International Journal of Research Studies in Computer Science and Engineering (IJRSCSE)
Volume. 1, Issue 4, August 2014, PP 38-45
ISSN 2349-4840 (Print) & ISSN 2349-4859 (Online)
www.arcjournals.org

An Extension of Modified Three Phase Commit Protocol for Concurrency Control in Distributed Systems

Mohit Kumar, Anjali Singh, Arjun Singh, Sachin Saxena

Department of Computer Science and Engineering, Invertis University Bareilly, Uttar Pradesh, India

Abstract: *The paper entitled "An Extended Three Phase Commit Protocol for Concurrency Control in Distributed System" proposed an algorithm which included four cases to remove the global abortion problem, but this algorithm solved global abortion caused by some unimportant sites. In this paper we have introduced extended algorithm to resolve global abortion caused by either important (primary) or unimportant (secondary) sites. In this research we have introduce one new table TIT (Transaction Information Table). Previously one transaction was sub-divided into sub-transactions and then each sub-transaction was issued at different sites. There was a variable named FLAG which has two values CONSISTENT or INCONSISTENT, but it failed to solve the global abortion problem caused by important (primary) sites.*

In our algorithm we used a table named TIT (Transaction Information Table) along with the variable FLAG which finds out the solution of the global abortion problem caused by primary sites.

General Terms: *Modified Three Phase Commit Protocol, Distributed System et.al.*

Keywords: *Concurrency Control, Two-Phase Commit Protocol, Three Phase Commit Protocol, Modified Three–Phase commit protocol, Primary Sites, Secondary Sites.*

1. INTRODUCTION

There are two types of commit protocols used for concurrency control one is the **Two Phase commit protocol** and other is **the Three Phase commit protocol**. In two phase commit protocol the sites having more queries become **primary site** and those which are having less queries become secondary sites. There are two phases in two phase commit protocol:

1. *Voting phase:* In this the primary site asks all secondary sites to vote either to commit or to abort. Then secondary sites cast their votes.
2. *Commit phase:* Based on the votes cast by secondary sites, coordinator decides to **commit** if all secondary sites votes commit or **abort** if any of the secondary sites votes to abort and after making decision coordinator notifies the result to all the sites.

Two phase commit protocol has a blocking disadvantage in which either the coordinator or some participating site is blocked but still 2 PC is most widely used in distributed systems.

Three phase commit protocol was introduced as a remedy to the blocking disadvantage of two phase commit protocol. It is the extension of two phase commit protocol. It introduces an extra phase which ensures the non blocking property of this protocol. The site on which transaction is generated becomes **coordinator** and other becomes **cohorts.** The three phases are as follows:

1. *Voting Phase:* Firstly the site at which transaction originates become **coordinator** and then it asks the other cohorts to vote to either commit or to abort. The **cohorts** cast their votes to coordinator and based on the voting done by cohorts, coordinator decides to commits the transactions if all cohorts are in favour of commit. Otherwise it decides to abort even if any of the cohort is against of committing the transaction.
2. *Prepare to commit:* Now in this phase the coordinator notifies its decision to all cohorts .If his decision is to committing the transaction then a message "**enter into ready to commit stage**" is send to all cohorts.

3. *Decision phase:* If the decision made by coordinator is to commit the transaction then it will send **global_commit** to all cohorts and wait for receiving their acknowledgement. And after receiving their acknowledgement it decides to commit the transaction. If the decision made by coordinator is to abort the transaction then it will send **global_ abort** to all sites and aborts the transaction. In this protocol the final decision is made after receiving the acknowledgements.

2. Previous Work

Modified Three Phase Commit Protocol reduces the blocking disadvantage of two phase commit protocol and also overcomes in some failure mode situations. This protocol involves some assumptions-

1. Each site is either a primary site or a secondary site.
2. While choosing the site as primary or secondary we would also consider hardware and software vulnerabilities.
3. All the primary sites do not crash at a time.
4. We use a local clock at each site that runs at regular intervals.

2.1. Algorithm

In distributed systems the site at which transaction is originated becomes the coordinator .The transaction is then broken down into sub-transactions and then each sub-transaction is issued at different sites. So there are 4 cases to be considered-

2.1.1. The Coordinator is a Primary Site

```
If (all the sites vote to commit)
begin
then start Phase two and send ―Enter into ready to
commit stage‖ message to all primary sites and
enter into Phase Three sending global_commit to all
sites and upon receiving the acknowledgement
commiting the transaction.
end
else
If (any of them votes to abort)
then check ― if it is a primary site or not‖
if (yes)
begin
then goto Phase Three and send global abort
end
else
begin
goto Phase Three and send global commit to all sites
who have voted to commit and set the flag with
respect to that database object as ―inconsistent ― at that
primary site which is the coordinator and also at site
which has voted to abort, sets its flag with respect to
that database object as ―inconsistent‖.
```

```
end
```

2.1.2. The Coordinator is a Secondary Site.

```
Check flag status of the database object which will be
in use.
if (flag ="Inconsistent")
begin
Contact the nearest primary site to remove inconsistency and
set the flag as ―consistent‖.
and enter into voting phase.
If (all sites vote to commit)
then
begin
enter into Phase two and send prepare to commit message to all primary sites, after that enter into phase three and send global commit.
 End
    Else
Begin
   Enter into Phase two and send prepare to abort message to all primary sites and then enter into phase three and send global
  abort
   end
     end
 else if (flag =‖consistent‖)
   begin
     enter into voting phase.
     If (all sites vote to commit)
          then
            begin
             enter into Phase two and send prepare to commit message to all Primary Sites, after that enter into phase three and send global commit
             end
    else
            begin
        enter into Phase two and send prepare to abort message to all primary sites and then enter into Phase Three and send global
abort
      end
end
```

2.1.3. A Sub Transaction is Issued at a Primary Site

```
If (sub-transaction is issued at a primary site)
```

```
begin
Continue with the transaction.
end
```

2.1.4. A Sub Transaction is Issued at a Secondary Site

```
Check the flag status of the database object that will be in use
if(flag is ― inconsistent‖)
begin
Then contact the nearest primary site to remove consistency
and set flag as ―consistent‖ and then continue with the
transaction.
End
else
begin
carry on with the transaction.
    end
```

3. Proposed Work- An Extension to Modified Version of Three Phase Commit Protocol

Modified Three Phase Commit Protocol solves the problem of global abortion but do not show good result when any PRIMARY SITE aborts. The site at which the transaction originated becomes the coordinator and other becomes the cohorts. We have introduced new table called **TIT (Transaction Information Table)** which has three fields namely-

Transaction Number: In our protocol every transaction has a unique no for its (transaction) identity.

Value: This field has two values for each site either "COMPLETE" (if the transaction commit at that site) or "INCOMPLETE" (if the transaction abort at that site).

Site – ID: This field has unique no. for each site for its(site) identity.

We also use message passing mechanism-

1. *Message 1-* Coordinator send message to inconsistent site asking the site to complete the transaction. it will transaction no. and id of inconsistent site along with the message.
2. *Message 2-* Inconsistent site send this message to nearest site for updating information. It will send transaction no. along with message.
3. *Message 3-* Inconsistent site sends this message to coordinator after completing the transaction with transaction no.

3.1. Algorithm

The previous algorithm consists of four cases, in first case When the primary site aborts, the whole transaction will be globally aborted, but our algorithm comes as a remedy to this problem. Our algorithm is-

```
If (all the sites vote to commit)
    Begin
    then start Phase two and send "Enter into ready to
    commit stage" message to all primary sites and enter
    into Phase Three sending global commit to all
```

```
      sites and upon receiving the acknowledgement
      committing the transaction.
     End
Else
{
    If (any one of them votes to abort)
       Begin
Go to   phase 3 and send global commit to all sites who      have voted to commit and set the value in table TIT="INCOMPLETE" with respect to that site who have abort and set the value of the FLAG="INCONSISTENT" with respect to that site..
      End
}
```

Now at regular interval when local clock run recover algorithm start:

```
Check flag status of the database object which will be in use .
If (FLAG=INCONSISTENT)
     Begin
Coordinator check the site id and send message1 to  that   inconsistent site.
    End
```

When inconsistent site receives the message1 (Transaction number, Site id).

```
 It sends the message2 to nearest primary/secondary site.
  If (site is not idle)
   {
     Inconsistent site sends the message2 at regular interval for a
     time which is fixed.
        If (site become idle)
             Begin
 Step 1-It check the transaction no. and site ID and all   sent all    transaction related information to that inconsistent site
 Step 2- after receiving the information inconsistent site update database according to it and set TIT's  value= "complete".
Step 3-send message3 to coordinator site and coordinator will set the TIT's value="complete".
Now coordinator site will check the TIT's value field
            If (value! ="incomplete")
                   {
                   Delete the table related to that transaction
                    }
             End
  Else
     Begin
           It sends the message2 to coordinator
```

```
                If (coordinator is idle)
                    Begin
           Repeat step 1, 2 and 3.
        Now coordinator site will check the TIT's value field
            If (value! ="incomplete")
                    {
                    Delete the table related to that transaction
                     }
            End
   Else
    {
   Inconsistent site will send the message2 at regular interval   until all the information will not
received.
     }
}
```

3.2. Analysis of Extended Modified Three Phase Commit Protocol

This Extension to modified version of three phase commit protocol, like the original modified 3 phase commit protocol avoids the blocking limitation of 2 phase commit protocol. It also removes the global abortion problem caused by primary sites.

As we have seen in this protocol that when a transaction is issued and any sites votes to abort then coordinator set the FLAG value =”inconsistent” and TIT ‘s value =”incomplete”.

So there is inconsistencies that have arisen. To remove these inconsistencies we have introduced the concept of local clock which runs at each site. This clock runs at regular intervals and checks for the database objects for which the flag is set as “Inconsistent” Then it issues a transaction to remove this consistency. Thus in regular intervals the local clock runs at each site to remove inconsistency. Recovery transaction will be originated by coordinator through message passing. After receiving the message inconsistent site will update its database by fetching the transaction related information from either their nearest site or the coordinator. After recovery transaction coordinator site delete all the information related to that transaction in the TIT table.

3.3. Advantages of Extended Modified Three Phase Commit Protocol

This Extension to modified version of 3 phase commit proves to be very useful in the cases where we perform some transaction on single database object in distributed systems.

a) Firstly this protocol ensures the commitment of transaction originated at a primary site even when some important site has voted to abort. This is different from the original modified3 phase commit protocol in which if one of the primary site votes to abort then also the whole transaction will be aborted. So this proposed protocol ensures that if the transaction has originated from a primary site and even if any primary sites have voted to abort then also the transaction will be commit

b) Second advantage of this protocol is that this protocol is non blocking protocol. If the coordinator fails then one of the primary site is chosen as the new coordinator and the transaction proceeds.

c) Third advantage of this protocol is that after completing the transaction coordinator will delete the unused information from the TIT table that will save the memory storage.

4. CONCLUSION AND FUTURE WORK

It can be concluded that the Extension to modified version of three phase commit protocol can be used only when there is a transaction that accesses a single database object and ensures commitment of the some transactions that would have otherwise failed in modified Three phase commit protocol so it definitely reduces the probability of a transaction abortion and improve the overall performance of distributed systems.

In future we will try to eliminate number of the messages to reduce the overheads.

ACKNOWLEDGEMENT

The authors are very thankful to their respected Mr. Y.D.S Arya, Pro Vice Chancellor, Invertis University, Bareilly, Mr. Zubair Khan H.O.D (Computer Science Department) and Mr. Ravi Shankar Shukla H.O.D(Information Technology Department), for providing excellent facilities and support in the University campus. We are very much thankful to Poonam Singh for her useful knowledge and time. We are very much thankful to all the technical and non-technical staffs of the Invertis University for their assistance and co-operation. Lastly we thank our friends and parents who directly or indirectly supported us.

REFERENCES

[1] Tanenbaum Andrew S. 2006, Distributed Systems: Principles and Paradigms, 2/E Maarten Van Steen.2006.

[2] Bernstein Dr. Philip. 2001 Concurrency Control, Database Hall of Fame (WS2001)

[3] Krishna Reddy P., Bhalla Subhash, TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING, VOL. 15, NO. 3, MAY/JUNE 2003, Lectures on distributed systems, Distributed Deadlock, Paul Krzyzanowski, Philip Bernstein, Eric Newcomer, Principles of Transaction Processing (for the Systems Professional), Morgan Kaufmann Publishers, January 1997

[4] Philip Bernstein, Vassos Hadzilacos, Nathan Goodman Concurrency Control and Recovery in Database Systems,Addison-Wesley, 1987

[5] Commit Protocols CS60002: Distributed Systems Distributed Systems Pallab Dasgupta Dept. of Computer Sc. & Engg Indian Institute of Technology Kharagpur.

[6] Shanker Udai, Agarwal Nikhil ACTIVE-A Real Time Commit Protocol Scheduling Real-Time Transactions: A Performance Evaluation, Proc. 14th Int'l Conf. Very Large Databases, Aug. 1988.

[7] Agrawal R., Carey M., and Livny M., ªConcurrency Control Performance Modeling: Alternatives and Implications,º ACM Trans. Database Systems, vol. 12, no. 4, Dec. 1987.

[8] Bernstein P., Hadzilacos V., and Goodman N., Concurrency Control and Recovery in Database Systems. Addison- Wesley, 1987.

[9] A. Bestavros and S. Braoudakis, ªTimeliness Via Speculation for Real-Time Databases,º Proc. 15th Real.

[10] Bhargava B., ed. Concurrency and Reliability in Distributed Database Systems. Van Nostrand Reinhold, 1987.

[11] Carey M. and Livny M , ªDistributed Concurrency Control Performance: A Study of Algorithms, Distribution, and Replication, Proc. 14th Int'l Conf. Very Large Databases, Aug. 1988.

[12] Chrysantis P., Samaras G., and Al-Houmaily Y., ªRecovery and Performance of Atomic Commit Processing in Distributed Database Systems,º Recovery Mechanisms in Database Systems. V. Kumar and M. Hsu, eds. Prentice Hall, 1998.

[13] Skeen, D., Stonebraker, M.: A formal model of crash recovery in a distributed system. Concurrency control and reliability in distributed systems, 295–317 (1987).

[14] M. Atif. mCRL2 code for two-phase and three-phase commit protocols. [Online]. Available: http://www.win.tue.nl/~atif/docs/2pc3pc.zip

[15] D. Skeen, “Nonblocking commit protocols,” in SIGMOD Conference, Y. E. Lien, Ed. ACM Press, 1981, pp. 133–142

[16] T. Härder and A. Reuter, “Principles of transaction-oriented database recovery,” ACM Comput. Surv., vol. 15, no. 4, pp. 287–317, 1983.

[17] P. A. Bernstein, V. Hadzilacos, and N. Goodman, Concurrency Control and Recovery in Database Systems. Addison-Wesley, 1987.

[18] Yousef J. Al-Houmaily, Panos K. Chrysanthis: 1-2 PC: The one-two phase atomic commit protocol., 2004.

[19] Silberchatz, Korth, Sudarshan: Datbase System Concepts- Fundamentals of Database, Tata Mc Graw Hills

AUTHORS' BIOGRAPHY

Er. Mohit Kumar is currently working as an Assistant Professor, Department of Computer Science at Invertis University, Bareilly. He has completed his Post-Graduation from Amity University & Research Thesis from Indian Institute of Technology, Banaras Hindu University, Varanasi. He has published three papers in International Journal (I.J.C.A).

Miss Anjali Singh is Final year student of B.tech (CSE) from Invertis University, Bareilly.

Er. Arjun Singh is currently working as an Assistant Professor, Department of Electronics & Instrumentation at Invertis University, Bareilly.

Mr. Sachin Raj Saxena is Final year student of B.tech (CSE) from Invertis University, Bareilly.