Comparative Study of Concurrency and Replica control protocols in Distributed Environment

Anupam Singh
Pranveer Singh Institute of Technology, Kanpur, India

Raghuraj Singh Suryavanshi
Pranveer Singh Institute of Technology, Kanpur, India

Arun Kumar Singh
Rajkiya Engineering College, Kannauj,India

Abstract

In a Distributed system, data replication is a well-known technique to improve availability and fault tolerance. It improves availability because data is present local to site and it provides fault tolerance because data can be recovered from other sites during failure. Depending on the application, replication may be full or partial. The main problem of replication is to maintain the consistency of replicas allocated at different sites or node. In order to control the replicas several replica control protocols have been proposed. In this paper, an investigation of replication protocols in a distributed environment is done. A brief discussion on consistency models in data replication is also presented.

Keywords: Distributed Database System, Concurrency Control, Consistency, Replication

I. Introduction

Data availability in the database system is a challenge if the database is quite large. A larger database system [1,2] is fundamental requirements in a cloud, grid environment [3], peer to peer network [4] or similar systems. In large scale database system, for data availability at run time replication is one optimal approach. It increases reliability, availability, and performance of the system. The system with extensive database system [5] typically assigns computing replicas near their input data [6]. Replication strategy can be implemented by two ways specifiedas static and dynamic replication system [7,8], which manages replica locations based on session information of requests [9]. In a distributed environment performance and scalability both cannot be optimized if data is replicated [10].

To implement a distributed system, consistency, scalability, performance [11], accessibility and fault tolerance are areas of consideration. We have to make a trade-off between consistency and replication. Level of consistency may be weak or strong. Weaker level consistency can be managed through one-copy- serializability. Consistency can be classified as data-centric or client-centric for data replication. It is necessary to classify the consistency model for data replication techniques [12].

In Section 2, we have discussed consistency models in both ways as client view and server view, taken from the theory of distributeddatabase concurrency control techniques. After this discussion, we have defined concurrency control protocols in Section 3. In this section, we have given a comparison of different protocols to manage consistency in a distributed environment. Implementation properties can also be analyzedin the same sequence. In the next section, wediscuss replication models anddata propagation techniques in a distributed environment. We have presented replica control protocols according to the foundation propagation techniques and replication protocols in Section 5. With the above discussion section, 6 concludes the paper and describes future work.

II. Consistency Model

In this section, we have expressed a number of consistency models which is used to maintain concurrency without compromising on consistency. We have also discussed differences between these consistency models. Different methods to categorizing the consistency models can be taken from [13] and [14]. During system design, consistency model is an important property.

Consistency model can be seen as one of the prime

requirement of a distributed system. Consistency model depends on some criterion which should be satisfied during the assessment of operations [15] of a transaction. These are based on some properties known as ACID consistency requirements [16]. When a transaction changes its state from running to committed all consistency requirements should be satisfied. Data-centric and client-centric are two basic types of defied consistency models [17]. In data-centric, all read and write operations performed on data can be replicated to different workstations or sites immediately. While in client-centric simultaneous updates cannot be handled. To maintain consistency, individual updates will be carried out from one location to another location.

III. Concurrency Control Protocols

In this section, we describe the concurrency control protocols [18]. The ordering of operations of transaction plays a vital role during the concurrent execution of multiple transactions on the same data item. Conflicting operations can be controlled through concurrency control techniques which are categorized as pessimistic and optimistic based on synchronization techniques.Pessimistic algorithms can be further divided as locking protocol, time stamping, and hybrid protocols while optimistic approach can be seen as locking based and time to stamp based algorithms. Locking protocols can be seen as centralized 2PL,Primary copy 2PL, distributed2PL, Majority, and Quorum-based protocols. We consider our conversation on concurrency control models [19].

a) Centralized 2 PL

In centralized 2 PL, a single site is responsible for managing all activities of the lock manager in a distributed system. Before accessing any data item at any site, appropriate locks must be obtained from the central lock manager.

The transaction manager at the site S partitions the global transaction into multiple sub-transactions using information stored into global system catalog. Thus coordinator f transaction is responsible for maintaining consistency of data. If the data item is replicated, then it is the responsibility of transaction coordinator to ensure that all replicas are updated. In this protocol, deadlock detection can be handled easily.

b) Primary Copy 2PL

In Primary replica protocol, one specific replica which is known as the primary replica [20], is responsible for the execution of the submitted transaction. This copy(replica) of data is also responsible for updating other replicas. There are certain requirements [21] of this protocol to maintain consistency as all operations whether read or write should be circulated on all replicas at some time with the same order of their occurrence.

c) Distributed 2PL

Distributed 2PL method implements the lock manager at each site of a distributed system. Thus the lock manager at each site is responsible for maintaining locks on data items that are stored locally. If data is not replicated, then distributed 2PL becomes same as primary copy 2PL.

In this protocol, when a transaction submitted at any site, the transaction coordinator sends a lock request to local lock manager to all participating sites. In distributed 2 PL locks are managed in a decentralized manner which overcomes the drawback of centralized 2PL.

d) Majority locking protocol

Majority locking protocol overcomes the disadvantage of distributed 2 PL that all replicas of a data item must be locked before an update operation. In this algorithm, a lock manager is implemented each site to manage locks for all data items stored at each site locally.

When a transaction is submitted at any site and the same data item is replicated on n sites, then coordinator of transaction sends lock requests to more than half of the sites.Each lock manager determines whether the lock can be granted or not immediately. If the lock compatibility matrix is not satisfied, then the request will be not compatible and the response will be delayed.Majority based replication protocol is used to maintain consistency in a distributed environment which consider the commitment of transaction on more than half of the sites. In this protocol, transaction restart will also be there if more than half of the sites are available but the responses from remote sites (participant sites) to decision maker site (coordinator site) are delayed. It causes restart overhead of the transaction which incurs extra cost. If the majority is not achieved in a certain time period, then the transaction coordinator sends an abort message to all sites.

e) Quorum consensus protocol

This protocol is a generalized version of majority based protocol. To implement this protocol a non-negative weight is assigned to each site. It also assigns integer values to read operation known as reading quorum and write operation known as write quorum of the data item.Read quorum of a data item D is denoted as $D_r$and write quorum is denoted as D_w then the following condition should hold:

$$D_r+D_w> N$$
$$\text{and } 2*D_w> N$$

where N is total weights of all sites in a distributed system.To perform a read operation on a data item D, enough number of data item stored at multiple sites must be read, so that their total weight is greater than or equal to D_r. On the other hand, to perform a write operation enough

number of copies of data item should be written, so that their weight is greater than or equal to D_w. The major advantage of quorum based protocol is that it allows to reduce the cost of reading and write operations selectively assign read and write quorum of the data item. The difficulty with the quorum consensus is that transaction are required to obtain a quorum even to read a data item when there will be data consistency always.

f) Network Partitioning

Owing to communication link failure, the network can partition in a distributed system. If this partition is exactly into two parts, then it is called simple partitioning otherwise it is called multiple partitioning. To maintain consistency is a challenge in the case of network partitioning. During network partitioning, processing involves a trade-off between availability and consistency. Availability can be maximized if there is no restriction on processing of replicated data. The recovery technique that is used to handle network partitioning depends on the particular concurrency control mechanism being used.

Recovery techniques that are used to handle network partitioning, can be classified as Pessimistic and Optimistic approach. Pessimistic focuses on consistency of the database, therefore, do not allow the transaction to proceed in case of network partitioning while optimistic protocol emphasizes on availability that allows proceeding independently in various partitions of the transaction.

IV. Update Propagation Strategies

Update propagation can be defined in two ways [25]. To maintain consistency in the system, all updations should beapplied to all replicas in the same order of operations oftransactions. While updating data object each replicashould be updated according to original replica which isavailable at the site of the submitted transaction [26]. Update on data item can be propagated by following ways: Eager method and lazy method. In general, the eagerprotocolsare known as read-one-write-all (ROWA) protocols.Theseprotocols have not inconsistency during execution ofthe transaction. Any update transaction can read a data itemwhich is the latest one. The remote read operation is not essential.From several replicas, the variations are fully atomic in nature. The update time of a replica is slow when we are using 2PCexecution, eventually which affects the system andperformance of update transaction will be slower. If a copy ofthe data object is not accessible, the update transaction will notterminate meanwhile all copies(replicas) are updatednecessarily. For better performance of the system and overcomeinconsistencybetween several replicas, the lazy protocol isused which assuresstrong mutual consistency [27,28].Data can be accessed from local and remote databases ina distributed environment [29].

In a distributed system, the first updation can take place to the local copy of data object where the transaction was submitted, then all updates are broadcasted to other sites where replicas are stored to maintain consistency. If distribution methods are associated with eager propagation methods, then the distributed concurrency control methods can specifically define the concurrent updates problems [13]. A comparison is given among different propagation techniques in table1.

Table 1. The detailed comparison of update propagation

	Eager update	**Lazy update**	**Centralized**	**Distributed techniques**
Consistency	Strong consistency	Weak consistency	-	-
Updating	Up-to-date with High response time	Lower response time	No synchronization	Better with concurrency control techniques
Performance	Not transactional inconsistency, Changes are atomic	Better response time	Better for some master sites	System availability is higher
Failure	Lower data availability	The inconsistency of data object	High overload and bottleneck problems	Management problem

V. Replication Protocols

Replication [13] isan important phenomenon when data propagation technique is maybe eager or lazy which depends upon a mechanism like primary copy during updations. In the eager approach, the update propagation encompasses by the restrictions of a transaction. A sufficient number of replica updations can notify a commit message to the client.While in Lazyapproach, update propagation can be accomplished [31] when the update process of a local copy is committed at transaction submission site. There is a costly way of providing response time and message overhead in the consistency of the eager approach. An optimized method to prevent these problems is using Lazy replication approach. While the updates of any data object are executed separately, therefore there may be a possibility of data inconsistency [32].

During the transfer of updates either by the eager method or lazy, there are two identified architectures which are required for updations are known as centralized or distributed.

a) Eager Centralized Protocol

If the single site is responsible for all read and write operations, the probability of inconsistency will be minimized. This protocol basically depends on this approach. This protocol ensures higher consistency approach for the

propagation of updations being held. During updations,in perspective of update transaction, all updates are applied to alogical data object. After successful completion of an update transaction, all replica returns the similar value of data item. This is called one serializability replication process.The subcategories of eager centralized are an eager primary copy, single master copy by limited transparency and single master by complete transparency.

In eager primary copy, any data object has a master copy. One replica of the data object is specified as the primary copy. In this case,for controlling serializability condition, there is no single master. All updates have been sent to the specified master directly,In the single master by restricted transparency. To perform a read operation, a shared lock (read lock)acquired on the data item and the read operation is executed after lock conditions are fulfilled. The result of the operation is returned to the client. Also for a write operation, an exclusive lock (writelock)acquired on the data object and the operation is executed after locking it. The result of the write operation is returned to the client in the same way as a read operation. Replica coordination level has been performed by a router in a single master using full transparency. The router sends all operations of the transaction to the master directly.The master computes all operations and returns the result of execution to the client in the same order.

b) Eager Distributed Protocol

The eager distributed approach depends upon local replicas where all updations take place, to maintain consistency all these updates transferred to other replicas. The eager update everywhere is a variant of eager distributed approach.Changes are propagated within the scope of the transaction making the changes. The ACID properties apply to all copy updates.ROWA protocol: Read-one/Write-all. During the execution of a transaction, there is no inconsistencies of the data item.A local copy of data object has the latest value.Changes are atomic in nature. A transaction has to update all sites to maintain consistency. This protocol guarantees longer execution time and lowers data availability.

c) Lazy Centralized Protocol

In this protocol, updates are transferred from master to client which isalmost the same as eager centralized protocol.The propagation process does not take place using the update procedure which is a significant change in this protocol.However, when the transaction completes commit operation successfully but if at the same time another transaction read the value at the client node, it will read old value of data objects. The value at the client side will be not be updated immediately. It will be done eventually when it receives an update message. The categories of lazy centralized are lazy primary copy and single master by restricted transparency. In lazy primary copy, all read and write operation sends to a master. All of the updating results have been sending back to the client. In single master by limited transparency, the update operationsare executed to the master directly. After the commitment an update operation, the new (fresh) transaction set to the client.

d) Lazy Distributed Protocol

Update transaction can execute on each replica in the lazy distributed protocol.It is also noticed that these updates are transferred to the other replicas idly. Lazy update everywhere is a type of lazy distributed protocol. In this type, each read and writes operation is applied on the local copy and the update transactions commit locally.

A detailed Comparison of replication protocols about maintaining consistency is shown in Table 2.

Table 2 Comparison of replication protocols

Replication strategies	Eager Centralized	Lazy Centralized	Eager Distributed	Lazy Distributed
Advantages	The coordination do not need for Update transactions, there are no inconsistencies	The coordination do not need for Update transactions	No inconsistencies	Lowest response times, No centralized coordination
Disadvantages	Extensive response time, Local copies are can only be Read, Only useful with few updates	Inconsistencies	Updates need to be coordinated, Larger response times	Inconsistencies, Updates can be lost

e) Voting Protocols for Data Replication

In a distributed environment, data replication [33,34]at several sites is the common approach in the distributedenvironment to increase data availability and fault tolerance. Due to any failure reason if a site is not available data can still be obtained from the other sites. Commit protocols can be employed to update multiple copies of data. In commit protocols, when a site is not available, the coordinator site sends messages again and again eventually it may decide to abort the transaction, thereby denying access to data. However, it is desirable that the sites continue to operate even when other sites have crashed, or at least one fragment of data

object should continue to operate after the system has been partitioned.

Voting protocols are replica control mechanism by which consistency of data object can be maintained in the distributed environment under network failure. In this approach, votes are assigned to sites. The value of the vote depends on the priority of the site. The site having highest priority will have largest vote count value. In order to do read or write operation sufficient number of votes need to be collected. A voting mechanism can easily handle the problem of a site failure, message failure and network failure than commit protocol.

Due to site or communication link failure, there are several replica control protocols to handle when network partitioning is there. Using the pessimistic approach, consistency can be maintained on reduced availability; it allows one partition to do the updates. When all the partitions are known in advance then static voting will be used. On the other hand, dynamic voting is a more appropriate choice in case of network portioning. This algorithm allowsupdations according to a number of available sites provided it contains the majority of updated copies of the replica.

VI. Conclusion

We have given a review on replication protocol in a distributed environment. In this paper, we have discussed consistency models for data replication protocols in different propagations. We have also presented a brief comparison of replication protocols. By comparison, it has been noticed consistent replication mechanism is an important parameter while managing and implementing database systems. We have also discussed how to handle network partitioning in a distributed environment. We have given a comparative analysis of eager and lazy techniques in centralized as well as the distributed environment.

In future work, we will try to work on other factors that can affect data replication in a distributed environment and also give a formalization of these protocols.

References

[1] S. Çokpınar and T. I. Gundem, "Positive and negative association rule mining on XML data streams in the database as a service concept," Expert Systems with Applications, vol. 39, pp. 7503-7511, 6/15/ 2012.

[2] X. Wang, X. Zhou, and S. Wang, "Summarizing Large-Scale Database Schema Using Community Detection," Journal of Computer Science and Technology, vol. 27, pp. 515-526, 2012/01/01 2012.

[3] N. Saadat and A. M. Rahmani, "PDDRA: A new pre-fetching based dynamic data replication algorithm in data grids," Future Generation Computer Systems, vol. 28, pp. 666-681, 4// 2012.

[4] G. Gao, R. Li, K. Wen, and X. Gu, "Proactive replication for rare objects in unstructured peer-to-peer networks," Journal of Network and Computer Applications, vol. 35, pp. 85-96, 1// 2012.

[5] F.-C. Tseng, "Mining frequent itemsets in large databases: The hierarchical partitioning approach," Expert Systems with Applications, vol. 40, pp. 1654-1661, 4// 2013.

[6] A.-H. Wu, Z.-J. Tan, and W. Wang, "Annotation Based Query Answer over Inconsistent Database," Journal of Computer Science and Technology, vol. 25, pp. 469-481, 2010/05/01 2010.

[7] N. Mansouri, G. H. Dastghaibyfard, and E. Mansouri, "Combination of data replication and scheduling algorithm for improving data availability in Data Grids," Journal of Network and Computer Applications, vol. 36, pp. 711-722, 3// 2013.

[8] N. Mansouri and G. H. Dastghaibyfard, "A dynamic replica management strategy in the data grid," Journal of Network and Computer Applications, vol. 35, pp. 1297-1303, 7// 2012.

[9] M. Tang, B.-S. Lee, X. Tang, and C.-K. Yeo, "The impact of data replication on job scheduling performance in the Data Grid," Future Generation Computer Systems, vol. 22, pp. 254-268, 2// 2006.

[10] X. Liao, H. Jin, and L. Yu, "A novel data replication mechanism in P2P VoD system," Future Generation Computer Systems, vol. 28, pp. 930-939, 6// 2012.

[11] J. E. Westfall, C. M. Kim, and A. Y. Ma, "Locking the virtual filing cabinet: A researcher's guide to Internet data security," International Journal of Information Management, vol. 32, pp. 419-430, 10// 2012.

[12] R. Garcia, R. Rodrigues, N. Pregui, and #231, "Efficient middleware for byzantine fault-tolerant database replication," presented at the Proceedings of the sixth conference on Computer systems, Salzburg, Austria, 2011.

[13] J. Gray, P. Helland, P. O'Neil, and D. Shasha, "The dangers of replication and a solution," SIGMOD Rec., vol. 25, pp. 173-182, 1996.

[14] M. Wiesmann, F. Pedone, A. Schiper, B. Kemme, and G. Alonso, "Understanding replication in databases and distributed systems," in Distributed Computing Systems, 2000. Proceedings. 20th International Conference on, 2000, pp. 464-474.

[15] W. He and I.-R. Chen, "A proxy-based integrated cache consistency and mobility management scheme for a client–server applications in Mobile IP systems," Journal of Parallel and Distributed

Computing, vol. 69, pp. 559-572, 6// 2009.
[16] A. S. Tanenbaum and R. V. Renesse, "Distributed operating systems," ACM Comput. Surv., vol. 17, pp. 419-470, 1985.
[17] D. B. Terry, A. J. Demers, K. Petersen, M. Spreitzer, M. Theimer, and B. W. Welch, "Session Guarantees for Weakly Consistent Replicated Data," presented at the Proceedings of the Third International Conference on Parallel and Distributed Information Systems, 1994.
[18] D. Imbs and M. Raynal, "Virtual world consistency: A condition for STM systems (with a versatile protocol with invisible read operations)," Theoretical Computer Science, vol. 444, pp. 113-127, 7/27/ 2012.
[19] H. Yu and A. Vahdat, "Efficient Numerical Error Bounding for Replicated Network Services," presented at the Proceedings of the 26th International Conference on Very Large Data Bases, 2000.
[20] N. Budhiraja and K. Marzullo, "Tradeoffs in implementing primary-backup protocols," presented at the Proceedings of the 7th IEEE Symposium on Parallel and Distributed Processing, 1995.
[21] N. Budhiraja, K. Marzullo, F. B. Schneider, and S. Toueg, "The primary-backup approach," in Distributed systems (2nd Ed.), ed: ACM Press/Addison-Wesley Publishing Co., 1993, pp. 199-216.
[22] L. E. T. Rodrigues, H. Fonseca, and P. Verissimo, "Totally ordered multicast in large-scale systems," in Distributed Computing Systems, 1996., Proceedings of the 16th International Conference on, 1996, pp. 503-510.
[23] D. K. Gifford, "Weighted voting for replicated data," presented at the Proceedings of the seventh ACM symposium on Operating systems principles, Pacific Grove, California, USA, 1979.
[24] P. Jalote, Fault tolerance in distributed systems: Prentice-Hall, Inc., 1994.
[25] H. Garcia-Molina, "Performance of update algorithms for replicated data in a distributed database," Stanford University, 1979.
[26] W. Zhou, L. Wang, and W. Jia, "An analysis of update ordering in distributed replication systems," Future Gener. Comput. Syst., vol. 20, pp. 565-590, 2004.
[27] E. A. Boiten and J. Derrick, "From ODP viewpoint consistency to Integrated Formal Methods," Computer Standards & Interfaces, vol. 35, pp. 269-276, 3// 2013.
[28] Y. Saito and M. Shapiro, "Optimistic replication," ACM Comput. Surv., vol. 37, pp. 42-81, 2005.
[29] S. Goswami and C. Kundu, "XML based advanced distributed database: implemented on the library system," International Journal of Information Management, vol. 33, pp. 28-31, 2// 2013.
[30] M. T. Ozsu, Principles of Distributed Database Systems: Prentice Hall Press, 2007.
[31] J. Ma, W. Liu, and T. Glatard, "A classification of file placement and replication methods on grids," Future Generation Computer Systems, vol. 29, pp. 1395-1406, 8// 2013.
[32] B. Kemme, R. J. Peris, and M. Patio-Martnez, Database Replication: Morgan and Claypool Publishers, 2010.
[33] R. Suryavanshi and D.Yadav, ”Rigorous Design of Lazy Replication System Using Event-B”. Communications in Computer and Information Science ISSN: 1865-0929, Volume 0306, pages 400-411, Springer, Verlag Germany 2012.
[34] J. Gray, P. Helland, P. E. O’Neil, and D. Shasha. The dangers of replication and a solution. In Proc. of the ACM SIGMOD Int. Conf. on Management of Data, pages 173–182, Montreal, Canada, June 1996.