 (doi>10.1145/984505.984510)

[6]. http://www.cs.helsinki.fi/u/jplindst/

[7]. http://www.vldb.org

[8]. Kumar, V., "Performance of Concurrency Control Mechanisms in Centralized Database Systems", Prentice-Hall, 1996.

[9]. Özsu, M.T. and Valduriez P., "Principles of Distributed Database Systems", Prentice-Hall, 3rd Edition, 2011

[10]. Ramakrishnan, R., and Gehrke, J., "Database Management Systems", McGraw Hill International Edition, 2nd Edition

[11]. Silberschatz, A., Korth, H.F., and Sudarshan, S., "Database System Concepts", McGraw Hill International Edition, 5th Edition

[12]. Joshi, H. and Bamnote, G.R., "Distributed Database: A Survey", International Journal Of Computer Science And Applications Vol. 6, No.2, Apr 2013

[13]. Rawashdeh, Obaidah A., Hiba A. Muhareb, and Nedhal A. Al-Sayid. "An optimistic approach in distributed database concurrency control", 2013 5th International Conference on Computer Science and Information Technology, 2013.

[14]. Gupta, S., Saroha, K. and Bhawna, "Fundamental Research of Distributed Database", Intl. Journal of Computer Science and Management Studies, Vol. 11, Issue 02, Aug 2011

[15]. "Distributed Databases", Encyclopedia of Information Systems, 2004

[16]. Zubi, Z.S., "On Distributed Database Security Aspects", IEEE Explore, 2009

[17]. Tomar, P. and Megha, "An Overview of Distributed Databases", International Journal of Information and Computation Technology, Volume 4, Number 2 (2014), pp. 207-214

[18]. Yadav, A.K. and Agarwal, A., "A Distributed Architecture for Transactions Synchronization in Distributed Database Systems", International Journal on Computer Science and Engineering, Vol. 02, No. 06, 2010, pp. 1984-1991

[19]. Harding, R. and Aken, D.V., "An Evaluation of Distributed Concurrency Control", Proceedings of the VLDB Endowment, Vol. 10, No. 5, 2017

[20]. Batra, N. and Kapil, A.K., "Concurrency Control Algorithms and its Variants: A Survey", AIP Conference Proceedings, 2010

[21]. Sinde, V. and Aware, P.A., "Concurrency Control in Distributed Database Systems", International Journal for Research in Engineering Application & Management (IJREAM), Vol-01, Issue 10, JAN 2016

[22]. Kahoro, P., Wanjiru, C. and Karumba, N., "Concurrency Control In Distributed Databases", United States International University Africa, July 22, 2014

[23]. Abuya, T.K. and Cheruiyot W.K., "Guaranteeing Global Conflict Serializability in Concurrent Distributed Database Systems using Commitment Ordering", International Journal of Computer Science and Mobile Computing, Vol.3 Issue.6, June 2014, pg. 700-707

[24]. Sah, M.K., Kumar, V. and Tiwari, A., (2014) "Security and Concurrency Control in Distributed Database System", International Journal of scientific research and management (IJSRM), Volume 2, Issue 12, Pages 1839-1845, 2014

[25]. Gupta M.K. and Arora, R.K. (2009), Concurrency Control Techniques in Distributed DBMSs: A Comparative Study, IIMT Business Review 1 (1), 27-31

[26]. Vasileva S. And Milev A. (2016), "Simulation Studies of Distributed Two-phase Locking in Distributed Database Management Systems" , Information Technologies and Control, 13(1-2), DOI: 10.1515/itc-2016-0010

Author's Biography

Dr. Manoj Kr. Gupta is presently working as Professor at Rukmini Devi Institute of Advanced Studies (Aff. to Guru Gobind Singh Indraprastha University), Delhi, India. He is also Dean Examination, Admission & Administration and Course Coordinator MBA in the Institute. He has more than 20 years of experience in teaching and administration. His interest areas are Database Systems, Data Warehousing and Data Mining. He has 4 books and 18 international / national research papers to his credit.

Dr. Rakesh Kr. Arora is currently working as Associate Professor at Krishna Engineering College, Mohan Nagar, Ghaziabad, Uttar Pradesh, India. He has more than 16 years of teaching experience in reputed institutes. His interest areas are Data Warehousing and Data Mining, Database Systems and Operating System. He has number of papers in International Journals and Conferences to his credit.

Bhoopesh Singh Bhati is Pursuing Ph.D. degree from the Guru Gobind Singh Indraprastha University, Delhi. He has obtained his M.Tech. Degree in Information Security and B. Tech. (Computer Science and Engineering) from the Guru Gobind Singh Indraprastha University, Delhi. He is presently working as Assistant Professor in the Department of Computer Science and Engineering at Ambedkar Institute of Advanced Communication Technologies & Research, Govt. of NCT, Delhi, India. He has a number of Research Papers in International Journals and Conferences. His current research area is Information Security.

How to Cite

Gupta M. K., Arora R. K.,& Bhati B. S. (2018). Study of Concurrency Control Techniques in Distributed DBMS. *International Journal of Machine Learning and Networked Collaborative Engineering*, *2*(04) pp 180-187.

https://doi.org/10.30991/IJMLNCE.2018v02i04.005