ISSN (Online) 2394-2320

International Journal of Engineering Research in Computer Science and Engineering (IJERCSE)
Vol 4, Issue 10, October 2017

Multi version Concurrency Control for Object Oriented Database Management System

[1] Sonal Kanungo, [2] Rustom. D Morena
[1] Smt.Z.S.PatelCollege of Computer Application, Surat
[2] Department of Computer Science, Veer Narmad South Gujarat University, Surat

Abstract - **Now a day object-oriented databases (OODB) are being used in large scale applications in a different industry including telecommunications, banking, manufacturing, insurance, and shipping. These applications are categorized by having complex data, that is, data which is represented in the highly interconnected in an object model.**

Object databases are very good at storing the complex data model, but it is generally up to the application developer to figure out how to scale the application so that it runs efficiently with many concurrent users. The purpose of this paper is to explore various techniques that can be used in object oriented databases to achieve high concurrency.

1. INTRODUCTION

An Object-Oriented Database Management System (OODBMS) is a database that supports the modeling and creation of data as objects. The set of related class participating in the domain defines the structure of an OODBMS. Database relationships like aggregation, inheritance, composition and association relate the classes with each other. These relationships can be represented and understood as following - Inheritance as a "Is-a" relationship, Association as a “Has-a” relationship, Composition/ Aggregation as a “Is-part of” relationship.

In the traditional transactions, values of attributes are altered and read in the database as per the requirement. The effect of the frequent changes occurring in the business requirement is well reflected in the corresponding databases. The inherent schemas are changed in order to adapt to this change. This makes it imperative that the parallel execution of transactions be in order to maintain the concurrency with the changes.

Several multigranular locking models are presented for OODBMS. However, due to complex nature of OODBMS and its ability to create the relationship chain in domain gives birth to more relations. Due to this property, there is a need for developing more sophisticated concurrency control techniques.

1.1. Object Oriented Database Management System

There are areas where complex data are used such as Computer-Aided Design (CAD), Computer-Aided Software Engineering (CASE), Office Information System(OIS), and Multimedia Systems, where in enriched modelling capabilities of OODBMS supports these applications.

Object-oriented databases system can reduce maintenance, provide reusability of code and also improve flexibility and reliability of system.

1.2. Benefits of OODBMS

1.2.1. OODBMS is combination of database and object-oriented properties:

In OODBMS database properties are integrated with object oriented programming language capabilities. Object-oriented databases can work efficiently with object-oriented programming languages like Java, C#, and C++ etc.

1.2.2. No impedance mismatch problem:

Usage of RDBMS with object oriented languages results in loss of time; as the objects have to be mapped to table and vice versa. As there is no impedances mismatch problem in OODBMS, it results into improvement in performance.

1.2.3. Support complex structured data:

By using SQL, ODBC, and JDBC, the data can be stored in the tables in an easy and convenient form in OODBMS using Java or C++. An object can be a complex entity. The object contains references to both dependent objects and independent objects. An OODBMS can give performance that is much faster than RDBMS. The reason responsible for the higher efficiency is that, the data were already in the well accepted format of Java and C++; hence no translation is required at any stage of execution. Complex data is processed with much ease.

2. CONCURRENCY CONTROL

The interaction of simultaneous transactions is controlled by the various concurrency-control techniques. These techniques keep consistency in the databases. In order to provide better throughput, various concurrency control techniques are provided in the literature. The lock

mechanism is used in lock based protocols. In order to give concurrent access for data item, the permission is only given when the data item is holding a lock. In this condition, a lock cannot be granted to other incompatible requesting transactions and these transactions have to wait until all
incompatible locks held by previous transaction are released, then only lock is granted. Obtaining and releasing locks must be done properly. This is done in order to ensure proper consistency and also avoid deadlocks [1]. The protocols, which are Timestamp-Based, a unique timestamp is given to each transaction in order to ensure consistency. Timestamp is given when the transaction is started. For each data item two timestamps are stored. Write-timestamp is timestamp of last successful write for data item and Read-timestamp is the time stamp of last successful read. In order to access a data, the data's read and write timestamps must always be older than then requesting transaction, else this access is denied and aborted. If the current transaction is late then it has to be aborted. This mechanism ensures safer and consistent data transaction which is free from deadlocks. The only drawback is that the schedule may not be cascade-less, and also it makes the system non-recoverable.
The concept that the transaction will be executed successfully is called as Optimistic concurrency control. In the Optimistic concurrency control method, the transaction executes fully with the hope that all will go well during validation. The technique is called "optimistic" because it assumes that little interference will occur and hence there is no necessity to carry out checking during transaction execution. This is called optimistic as it is mainly dependent on transaction backup for the control mechanism in an optimistic way. The assumption behind Validation based protocols is that the occurrence of read and write conflicts will be very rare. During the transaction processing, the uncontrolled access to shared data objects is allowed. The presence of any conflicts must be checked by the validation based protocol. Conflict resolution mainly leads to transaction abort.
When most of the transactions are read-only transactions, then the rate of conflicts will be low. In the optimistic scheme, no records are locked and hence no deadlock will occur [3].
The condition where the transaction needs to access the whole database and the locking protocol has been utilized, everything in the database must be secured. This is a very tedious process. Hence the whole database is locked for better performance. Thus, various levels of granularity are characterized by the framework. Multi granularity locking protocol is graphically understood in the form of a tree [4]. Top-down order or the leaf to root structure is proposed by various granularity protocols. Deadlocks are found in this protocol [5].
One more issue that arises with concurrency control is that, a read operation might be delayed because the appropriate value has not been written yet; or it might be rejected

because the data that should have been read by this request has already been overwritten [6]. By keeping old duplicates of every data item, this issue could be taken care off very easily. A new duplicate is created with each successful write in a Multiversion method [5]. This offers better controlling on the request of reads and writes. The read and write don’t overwrite each other. The read request is never rejected in a Multiversion scheme [2].
The Multiversion mixed method uses the Timestamp ordering and Two-phase locking with multiple versions of data [3]. Each data and its version are labeled with the timestamps. Multiversion Update transactions follows rigorous Two-phase locking(2PL) protocol [5]. The Update transactions hold the read and write locks till the end of transaction. Read-only transactions can continue to read the committed version of data, while the transaction is holding the write lock [11]. Due to quick proper version return, the reads never need to wait [6]. Since writes don't overwrite each other and reads can read any version, it has greater adaptability in controlling the request of reads and writes [4].

3. NEED OF NEW CONCURRENCY CONTROL TECHNIQUE

The properties like implementation, correctness and maintaining consistency for the database locking mechanism is widely used and accepted. For object-oriented environments, basic concurrency control schemes cannot be adopted because of the inherent complex nature of the objects. Complex objects are not exploited in promoting concurrency. Also, as the transactions are of long duration in nature, the prevailing schemes for concurrency control aren’t equipped to support them. Due to this, system resources are underutilized and the throughput is reduced by letting the transaction to hold the resources for a longer period.

ISSN (Online) 2394-2320

International Journal of Engineering Research in Computer Science and Engineering (IJERCSE)
Vol 4, Issue 10, October 2017

Multi-granularity locking techniques support concurrency control for the object-oriented environment, but still these techniques are not up to the desired levels. Proposed technique works
very efficiently with changing environments and provide better throughput while working with the complex objects.
Both the Class transactions and Object transactions deal with the Concurrency control in OODBMS. The proposed technique satisfies both schemas and operations along with the data. The objective of this research is to find an efficient mechanism which can improve the performance and can also handle deadlock. To achieve this objective, the technique should provide all lock types, compatibility matrix and lock granularity for all types of class relationships like composition, inheritance, and association for an object-oriented database system.

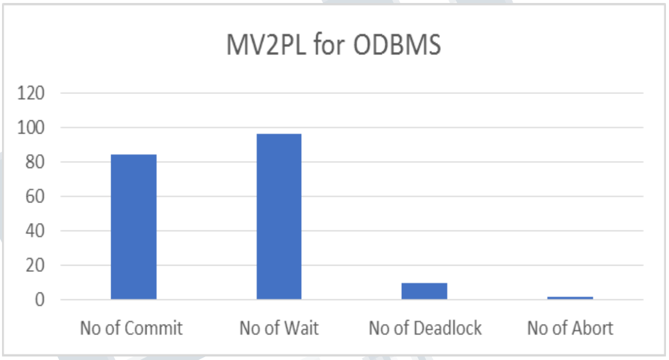
4. METHODOLOGY

Transaction generator can create random transactions which can Read or Update classes and objects. Transactions are serialized and follow modified Multiversion two phase locking that means all operations will get locked and will not be able to release until all operations will commit or rollback. Other transaction will get the lock only after the release of previous lock.
Each version of object and class has a unique timestamp. When the query (read only) wants to read data, it reads the latest committed versions, therefore read never fails in this technique.
When a group of blocked processes holding resources, and waiting to get resources held by another process in the set, results into a deadlock. By removing the victim, this technique also handles deadlock.

5. RESULT AND DISCUSSION

Experimental details and analysis of the simulation is presented in this section. The simulation is
implemented using visual C++ and results are generated in a standard environment. The transactions are served in first-in-first-out (FIFO) order and it is generated by the transaction generator. The Transaction generator produces the update and read-only transactions. If the lock mode is compatible with the existing transactions, then the transaction request is served; and if a request is found to be incompatible, the transaction will be blocked. Read-only will read consistent last committed version.
Four types of testing parameters viz number of commit, deadlock and abort, are considered for evaluating the performance of the existing techniques with our proposed technique.
We have taken 100 transactions, where total number of transactions in one Run is 10 and total number of Runs is 10, also Total numbers of operations on Object and Class in transaction vary in each Run.

Figure 1



CONCLUSION

This research is focused to examine the impact of read-write proportions on the performance of the Concurrency Control mechanism. For the analysis, the behavior of performances for different Multiversion Concurrency Control mechanisms, Simulation modeling has been used. We briefly review these works and their findings and argue that more work in this area is required for a good understanding of the behavior of Concurrency Control mechanism based on multiversion schemes.

REFERENCES

1. K.P. Eswaran, J.N. Gray, R.A. Lorie, and I.L. Traiger: "The Notions of Consistency and Predicate Locks in a Database System", Communications of the ACM, November 1976 of Volume 19, Pages 624-633.

2. DANIEL J. ROSENKRANTZ, RICHARD E. STEARNS, PHILIP M. LEWIS II: "System Level Concurrency Control for Distributed Database Systems",

ACM Transactions on Database Systems, Vol. 3, No. 2, June 1978, Pages 178-198.

3. H.T. KUNG AND JOHN T. ROBINSON: "On Optimistic Methods for Concurrency Control", ACM Transactions on Database Systems, Vol. 6, No. 2, June 1981, Pages 213-226.

4. RICHARD E. STEARNS* AND DANIEL J. ROSENKRANTZ: "DISTRIBUTED DATABASE CONCURRENCYC OIITROLS USING BEFORE-VALUES", l981 ACM, Pages
74-83.

5. CHRISTOS H. PAPADIMITRIOU, PARIS C. KANELLAKIS:" ON Concurrency Control by Multiple Versions", 1982 ACM Publication.

6. PHILIP A. BERNSTEIN and NATHAN GOODMAN: "Multiversion Concurrency Control- Theory and Algorithms", ACM Transactions on Database Systems, Vol. 8, No. 4, December 1983, Pages 465-483.

7. HENRY F. KORTH: Locking Primitives in a Database System, Journal of the Association for Computing Machinery, Vol 30, No1, January 1983, Pages 55-79.

8. CHRISTOS H. PAPADIMITRIOU: "On Concurrency Control by Multiple Versions", ACM Transactions on Database Systems, Vol. 9, No. 1, March 1984, Pages 89-99.

9. RONG SUN, GOMER THOMAS: "Performance Results on Multiversion Timestamp Concurrency Control with Predeclared Write sets", 1987 ACM, Pages 177-184.

10. DIVYAKANT AGRAWAL, AMR EL ABBADI, and AMBUJ K. SINGH: "Consistency and Orderability: Semantics-Based Correctness Criteria for Databases", ACM Transactions on Database Systems, Vol 18, No. 3, September 1993, Pages 460-486.

11. MIHALIS YANNAKAKIS: "Issues of Correctness In Database Concurrency Control By Locking", ACM 1981, Pages 363-367.

12.SONAL KANUNGO, R.D. MORENA, "Analysis and Comparison of Concurrency Control Techniques", International Journal of Advanced Research in Computer and Communication Engineering, Vol. 4, Issue 3, March 2015.

13. SONAL KANUNGO, R.D. MORENA, "Comparison of Concurrency Control and Deadlock Handing in Different OODBMS", International Journal of Engineering Research & Technology, Vol. 5 Issue 05, May-2016.